Journal of Organizational and End User Computing
Volume 33 • Issue 4 • July-August 2021

Security Threat Modelling With Bayesian Networks and Sensitivity Analysis for IAAS Virtualization Stack

Asvija B., Centre for Development of Advanced Computing (C-DAC), India
Eswari R., National Institute of Technology, Tiruchirappalli, India
Bijoy M. B., Centre for Development of Advanced Computing (C-DAC), India

ABSTRACT

Designing security mechanisms for cloud computing infrastructures has assumed importance with the widespread adoption of public clouds. Virtualization security is a crucial component of the overall cloud infrastructure security. In this article, the authors employ the concept of Bayesian networks and attack graphs to carry out sensitivity analysis on the different components involved in virtualization security for infrastructure as a service (IaaS) cloud infrastructures. They evaluate the Bayesian attack graph (BAG) for the IaaS model to reveal the sensitive regions and thus help the administrators to secure the high risk components in the stack. They present a formal definition of the sensitivity analysis and then evaluate using the BAG model for IaaS stack. The model and analysis presented here can also be used by security analysts and designers to make a selection of the security solutions based on the risk profile of vulnerable nodes and the corresponding cost involved in adding a defense against the identified vulnerabilities.

KEYWORDS

Bayesian Attack Graphs, Cloud Computing, IAAS, Security, Sensitivity Analysis, Virtualization

INTRODUCTION

IaaS (Infrastructure as a Service) has become a prominent service delivery model of modern day cloud infrastructures. Many businesses, especially the MSMEs (Micro, Small and Medium Enterprises) have adopted to use these services from public service providers for their business needs. Virtualization is the key technology enabler behind these massive public cloud infrastructures that can offer commodity virtual servers for business clients over the internet.

However security tops the areas of concern among the users of the cloud services. The inherent nature of virtualization which makes it possible for multiple tenants to share the same physical hardware, brings in a host of challenges related to VM isolation. This would also raise data confidentiality and privacy concerns. In addition to these new security threats unique to virtualized platforms, the cloud infrastructures are also susceptible to the traditional attacks on cyber infrastructures in the Internet world. Thus addressing security in cloud infrastructures becomes a challenge.

In this paper, we present a model for analyzing security threats unique to IaaS virtualized environments. Based on the concept of Bayesian networks, we formulate the threats with the help of an attack graph. Using the principles of sensitivity analysis, we demonstrate the usage of this model

DOI: 10.4018/JOEUC.20210701.oa3

to analyze the impact of various threats and thus implement an optimal security defenses against these threats.

Security analysis is challenging, as the analysts have to deal with the inherent uncertainty with the attack process. The success rate of the attacks varies significantly depending on various factors including the actual targeted setup, the components involved, the strategies employed and the prior defense mechanisms put in place in the infrastructure. A great deal of uncertainty exists regarding the attacker behavior. There are also variations among the different vulnerabilities listed at the corresponding layers. Thus a probabilistic model presents a best approach to capture this inherent uncertainty and carry out analysis to design the security framework for large, critical infrastructures. Bayesian approach can be used effectively to carry out probabilistic reasoning and draw inferences for hypothetical scenarios.

An attack graph is a graphical representation of the security threats in a system with nodes representing the system components and the edges representing the vulnerabilities / the exploits that can be used to reach a specific node. Bayesian Attack Graphs (BAG) are an extension of the attack graphs that employ the Bayesian approach to model the identified security threats in an infrastructure and to draw statistical inferences to various queries, such as the probabilities of attackers reaching particular security conditions by exploiting specific vulnerabilities. The model can also be used to set evidences of particular security violations hypothetically and then evaluate the probable explanations for the same. Specifically, consider an attack graph with *n* nodes. Let us consider $X_i\, where i = 1, \ldots n$, which indicates one of the nodes of the attack graph under consideration. If X_i represents a particular system component or a specific security condition of the system, then the BAG can be used to calculate the unconditional probability distribution $p\left(\mathrm{X}_i\right)$, which indicates the probability of that particular component getting compromised or the probability of reaching a particular security violation state.

Owing to the transitive nature of the virtual machines in a cloud infrastructure, it becomes difficult to impart security fixes to all the reported vulnerabilities across all the components present in the stack. Hence prioritizing the security patches becomes a critical step to identify the crucial components and patches that have to be applied, to bring in a steady state of security to the system. This process necessitates identifying and quantifying the risk exposure of all the components involved in the setup. Sensitivity analysis (SA) on the model can thus be used to prioritize the threats and hence incorporate important defenses in the infrastructure on a priority basis. It can also lead to a better return on investment to the service providers, resulting from the optimized selection of security components to be procured and installed for the resources. The discussion presented in this article helps in achieving this task for IaaS cloud infrastructures. We have used Bayesian attack graphs to depict the attack paths and present a probabilistic model for performing inferences, based on the practical vulnerability scores of the exploits. We demonstrate that sensitivity analysis can be performed on the presented model for multiple scenarios.

The contributions in this paper are unique and useful, because of the following aspects. To the best of our knowledge, security design in virtualized infrastructures has not been analyzed using the concept of sensitivity analysis. While BAGs have been discussed in the context of analyzing conventional network and cyber infrastructures, they have not been employed for analyzing virtualization security in specific. Virtualization brings in new classes of threats that are unique only to such virtualized environments, as detailed in Asvija, Eswari, & Bijoy (2019) and Pék, Buttyán, & Bencsáth (2013). Hence a separate analysis is required to study and analyze the impact of these threats on the overall infrastructure. This paper presents such a model, based on the actually reported threats against different virtualization components. The presented BAG model is drawn by enumerating the reported threats, against the impacted virtualization component(s), which are represented as nodes in the BAG. The exploitability of these threats, to compromise particular components of the virtualization stack, has been modeled using the conditional probabilities that are assigned at each of these nodes. These values have been derived based on the severity metrics of the reported threats. The model can be readily employed by security designers for risk assessment, inferencing, SA and other analytical studies on real-world virtualized infrastructures.

The paper discusses in detail on how SA can be efficiently used for analyzing the security threats in the context of virtualized infrastructures. We have provided illustrative scenarios of using the SA on the presented BAG model. Given a particular component of interest in the infrastructure, sensitivity tornado plots can help the security analysts to identify the most influential nodes on this component. Thus it can be used in designing the defense mechanisms for securing individual critical components in the infrastructure. Further, we have demonstrated the usage of SA, by setting hypothetical evidences on different components / nodes in the infrastructure and study their impact at both local and global levels. This study can be used to assess the impact of a particular security condition, on the other components in the infrastructure. It can also be used to study the change in the probabilities at a particular node, based on different hypothetical evidences. As another illustrative use of SA, we have demonstrated its employability, to identify the possible range of probabilities at a particular node that can lead to a desired target condition. This task is an extension of statistical inference, which will be useful for inferring the possible exploits and their success rates for meeting the conditions of a particular security breach. We have also demonstrated the usage of SA to identify the amount of changes required in the parameters of select components, to satisfy a particular constraint. This study will be useful in scenarios wherein the administrators are presented with a security condition of interest, and the goal of the experiment will be to find the amount of flexibility they have in altering with the defense systems at each of the components in the infrastructure. In this manner, the experiments conducted here demonstrate the employability of SA for a broad spectrum of security analysis tasks, specifically in the context of virtualized infrastructures.

Scope

IaaS infrastructures in general are susceptible to all the security threats that a conventional, physical, online cyber-infrastructure is vulnerable to. Security aspects at all the layers of these cyber-infrastructures have to be taken into consideration, while designing security mechanisms. This includes a gamut of security measures ranging from physical security and access control mechanisms installed at the data-centers to handling the perimeter, network and host level security. In addition to these aspects, virtualized infrastructures introduce additional attack surfaces and hence family of threats that are unique and specific to this layer (Colp, et al., 2011) (Gill & Buyya, 2018). The OS level security, storage security and network security areas are well studied and data center administrators are in general, equipped with knowledge to harden these layers with appropriate defenses periodically (Modi & Acha, 2017). However the importance of virtualization security is often not fully understood by the data center administrators (Tsai, Siebenhaar, Miede, Huang, & Steinmetz, 2011). Further implementing virtualization security is also hard and challenging (Joseph & Mukesh, 2019).

Hence in this article we attempt to develop a framework based on the BAG, to model the security risks that are unique to the virtualization layer. The scope of this article is limited to analysis of the model, based on the specific, identified threats in the virtualization and hypervisor layers. The scope of carrying out sensitivity analysis is also limited to these components.

Organization of The Paper

The rest of the paper is organized as follows. The next section provides a critical analysis of the related work in the field of using the BAG for analyzing security risks and employing sensitivity analysis on BAGs. To provide a background of the virtualization stack, an introduction to IaaS virtualized environments is provided in the next section, along with a discussion on the IaaS security / threat model. The next section introduces the reader to the concepts of Bayesian networks and attack graphs for modelling security attacks. A discussion on carrying out Sensitivity analysis on the Bayesian networks is also presented here. The next section provides a reference Bayesian attack graph model for IaaS environments and the conditional probabilities based on the reported attacks / vulnerabilities. Experimental results from carrying out sensitivity analysis on the model are presented in the last section.

RELATED WORK

Bayesian Networks (BN) and attack graphs have been used to model security threats in cyber networks (Liu & Man, 2005). A mechanism to provide a network security metric, which is a combined function of all the vulnerabilities in the network has been presented by Noel, Jajodia, & Singhal (2007). Wang, Singhal, & Jajodia (2007) have introduced the idea of using combining functions to assess the overall impact of the vulnerabilities in the network. The concept of modeling uncertainty with the number of attackers likely to exploit a given vulnerability in a network, has been introduced by Wang, Islam, Long, Singhal, & Jajodia (2008).

A number of authors have discussed on inference techniques in Bayesian attack graphs. Table 1 summarizes the related work in the field of inferences in BAGs and using Sensitivity analysis in them. Both approximate and exact inference techniques and various algorithms have been discussed in detail (Munoz-Gonzalez, Sgandurra, Paudice, & Lupu, 2017; Munoz-González, Sgandurra, Barrere, & Lupu, 2017). The concept of using Genetic Algorithms (GA) for security analysis in BAGs has been introduced by Poolsappasit, Dewri, & Ray (2012).

Sensitivity analysis in general, is a useful approach in identifying important uncertain factors in many real world scenarios. They have been employed to study models from multiple scientific domains including environmental (Razavi & Gupta, 2019) and hydrological modelling (Li, et al., 2019), chemical engineering (Xie, Schenkendorf, & Krewer, 2019) and many other fields (Marchioni & Magni, 2018; Salim, Ioannidis, Penlidis, & Górecki, 2019). Sensitivity analysis using Bayesian Networks has been discussed by Chan & Darwiche, (2004) and Laskey (1995).

Although not through the means of Sensitivity analysis, the concept of identifying critical nodes in the attack graph and thus allowing a subset of defenses to be implemented by the administrators, has been introduced by Dewri, Poolsappasit, Ray, & Whitley (2007) and Poolsappasit, Dewri, & Ray (2012). The authors introduce new cost metrics and formulate the scenario as a multi objective optimization problem, thereby employing GA for solving it.

The use of BAGs for modeling threats in virtualized systems has been presented by us in (Asvija, Eswari, & Bjioy, 2019). We build upon these principles to present a concrete model for IaaS and employ sensitivity analysis to measure the impact of small changes in the parameters on the identified target probabilities.

Our Contribution

The main contributions of this article are as follows:

We use the concept of Bayesian Networks to model and carry out sensitivity analysis on the security in virtualized platforms. Virtualization security has become an area of prime importance, as it forms the core technology enabling modern day cloud computing infrastructures. We use the concept of Bayesian Networks to build the attack graph for virtualized platforms and carry out a detailed sensitivity analysis on the various components of the IaaS virtualization stack. To the best of our knowledge, our work pioneers in carrying out a sensitivity analysis on IaaS virtualization security components, that can be extremely useful for security architects to evaluate their infrastructures.

The approaches described in works (Munoz-Gonzalez, Sgandurra, Paudice, & Lupu, 2017; Munoz-González, Sgandurra, Barrere, & Lupu, 2017; Poolsappasit, Dewri, & Ray, 2012) are generic in discussion with analysis carried out on example networks or test networks. However the vulnerabilities considered in this article, to derive the local conditional probability values and to carry out the sensitivity analysis on the model, are specific to the IaaS virtualization stack, and thus have a greater practical significance.

Table 1. Comparison of related work on BAG and Sensitivity analysis

Research work	Feature	Remarks
Munoz-Gonzalez, Sgandurra, Paudice, & Lupu, 2017 ; Munoz-González, Sgandurra, Barrere, & Lupu, 2017	Employing exact inference techniques to analyze BAGs.	The approach is generic in discussion with example networks and authors do not take into the account of specific threats of virtualized infrastructures. No discussion on Sensitivity analysis.
Noel & Jajodia, 2014	Proposed a suite of deterministic metrics for analyzing network attack graphs.	The presented metrics are well suited for carrying out a comparative analysis of different attack graphs. They might not be effective for carrying out risk analysis of a single network that is under consideration. Further uncertainty is not taken care as probabilistic models are not employed.
Dewri, Poolsappasit, Ray, & Whitley (2007); Poolsappasit, Dewri, & Ray (2012)	Propose a method for identifying critical nodes in a graph	SA is not employed. Scenario is modeled as a multi-objective optimization problem and Genetic algorithms (GA) are employed for solving them. No discussion on specific threats of virtualized infrastructures.
Razavi & Gupta, 2019; Li, et al., 2019; Xie, Schenkendorf, & Krewer, 2019; Marchioni & Magni, 2018; Salim, Ioannidis, Penlidis, & Górecki, 2019	Sensitivity analysis used in multiple scientific domains	These works discuss the usage and suitability of SA in multiple scientific domains. However they do not discuss SA in the context of Bayesian approaches and virtualization security.
Laskey, 1995	Sensitivity analysis in Bayesian Networks (BN)	Introduces a method of computing Sensitivity values in BNs. Seminal work in this area presenting general analytical methods to determine sensitivity values. However it presents no specific details on analyzing attack graphs or in the context of virtualization security.
Chan & Darwiche, 2004	Sensitivity analysis in Bayesian Networks	Provides a good overview of analytical approaches to carry out SA in BNs for single & multiple parameter changes. Authors also present a method to identify the weakest uncertain evidence. However the discussion is presented with example networks and authors do not consider modeling security or specific threats of virtualized infrastructures.

We present a practical model that has been created using the BN modeling software, which can be directly employed by system administrators for carrying out risk analysis of IaaS platforms. The local conditional probability values used in the analysis are derived from practical vulnerabilities listed in the National Vulnerability Database (NIST, 2018). Multiple sensitivity coefficients and indices have been explained and experimented, which can give cloud architects a better insight in selecting the hardening measures with the available constrained budget for implementing them.

Further, models based on static analysis and deterministic approaches such as the ones described in (Dewri, Poolsappasit, Ray, & Whitley, 2007; Idika & Bhargava, 2012; Noel & Jajodia, 2014), ignore the dynamic aspects of the attack process. The uncertainty inherent with the security attacks, calls for the employment of a dynamic model which can be revised based on new inputs and changes in the behavior. The BAG model presented here can be fed with new incident data and evidences. These can be used to carry out fresh analysis to obtain new set of results.

IAAS VIRTUALIZED ENVIRONMENTS

IaaS is a popular service delivery model in the cloud computing paradigm, in which the service providers offer computing and storage resources as services to be used by the end users for their business needs. These infrastructures rely on platform virtualization technology to offer *virtual machines* (VM) as services to the end users. Figure 1. depicts the components of a typical IaaS stack. Virtualization facilitates the creation of a logical view of a virtual machine that can encapsulate virtual hardware resources such as the virtual CPU (vCPU), virtual memory (vMem) and virtual disk (vDisk) along with an operating system (OS) to make them functional. The technology relies on sharing the underlying physical hardware among multiple *tenant* VMs. The physical machine facilitating this model is called as a *host* machine, while the VMs are referred to as the *guest* machines. The OS on the physical hardware is referred to as the *Host OS*, while the individual VMs run their own *guest OS*. The software component that facilitates the creation and management of the virtual machines is called the *hypervisor.* A typical large IaaS infrastructure, has multiple host machines pooled together, each running with its own hypervisor and capable of hosting multiple VMs. The *cloud middleware* is a software component that facilitates the co-ordination and management of multiple hosts, network and storage resources in the cloud setup. A *management server,* is typically setup to perform the administration tasks with the help of the cloud middleware. An image repository is a storage service which hosts the *image* files, that can be used by the end users to select their choice of guest OS and applications to execute on their own VMs.

IAAS Security

A benign user of the IaaS services from the cloud is granted access only to the VMs created / owned by him. Typically the virtualization technology is expected to guarantee full isolation among multiple tenants that share the same underlying hardware. However a break-in to the isolation among the VMs can result in a severe security compromise and data breach, as each co-hosted VM can be owned by different end users of the cloud services. We introduce the general threat model in this section. A detailed account of the individual attacks at various surfaces is further given in the section detailing the BAG and local CP values for IaaS virtualized environments.

Figure 2. depicts the threat model for attacks on VMs in a typical IaaS setup. The figure captures the possible attacks on the VMs during its lifetime in the infrastructure. Cloud platforms allow the end users to create and destroy virtual instances of resources easily. Compromised hosts and network nodes can be used for distributed denial of service and Botnet attacks on the infrastructure. A3 represents the attacks that could be arising out of modification / tampering of the virtual machine images and infecting them with Trojans. A4 represents attacks on data at rest, specifically on the image storages. A1 and A2 are the attacks that can be instrumented on the virtual resources at runtime. These attacks reveal confidential information to the attacker from a co-resident VM and this leakage cannot be eliminated by deploying access control policies. A5 and A6 represent the family of attacks where an under-privileged VM can gain access to the privileged kernel space or I/O space maliciously. A detailed enumeration of the reported attacks against each of these identified threat families is presented in the following sections.

It is thus evident from the above model, that IaaS virtualized environments expose multiple surfaces through which attackers can cause a security violation. Hence, the task of analyzing the IaaS security model becomes extremely important. We introduce the approach of threat assessment and sensitivity analysis using the Bayesian Networks and attack graphs in the next section.

BAYESIAN NETWORKS AND ATTACK GRAPH MODELLING

Bayesian Networks (BN) can be employed to model and analyze uncertainty in systems. They can be denoted as graphical representations comprising of nodes and directed edges, with the nodes

Figure 1. Components of the IaaS stack

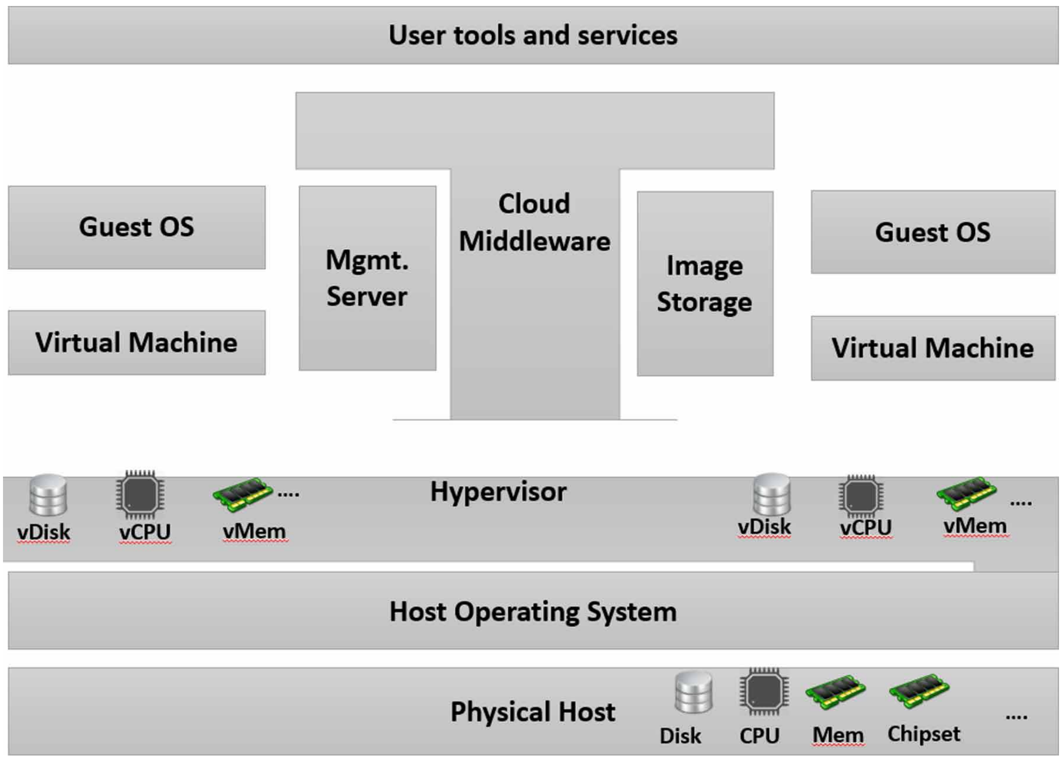
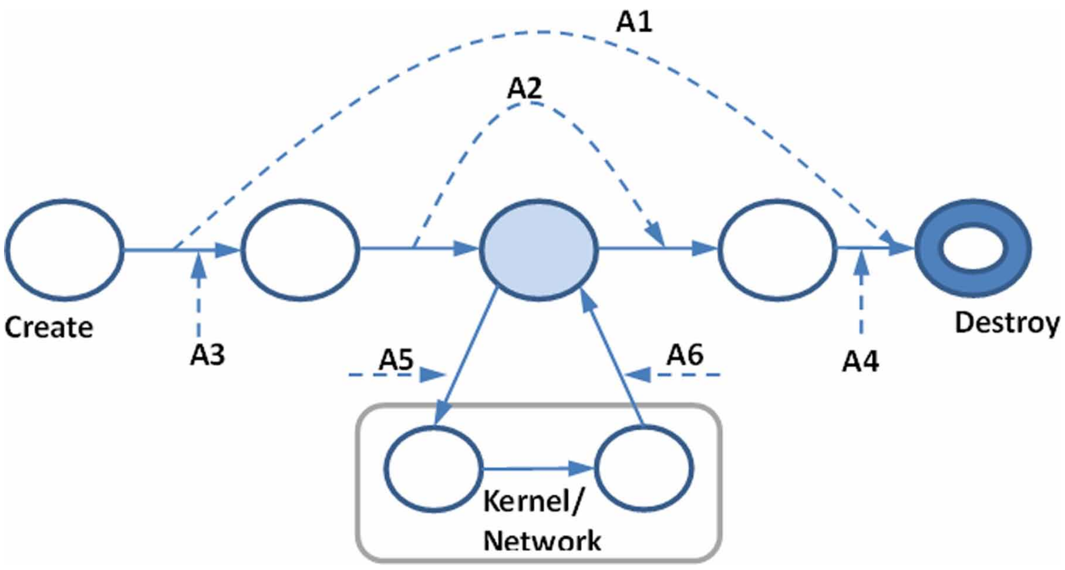


Figure 2. Threat model for attacks on VMs in IaaS setup



signifying the random variables in the system and the edges representing the conditions / correlations between them. The edges in the graph are directed in such a way that they do not result in cycles (Xie, Li, Ou, Liu, & Levy, 2010).

To define a BN formally, the network is a pair G,˜ . G represents the graph component that comprises of a set of nodes/vertices V and a set of directed edges E. Each vertex in the set V represents a random variable in the modeled system, while a directed edge indicates the conditionality between the vertices. An edge e from a vertex v_1 to v_2 in a BN indicates the occurrence of v_2 given the condition v_1. Each variable in the network is independent of its non-descendants, with the state of its parent variables given (Friedman, Geiger, & Goldszmidt, 1997). ˜ in the pair G,˜ , denotes the set of quantifying parameters for the BN.

Let X represent a vector of random variables involved in the network: $X = \{X_1, X_2, \ldots X_n\}$. The uppercase letter X_i shall be used to denote a random variable from this vector, while the lowercase letter x_i is used to indicate a value to this variable. Let PA_{X_i} denote the set of parents of X_i. For the root nodes in the graph, PA_{X_i} is an empty set. Given the particular values for PA_{X_i}, $p\left(X_i | PA_{X_i}\right)$ is the local conditional distribution for the random variable X_i. ˜ contains the parameters:

$$\theta_{x_i|PA_{x_i}} = p\left(x_i | PA_{x_i}\right),\ \forall x_i \in X_i\ and\ \forall PA_{x_i} \in PA_{X_i} \tag{1}$$

The joint probability for the Bayesian Network is thus defined as a product of these $p\left(X_i | PA_{X_i}\right)$.

$$p\left(\mathrm{X}_1,\ \mathrm{X}_2,\ \ldots,\ \mathrm{X}_n\right) = \prod_{i=1}^{n} p(X_i | PA_{X_i}) \tag{2}$$

Bayesian Attack Graph (BAG) can be interpreted as the notion of extending the concept of Bayesian Networks to depict the security attack scenarios on a network / infrastructure. Each node in the attack graph represents the network nodes or the components involved in the infrastructure, while the edges represent the vulnerability or the exploit to carry out the designated attack. The nodes can also depict security states of an infrastructure under consideration. Different vulnerabilities / exploits and the resulting violations can be modeled through the attack paths in the graph. Using the graph, one can also visually identify the possible attack paths in the system to reach a possible security violation condition. Conditional probabilities shall be assigned at every node indicating the probability of the attack or using the exploit, given the conditions depicted as parent nodes in the graph. The probability distribution function at the individual nodes can be defined based on two approaches. The AND decomposition in which each individual exploit has to be successful to result in compromising a child node, while the OR decomposition stipulates that any of the exploit from the parent to the child can be sufficient to compromise the targeted child node. System administrators and security professionals with wide expertise in the field can make the decisions to employ either of the schemes while modeling the infrastructure. The formal definition (Poolsappasit, Dewri, & Ray, 2012) of the local conditional probability distribution function for the AND scheme can be given as follows:

$$p\left(x_i | PA_{x_i}\right) = \begin{cases} 0, & \exists X_i \in PA_{X_i} \mid X_i = 0 \\ p\left(\bigcap_{X_i = 1} e_i\right), & otherwise \end{cases} \tag{3}$$

The product rule can be used to compute the probability in this case. For the OR decomposition, $p\left(x_i|PA_{x_i}\right)$ can be defined as follows (Poolsappasit, Dewri, & Ray, 2012):

$$p\left(x_i|PA_{x_i}\right) = \begin{cases} 0, & \forall X_i \in \ PA_{X_i},\ X_i = 0 \\ p\left(\bigcup_{X_i.=1} e_i\right), & otherwise \end{cases} \tag{4}$$

It is also shown that (Liu & Man, 2005) for the OR decomposition, the joint probability is given by the noisy-OR operator as defined as below:

$$p\left(\bigcup_{X_i.=1} e_i\right) = 1 - \prod_{X_i.=1} \left[1 - p\left(e_i\right)\right] \tag{5}$$

Sensitivity Analysis

Sensitivity Analysis (SA) can be used to identify how sensitive the changes in probabilities of the query nodes are to the change in parameters (Pollino, Woodberry, Nicholson, Korb, & Hart, 2007). It can also reveal the degree of relationship between the local parameters and the global conclusions in the model (Chan & Darwiche, 2004). Relevant parameters to a specific query can be identified with SA.

Sensitivity co-efficients are employed to quantify the change in the output ω, for a given change $”\rho$ in the input ρ. The sensitivity index $\zeta_{\rho\omega}$ can be defined as (Loucks & Beek, 2005):

$$\zeta_{\rho\omega} = \frac{\left\{\omega\left(\rho_0 + ”\rho\right) - \omega\left(\rho_0 - ”\rho\right)\right\}}{2”\rho} \tag{6}$$

where ρ_0 represents the base value of ρ and the output ω is a function of the input ρ. If we allow the variations to ρ_0 on the negative and positive sides to be represented by i and j respectively, then we can define $\zeta_{\rho\omega}$ as follows:

$$\zeta_{\rho\omega} = \frac{\left\{\left|(\omega_0 - \omega_i) / \left(\rho_0 - \rho_i\right)\right| + |\,(\omega_0 - \omega_j) / \left(\rho_0 - \rho_j\right)|\right\}}{2} \tag{7}$$

Or

$$\zeta_{\rho\omega} = \max\{\left|(\omega_0 - \omega_i) / \left(\rho_0 - \rho_i\right)\right| + |\,(\omega_0 - \omega_j) / \left(\rho_0 - \rho_j\right)|\} \tag{8}$$

Sensitivity Analysis In Bayesian Networks

Sensitivity analysis can be carried out with multiple methods. A simple and intuitive method involves identifying suitable ranges for every parameter in the model. Then the parameters for a chosen variable can be modified from the lowest to the highest identified range value, by keeping the values for the other variables unchanged. The resulting impact on the target variable can be recorded. In another method, partial derivatives of the target variable can be computed with respect to each of the parameters. It has been shown that (Laskey, 1995), this method is particularly advantageous in case of Bayesian networks and can also be used to identify the variables that can cause significant impact.

Consider a Bayesian Network that characterizes a probability distribution p and having a variable represented by a node X_i with its parent nodes PA_{X_i}. Let $\theta_{x_i|PA_{x_i}}$ indicate a parameter in the conditional probability table of X_i, where X_i represents the random variable itself and x_i represents a value to this variable. The sum of the conditional probabilities at a given node should be equal to a value of 1. Thus any modifications to the value of $p(x_i \mid PA_{x_i})$ effected by the change $\theta_{x_i|PA_{x_i}}$, will result in a corresponding change in the conditional probabilities $p(\overline{x_i} \mid PA_{x_i})$, for all $\overline{x_i} \neq x_i$. The function that assigns these probabilities according to the above rule is said to be a variation function. If the change $\theta_{x_i|PA_{x_i}}$ is added to $p(x_i \mid PA_{x_i})$, then a corresponding value is reduced from $p(\overline{x_i} \mid PA_{x_i})$ for every $\overline{x_i} \neq x_i$, to keep the sum of the conditional probabilities at the given node equal to a value of 1. Thus the new probabilities can be represented as follows:

$$p_{New}(x_i \mid PA_{x_i}) = p_{Old}(x_i \mid PA_{x_i}) + \theta_{x_i|PA_{x_i}} \tag{9}$$

$$p_{New}(\overline{x_i} \mid PA_{x_i}) = p_{Old}(\overline{x_i} | PA_{x_i}) + \frac{\theta_{x_i|PA_{x_i}} \, p_{Old}(\overline{x_i} | PA_{x_i})}{1 - p_{Old}(\overline{x_i} | PA_{x_i})} \tag{10}$$

SA can reveal the impact on the joint probability distribution $p_{New}(X_\tau | x_\varepsilon, \, , \,)$ to the corresponding changes in parameter θ, where $(X_\tau | x_\varepsilon)$ represents the condition of a target variable X_τ, when given with the evidence x_ε. If $\theta_{x_i[\lambda]}$ represents the λth element in the parameter vector for the random variable X_i, then the *sensitivity value* can be defined as follows (Laskey, 1995):

$$SI_{i[\lambda]} = \frac{\partial p_{New}(x_\tau | x_\varepsilon, \, , \,)}{\partial \theta_{x_i[\lambda]}} \Bigg|_{\theta_{x_i[\lambda]} = 0} \tag{11}$$

If $E(X)$ represents the expected value of the random variable X, then Laskey (1995) has shown that the partial derivative to be as follows:

$$\frac{\partial p_{New}(x_\tau | x_\varepsilon, \, , \,)}{\partial \theta_{x_i[\lambda]}} = p_{New}(x_\tau | x_\varepsilon, \, , \,)\left(E\left[F_{i[\lambda]} \mid x_\tau, x_\varepsilon\right] - E\left[F_{i[\lambda]} \mid x_\varepsilon\right]\right) \tag{12}$$

where:

$$F_{i[\lambda]}\left(X_i,\ PA_{X_i},\theta_i\right)=\frac{\partial\log\left(p_{New}(X_i|\ PA_{X_i},\theta_i)\right)}{\partial\theta_{x_i[\lambda]}} \tag{13}$$

One of the goals of carrying out sensitivity analysis on Bayesian networks is to identify all the changes to $\theta_{x_i|PA_{x_i}}$, which will result in satisfying the query constraint $p_N\left(z\ |e\right)\geq p$, where z is some event given the evidence e with p_{New} representing the new probability distribution after making the changes to $\theta_{x_i|PA_{x_i}}$. Let p_{Old} represent the old distribution before applying the changes and the assumption that p_{Old} and p_{New} are over the same set of worlds W. The goal is to narrow down on the changes which will result in a minimal value for the distance measure between p_{Old} and p_{New}. The distance measure itself is given by the following equation (Chan & Darwiche, 2005):

$$S\left(p_{Old},p_{New}\right)\overset{def}{=}\ln\max_W\frac{p_{Old}\left(W\right)}{p_{New}\left(W\right)}-\ln\min_W\frac{p_{Old}\left(W\right)}{p_{New}\left(W\right)} \tag{14}$$

Using the measure $S\left(p_{Old},p_{New}\right)$, one can infer the bounds on the amount of change in the value from p_{Old} to p_{New} for the query of the form $q_1\mid q_2$:

$$\frac{p_{Old}\left(q_1|q_2\right)e^{S\left(p_{Old},p_{New}\right)}}{p_{Old}\left(q_1|q_2\right)\ (e^{S\left(p_{Old},p_{New}\right)}-1)+1}\ \geq\ p_{New}\left(q_1|q_2\right)\ \geq\frac{p_{Old}\left(q_1|q_2\right)e^{-S\left(p_{Old},p_{New}\right)}}{p_{Old}\left(q_1|q_2\right)\ (e^{-S\left(p_{Old},p_{New}\right)}-1)+1} \tag{15}$$

In order to reduce the variation on the global changes as a result of the modifications in local parameters, one can attempt to minimize $S\left(p_{Old},p_{New}\right)$ (Chan & Darwiche, 2005).

BAG AND LOCAL CP VALUES FOR IAAS VIRTUALIZED PLATFORMS

Threat Actors and Attacks

We consider three types of threat actors in our discussion of security threats to the IaaS environments. A malicious guest user (MGU) is a subscriber of cloud services who intends to carry out malicious activities on the infrastructure. A remote attacker (RA) is an outsider who does not hold valid access credentials in the cloud, and is attempting to break-in to the infrastructure. A malicious administrator (MA) is an insider to the cloud infrastructure, who has turned malicious to compromise the setup.

A detailed survey of the publicly reported attacks by these various threat actors is presented by Asvija, Eswari, & Bjioy (2019) and Pék, Buttyán, & Bencsáth (2013). A remote attacker can trigger virtualization detection attacks (Rutkowska J. , 2004; Brengel et al., 2016) to identify if the targeted resources run a set of vulnerable software components. These vulnerabilities in the hypervisor or the applications deployed in VM can be used by the attacker to gain a privilege access to the infrastructure (Geffner, 2015; NIST, 2018). A malicious guest user can obtain escalated privileges by exploiting the vulnerabilities in the system firmware (Gorobets et al., 2015; Kallenberg et al., 2014). Further MGUs can construct covert side channels that can reveal sensitive information between co-resident VMs. This is can result in a serious data theft and DoS attacks on the cloud (Inci et al., 2016; Yarom & Falkner, 2014). As an insider, a malicious admin can have direct access to the hosts. A planted

VM based rootkit (Desnos, Filiol, & Lefou, 2011), can compromise the hypervisor and all the VMs hosted in the machine. Firmware attacks can also result in the hypervisor and thus multiple tenants in a compromised state (Gorobets et al., 2015).

Reference BAG Model For IAAS Attacks

Bayesian Attack Graphs can be used to carry out security assessment of large infrastructures such as public and private cloud computing platforms that leverage virtualization technologies. The reference model graph for virtualized platforms, introduced by Asvija, Eswari, & Bjioy (2019), can be used for carrying out a detailed sensitivity analysis. The model, as shown in Figure 3, has been developed based on the publicly reported attacks on the various components of the hypervisor and virtualization stack for the IaaS. The vulnerabilities which have been marked as either *Critical* or *High* are considered. The nodes of the graph represent the state of the components of the virtualization stack that can get compromised due to an attack on the infrastructure, while the edges represent the attack families themselves. Figure 4. displays the Conditional Probability (CP) values assigned at each of the nodes in the BAG. The method used to derive the CP values is based on the vulnerability scores from the NIST's Common Vulnerability Scoring System (CVSS) (Mell, Scarfone, & Romanosky, 2007), as suggested by (Poolsappasit, Dewri, & Ray, 2012). Munoz-Gonzalez et al. (2017) have shown that the exploitability metric is more suitable to derive the CP values in a BAG, as it gives a direct measure of the ease with which the attack can be carried out to compromise a component in the infrastructure. We have enlisted the reported severe and critical vulnerabilities against these attack surfaces in the virtualization stack from the National Vulnerability Database (NVD). The identified vulnerabilities with their CVE identifiers and the CVSS scores have been enlisted in the Appendix section of this article. The corresponding CVSS scores are used to derive the CP values at the individual nodes of the BAG. As shown in (Poolsappasit, Dewri, & Ray, 2012), the exploitability score has the components of the attack access vector A_{av}, access complexity A_{ac} and authentication instances A_{ai}. With these values available for a vulnerability v, the probability value has been computed as follows:

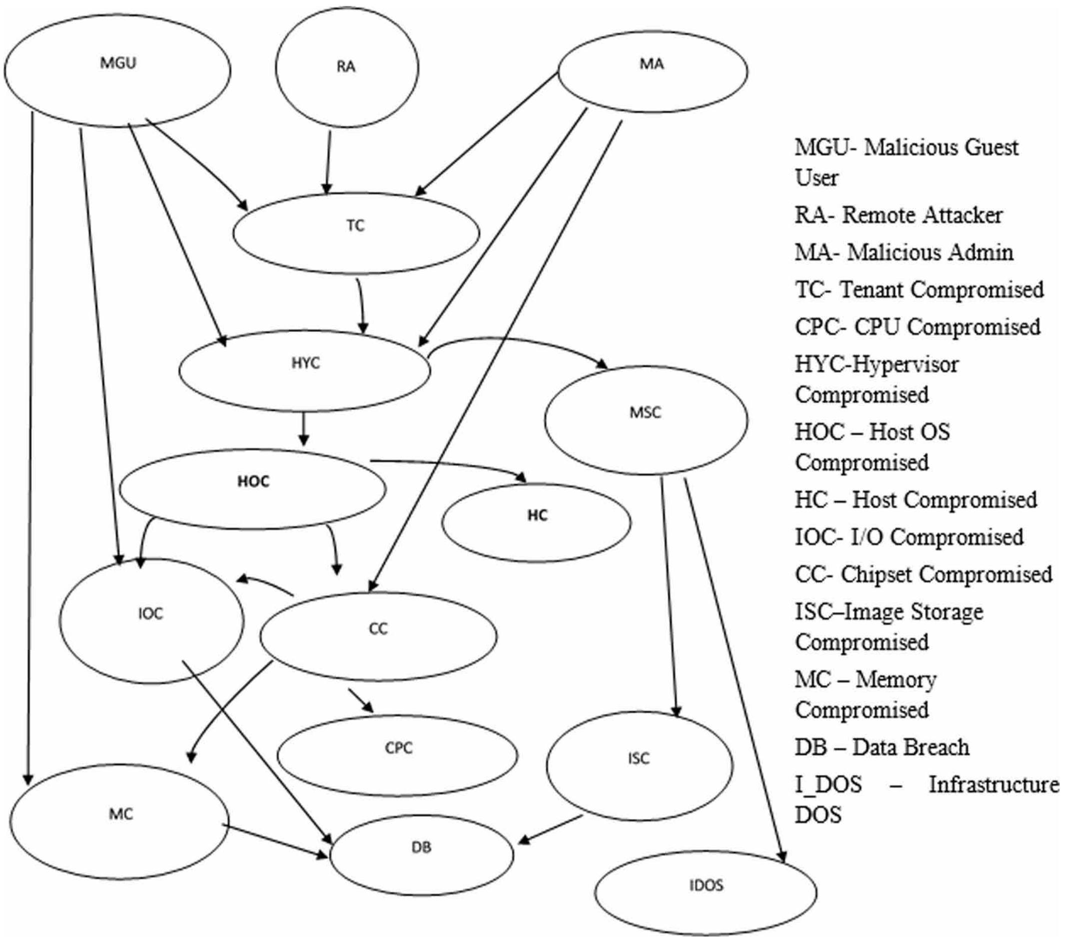
$$p\left(v\right) = 2A_{av}A_{ac}A_{ai} \tag{16}$$

EXPERIMENTAL RESULTS

The Bayesian Network has been modeled using two tools namely the UCLA Senstivity Analysis, Modeling, Inference and More (SamIam) (Darwiche, et al., 2017), and the GeNIe modeler (BayesFusion, L. L. C., 2017). The model was created and experiments were carried out on a system with Intel Xeon E3 family 1226 version 3 (3.3 GHz) CPU having 16 gigabytes of main memory with Windows 10 OS and Oracle JRE 1.8. The GeNIe 32 bit Academic version was used from BayesFusion with version 2.4.4519.0.

Sensitivity analysis can reveal the impact on the various sub components of the stack, when subject to a security incident. To illustrate this, we used the model presented here to carry out sensitivity analysis using the GeNIe modeler on the node labeled Chipset_Compromised (CC). Figure 5 depicts the tornado plot for sensitivity values for the target condition CC=true. Sensitivity tornado plots can reveal the extent of impact of other components in the BAG, on a particular chosen node. The plots demonstrate the sensitivity values for the various nodes in the BAG, in the decreasing order of their impact on the target node of interest. These plots can also give an indication of the impact of the specific vulnerabilities at a given BAG node. Such a plot can be used while prioritizing the defense mechanisms for individual critical components or for preventing specific classes of security breaches in the infrastructure. With Figure 5, one can conclude that the most sensitive node for the current target

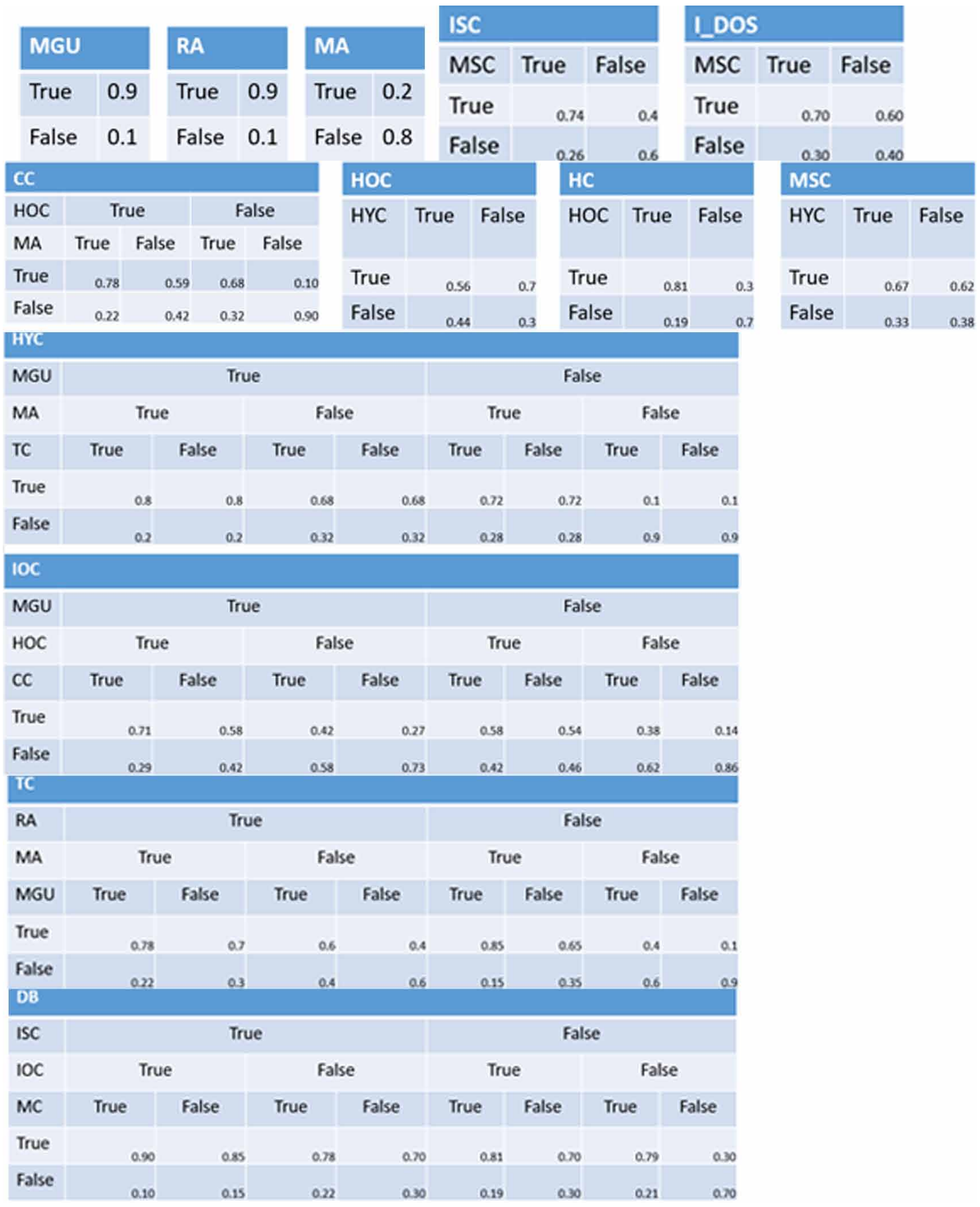
Figure 3. Threats in virtualized platforms modeled as a BAG



condition is the HOC. Thus the exploitability / defense at the HOC node, has the highest impact on the CC node. The next sensitive node is the Malicious Admin (MA) node. This can be attributed to the large family of chipset attacks can be carried out by a malicious admin as an insider. The other sensitive nodes include the hypervisor compromised state pointing to the presence of VM based Rootkits (Desnos, Filiol, & Lefou, 2011) and attacks on hypervisors through firmware (Gorobets et al., 2015). The malicious guest user (MGU) node also figures in the sensitivity diagram, indicating the possibility of VM escaping attacks resulting in compromised secure boot process of the system (Kallenberg et al, 2014).

Figure 6. shows another illustration of the sensitivity tornado plot for the target condition IO Compromised = true. The current value for the target condition is 0.51. The most sensitive nodes for this condition are the Host OS Compromised (HOC) and the CC nodes, indicating the criticality of IO attacks through these channels (Pék et al., 2014; Richter et al. 2014). The Hypervisor Compromised (HYC) node also appears as sensitive, owing to the critical hypervisor vulnerabilities such as the VENOM (Geffner, 2015), which could be exploited to target IO channels in virtualized systems. In this manner, the presented model can be used for identifying the most sensitive nodes for a given target condition.

Figure 4. CPT for the nodes in the BAG derived from CVSS exploitability metrics



MGU	
True	0.9
False	0.1

RA	
True	0.9
False	0.1

MA	
True	0.2
False	0.8

ISC		
MSC	True	False
True	0.74	0.4
False	0.26	0.6

I_DOS		
MSC	True	False
True	0.70	0.60
False	0.30	0.40

CC				
HOC	True		False	
MA	True	False	True	False
True	0.78	0.59	0.68	0.10
False	0.22	0.42	0.32	0.90

HOC		
HYC	True	False
True	0.56	0.7
False	0.44	0.3

HC		
HOC	True	False
True	0.81	0.3
False	0.19	0.7

MSC		
HYC	True	False
True	0.67	0.62
False	0.33	0.38

HYC								
MGU	True				False			
MA	True		False		True		False	
TC	True	False	True	False	True	False	True	False
True	0.8	0.8	0.68	0.68	0.72	0.72	0.1	0.1
False	0.2	0.2	0.32	0.32	0.28	0.28	0.9	0.9

IOC								
MGU	True				False			
HOC	True		False		True		False	
CC	True	False	True	False	True	False	True	False
True	0.71	0.58	0.42	0.27	0.58	0.54	0.38	0.14
False	0.29	0.42	0.58	0.73	0.42	0.46	0.62	0.86

TC								
RA	True				False			
MA	True		False		True		False	
MGU	True	False	True	False	True	False	True	False
True	0.78	0.7	0.6	0.4	0.85	0.65	0.4	0.1
False	0.22	0.3	0.4	0.6	0.15	0.35	0.6	0.9

DB								
ISC	True				False			
IOC	True		False		True		False	
MC	True	False	True	False	True	False	True	False
True	0.90	0.85	0.78	0.70	0.81	0.70	0.79	0.30
False	0.10	0.15	0.22	0.30	0.19	0.30	0.21	0.70

Further, evidence variables can be set on the model, and the sensitivity analysis can be re-run to obtain the fresh results based on the set evidences. Hypothetical evidences can be set in the model to analyze their impacts on the parameters at different nodes in the BAG. Evidences can represent the assumption of a particular security breach occurring or a node getting compromised. The outcome of this exercise indicates the most sensitive set of parameters for a given evidence. In case of our BAG, it can highlight the combination of exploits / threats that can have significant impact on the

Figure 5. Sensitivity tornado plot for Chipset_Compromised=true

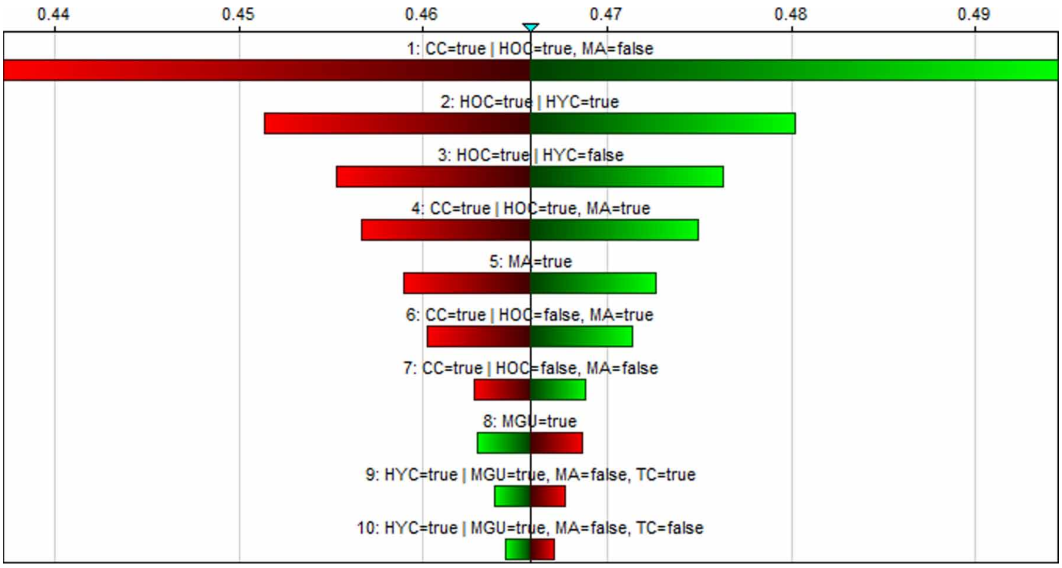
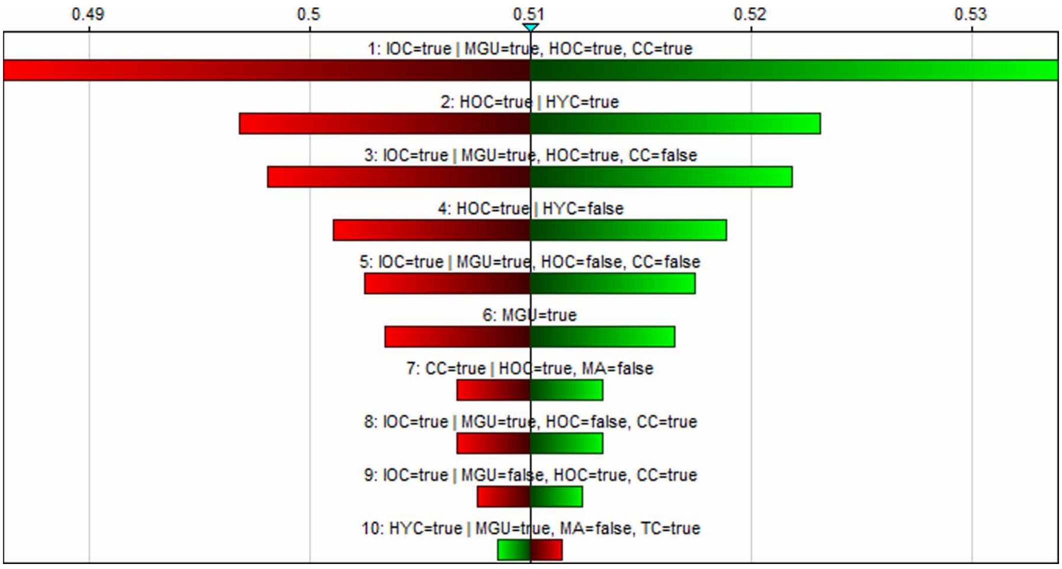


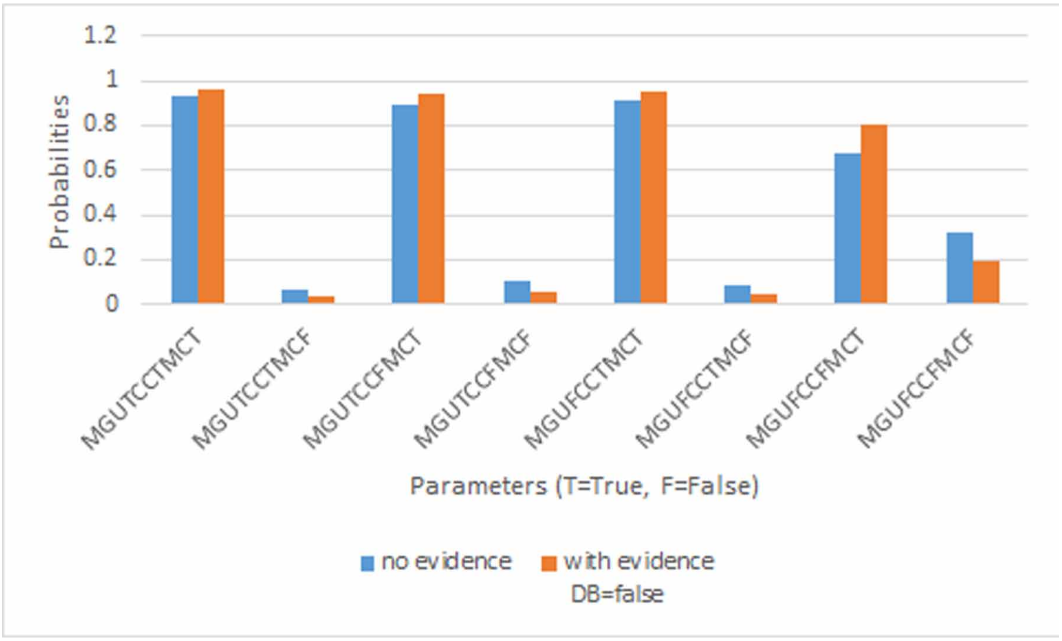
Figure 6. Sensitivity tornado plot for IO Compromised =true



current state of a node, in the light of a given evidence. Figure 7 shows an illustrative case for the node MC. The graph captures the parameter values at the node with no initial evidence. Further it captures the parameter values after setting the evidence as DB=false in the BAG. We can notice the way the parameter values change with the new evidence being set. In this case, all the parameters depict changes for this evidence. However the parameters $MGU_FCC_FMC_T$ and $MGU_FCC_FMC_F$ display a higher variation, indicating the sensitiveness of these parameters to the given evidence. In this manner, security practitioners can set the evidences from actual breach scenarios and try to analyze the sensitive parameters for these evidences, which can help them to choose the security defenses to be installed for avoiding similar violations.

To approximate changes in the target probability, one can use the sensitivity values as shown by Laskey (1995). Using this approximation, we have experimented by varying the probabilities in the model by a factor of 0.2, for the condition Data Breach (DB) = true. If after the variation, the resulting value is greater than 1, then the probability is reset by subtracting the value from 1. The resultant approximations were recorded and plotted to compare the quality of the approximation. Figure 8 shows the quantile-quantile plot of the actual probability changes and the resulting approximate changes. As discussed by Laskey (1995), the linear approximation is a good fit for the results. We have also

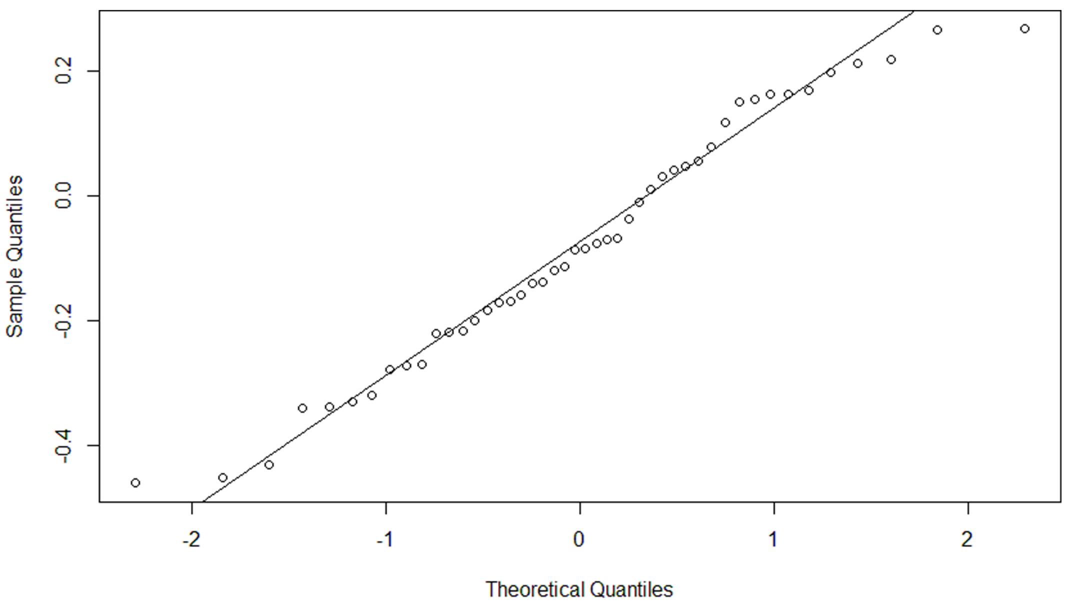
Figure 7. Changes in the probabilities of MC node, for different evidences



noticed that higher the probability difference, higher is the approximation error.

Sensitivity analysis can also be used to obtain the suggested modifications to parameters $\theta_{x_i|PA_{x_i}}$, which will result in satisfying the query constraint $p_{New}\left(z\ |e\right) \geq p$, where z is some event given the evidence e with p_{New} representing the new probability distribution after making the changes to $\theta_{x_i|PA_{x_i}}$. To illustrate this, we have carried out an experiment, on the node IOC in the presented model. The results of this experiment are depicted in Figure 9. The existing CPT values for this node will result in p(false) = 0.49, which indicates a nearly 50% chance of this node getting compromised. We can carry out SA to identify the changes in the parameters required for specific constraints. As an illustration query, if we are interested to assign a new value p_{New}(false) = 0.2. The required changes in the parameters to achieve this constraint, can be found out using SA. Figure 9 depicts the corresponding changes required for various CPT combinations for satisfying this constraint. With such an analysis, one can find out the set of parameters that can have a greater influence on a specific node, for a given constraint.

Figure 8. Q-Q plot of the Actual and approximate changes in probabilities for DB=true



To extend the discussion further, we have carried out experiments that can show the amount of changes required, if the desired probabilities at a node are within a specific range. Results from the analysis on the presented model for an example query ($p\left(IOC = true\right) \geq 0.8\, and\, p\left(IOC = true\right) \leq 0.2$) have been shown in Table 2The values depicted here indicate the changes required in the local conditional probabilities for various combinations of the truth values of the parent nodes of IOC to satisfy the constraint. With this analysis, one can infer the impact of the local conditional probability changes of the identified nodes on the selected target with query constraints.

Figure 10 depicts the comparison of different parameter values for multiple constraints for the node DB. Here the existing CPT values for the node result in $p\left(DB = false\right) = 0.2138$. This indicates a relatively low probability of the data breach not occurring. Security administrators will be interested to identify the requirements to avoid a specific security violation condition. In this case, if we assume that we are interested to assign a relatively high value $\left(DB = false\right) = 0.9$. We can carry out SA to reveal the necessary changes required in the parameter values for this condition. Figure 9 depicts these suggested parameter values. Similarly, to study the impact of these parameter changes on the overall node probability, we can re-run the SA with a different constraint $\left(DB = false\right) = 0.75\, and \left(DB = false\right) = 0.6$. The results from these analyses have also been captured in the graph. We can see that parameter values such as $ISC_T IOC_T MC_F$ and $ISC_T IOC_F MC_T$ are displaying high variation with the existing values for satisfying the given constraints. With this, one can comparatively analyze the way the parameter values impact, for different constraint values at a given node. This will be useful in selecting the most sensitive parameters for achieving a specific constraint at the given node and thus select the optimal defenses for securing the corresponding component in the infrastructure.

Figure 9. Changes in the probabilities of IOC node, as suggested by SA for the constraint pNew<=0.2

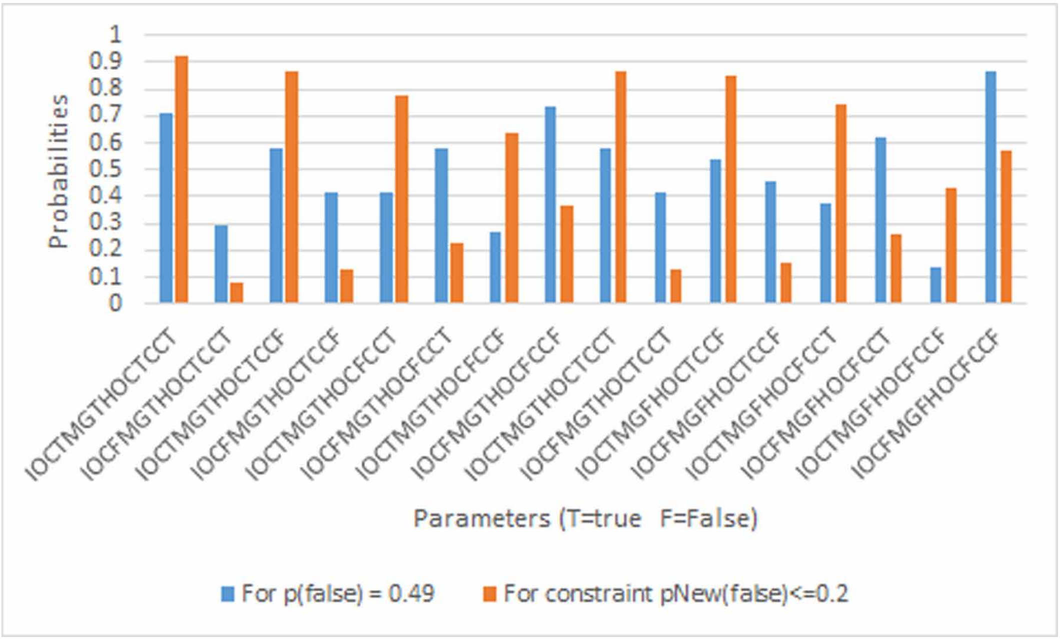
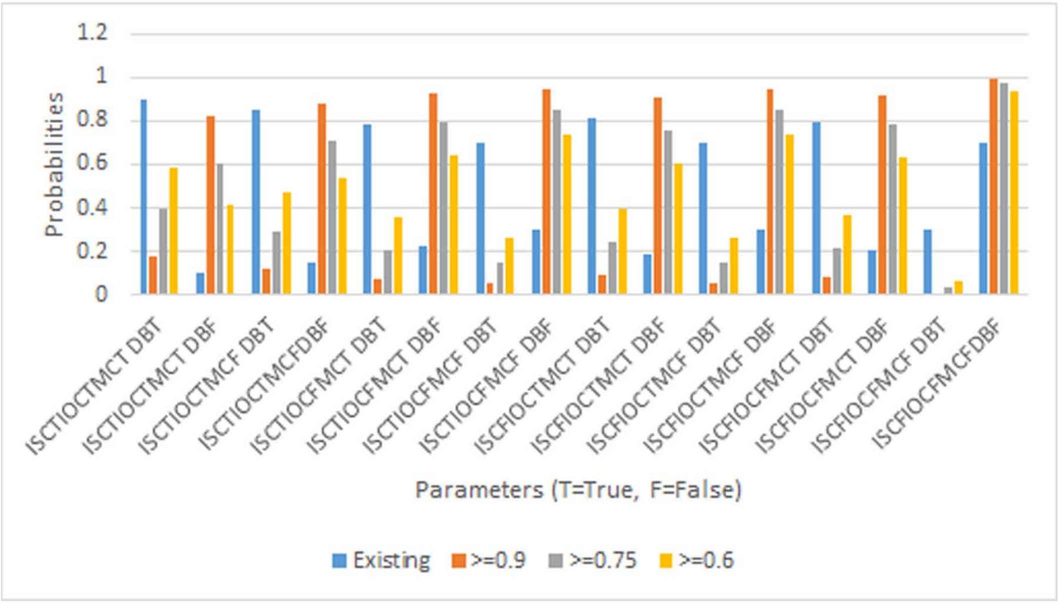


Table 2. Changes in θ values suggested by Sensitivity Analysis for the queries $p(\mathbf{IOC}) \geq 0.8\,\mathbf{and}\,p(\mathbf{IOC}) \leq 0.2$)

Parameter	Base Value	Lower Bound	Upper Bound
$IOC_T MG_T HOC_T CC_T$	0.708	0.328	0.920
$IOC_F MG_T HOC_T CC_T$	0.292	0.080	0.672
$IOC_T MG_T HOC_T CC_F$	0.583	0.219	0.869
$IOC_F MG_T HOC_T CC_F$	0.417	0.131	0.781
$IOC_T MG_T HOC_F CC_T$	0.419	0.127	0.774
$IOC_F MG_T HOC_F CC_T$	0.581	0.226	0.873
$IOC_T MG_T HOC_F CC_F$	0.267	0.068	0.633
$IOC_F MG_T HOC_F CC_F$	0.733	0.367	0.932
$IOC_T MG_F HOC_T CC_T$	0.583	0.219	0.869
$IOC_F MG_F HOC_T CC_T$	0.417	0.131	0.781
$IOC_T MG_F HOC_T CC_F$	0.541	0.191	0.848
$IOC_F MG_F HOC_T CC_F$	0.459	0.152	0.809
$IOC_T MG_F HOC_F CC_T$	0.378	0.109	0.742
$IOC_F MG_F HOC_F CC_T$	0.622	0.258	0.891
$IOC_T MG_F HOC_F CC_F$	0.137	0.031	0.430
$IOC_F MG_F HOC_F CC_F$	0.863	0.570	0.969

Figure 10. Comparison of changes in parameter values for the node DB, as suggested by SA for different constraints



CONCLUSION AND FUTURE DIRECTIONS

We have presented the approach for carrying out sensitivity analysis on Bayesian attack graphs for IaaS virtualized environments. We have presented the theoretical concepts behind sensitivity analysis and ways to derive sensitivity indices using single parameter value changes. The approach described here provides a practical model base that can be used by cloud administrators and security architects, as the local conditional probabilities have been derived after a careful analysis of the publicly reported vulnerabilities in the various layers of the IaaS virtualization stack. Sensitivity analysis approach presented here will be beneficial for selecting the optimal set of security countermeasures to be implemented by categorizing them based on the severity of their impact. It can also be beneficial in doing cost analysis of the various security subsystems to be implemented in the IaaS infrastructures.

Further research in this direction can include the development of a dynamic framework, which can incorporate automated querying of the newly reported threats in the public databases and re-model using the auto computed sensitivity values. Evidences from actual security breaches on cloud infrastructures and the specific vulnerabilities exploited to compromise them can be fed into the framework to obtain further realistic inferences from the model. The model can also consider incorporating approaches to perform SA using multiple parameter value changes. Research efforts are required to devise efficient techniques to perform these operations, as computing the partial derivatives in this case can become computationally intensive. To make the techniques computationally efficient, they have to be refined to find relevant parameters to check, instead of calculating for all the combinations of the parameter values in the CPTs, as presented in the current approach.

REFERENCES

Asvija, B., Eswari, R., & Bjioy, M. B. (2019). Security in Hardware Assisted Virtualization for Cloud Computing-State of the Art Issues and Challenges. *Computer Networks*, *151*(March), 68–92. doi:10.1016/j.comnet.2019.01.013

BayesFusion. L. L. C. (2017). *GeNIe modeler*. Retrieved from https://www.bayesfusion.com/genie/

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bouckaert, R. R. (1995). *Bayesian belief networks: from construction to inference* (PhD diss.). University Utrecht.

Brengel, M., Michael, B., & Christian, R. (2016). *Detecting Hardware-Assisted Virtualization*. DIMVA. doi:10.1007/978-3-319-40667-1_11

Chan, H., & Darwiche, A. (2004). Sensitivity analysis in Bayesian networks: From single to multiple parameters. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence* (pp. 67-75). AUAI Press.

Chan, H., & Darwiche, A. (2005). A distance measure for bounding probabilistic belief change. *International Journal of Approximate Reasoning*, *38*(2), 149–174. doi:10.1016/j.ijar.2004.07.001

Chan, H., & Darwiche, A. (2006). On the robustness of most probable explanations. *22nd Conference on Uncertainty in Artificial Intelligence*, 63-71.

Cheng, J., Greiner, R., Kelly, J., Bell, D., & Liu, W. (2002). Learning Bayesian networks from data: An information-theory based approach. *Artificial Intelligence*, *137*(1-2), 43–90. doi:10.1016/S0004-3702(02)00191-1

Colp, P., Nanavati, M., Zhu, J., Aiello, W., Coker, G., Deegan, T., & Warfield, A. et al. (2011). Breaking up is hard to do: security and functionality in a commodity hypervisor. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles* (pp. 189-202). ACM. doi:10.1145/2043556.2043575

Computer Security Resource Center. (n.d.). *National Vulnerability Database - NVD*. National Institute of Standards and Technology. Retrieved July 7, 2017, from https://nvd.nist.gov/

Cooper, G. F. (1990). The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, *42*(2-3), 393–405. doi:10.1016/0004-3702(90)90060-D

Darwiche, A., Casico, K., Allen, D., Chan, H., Chavira, M. P., Zaloznyy, D., & Zaloznyy, M. (2017). *Samiam: Sensitivity analysis, modeling, inference and more.* Retrieved from http://reasoning.cs.ucla.edu/samiam/

Desnos, A., Filiol, É., & Lefou, I. (2011, Feb.). Detecting (and creating !) a HVM rootkit (aka BluePill-like). *Journal in Computer Virology, 7*(1), 23-49.

Dewri, R., Poolsappasit, N., Ray, I., & Whitley, D. (2007). Optimal security hardening using multi-objective optimization on attack tree models of networks. In *Proceedings of the 14th ACM conference on Computer and communications security* (pp. 204-213). ACM. doi:10.1145/1315245.1315272

Fannon, R. (2014). *An analysis of hardware-assisted virtual machine based rootkits* (Diss.). Naval Postgraduate School, Monterey, CA.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997, November). Bayesian network classifiers. *Machine Learning*, *29*(2-3), 131–163. doi:10.1023/A:1007465528199

Frigault, M., Wang, L., Singhal, A., & Jajodia, S. (2008). Measuring network security using dynamic bayesian network. In *4th ACM workshop on Quality of protection* (pp. 23-30). ACM. doi:10.1145/1456362.1456368

Gauvain, J. L., & Lee, C. H. (1994). Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions on Speech and Audio Processing*, *2*(2), 291–298. doi:10.1109/89.279278

Geffner, J. (2015). *VENOM - Virtualized Environment Neglected Operations Manipulation*. Retrieved August 2016, from http://venom.crowdstrike.com/

Gill, S. S., & Buyya, R. (2018). SECURE: Self-protection approach in cloud resource management. IEEE Cloud Computing. *IEEE Cloud Computing*, *5*(1), 60–72. doi:10.1109/MCC.2018.011791715

Gorobets, M., Bazhaniuk, O., Matrosov, A., Furtak, A., & Bulygin, Y. (2015). Attacking Hypervisors via Firmware and Hardware. *Black Hat USA*.

Heckerman, D., Geiger, D., & Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, *20*(3), 197–243. doi:10.1007/BF00994016

Idika, N., & Bhargava, B. (2012). Extending attack graph-based security metrics and aggregating their application. *IEEE Transactions on Dependable and Secure Computing*, *9*(1), 75–85. doi:10.1109/TDSC.2010.61

Inci, M., Gulmezoglu, B., Irazoqui, G., Eisenbarth, T., & Sunar, B. (2016). Cache attacks enable bulk key recovery on the cloud. *ICCHE Conference, Springer Berlin Heidelberg,* 368-388. doi:10.1007/978-3-662-53140-2_18

Intel. (2016). *Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3C: System Programming Guide, Part 3.* Intel Corporation.

Intel. (2017). Intel® 64 and IA-32 Architectures Software Developer's Manual Combined Volumes. *Instruction Set Reference A-Z, 2A*(3), 190-206.

Joseph, L., & Mukesh, R. (2019). To Detect Malware attacks for an Autonomic Self-Heal Approach of Virtual Machines in Cloud Computing. *Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM),* 1, 220-231. doi:10.1109/ICONSTEM.2019.8918909

Kallenberg, C., Cornwell, S., Kovah, X., & Butterworth, J. (2014). *Setup For Failure: Defeating Secure Boot*. Hack In The Box - HITB.

Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models - Principles and Techniques*. MIT Press.

Kovah, X., Butterworth, J., Kallenberg, C., & Cornwell, S. (2013). Copernicus 2: SENTER the dragon. CanSecWest.

Laskey, K. B. (1995). Sensitivity analysis for probability assessments in Bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics*, *25*(6), 901–909. doi:10.1109/21.384252

Li, W., Lin, K., Zhao, T., Lan, T., Chen, X., Du, H., & Chen, H. (2019). Risk assessment and sensitivity analysis of flash floods in ungauged basins using coupled hydrologic and hydrodynamic models. *Journal of Hydrology (Amsterdam)*, *572*, 108–120. doi:10.1016/j.jhydrol.2019.03.002

Liu, Y., & Man, H. (2005). Network Vulnerability Assessment Using Bayesian Networks. *Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security*, *5812*(Mar), 61–71. doi:10.1117/12.604240

Loucks, D. P., & Beek, E. (2005). Model Sensitivity and Uncertainty Analysis. In *Water Resources SystemsPlanning and Management* (pp. 255–287). UNESCO.

Marchioni, A., & Magni, C. A. (2018). Investment decisions and sensitivity analysis: NPV-consistency of rates of return. *European Journal of Operational Research*, *268*(1), 361–372. doi:10.1016/j.ejor.2018.01.007

Mell, P., Scarfone, K., & Romanosky, S. (2007, June). *A Complete Guide to the Common Vulnerability Scoring System. CVSS Specification Document.* NIST and CMU. Retrieved from https://www.first.org/cvss/cvss-v2-guide.pdf

Modi, C. N., & Acha, K. (2017). Virtualization layer security challenges and intrusion detection/prevention systems in cloud computing: A comprehensive review. *The Journal of Supercomputing*, *73*(3), 1192–1234. doi:10.1007/s11227-016-1805-9

Munoz-González, L., Sgandurra, D., Barrere, M., & Lupu, E. C. (2017). Exact inference techniques for the analysis of Bayesian attack graphs. *IEEE Transactions on Dependable and Secure Computing*, 1–1. doi:10.1109/TDSC.2016.2627033

Munoz-Gonzalez, L., Sgandurra, D., Paudice, A., & Lupu, E. C. (2017). Efficient Attack Graph Analysis through Approximate. *ACM Transactions on Privacy and Security*, *20*(3), 10. doi:10.1145/3105760

Neapolitan, R. E. (2004). *Learning bayesian networks* (Vol. 38). Pearson Prentice Hall.

NIST. (2018, March 26). *National Vulnerability Database - CVEs for KVM*. Information Technology Laboratory. Retrieved from https://nvd.nist.gov/vuln/search/results?adv_search=false&form_type=basic&results_type=overview&search_type=all&query=KVM

NIST. (2018, March 26). *National Vulnerability Database - CVEs in Xen*. Information Technology Laboratory. Retrieved from https://nvd.nist.gov/vuln/search/results?adv_search=false&form_type=basic&results_type=overview&search_type=all&query=xen

Noel, S., & Jajodia, S. (2014). Metrics suite for network attack graph analytics. In *Proceedings of the 9th Annual Cyber and Information Security Research Conference* (pp. 5-8). Oak Ridge, TN: ACM. doi:10.1145/2602087.2602117

Noel, S., Jajodia, S., & Singhal, A. (2007). *Security risk analysis with attack graph metrics. Technical Report, CSIS, George Mason University*. George Mason University.

Oniśko, A., Druzdzel, M. J., & Wasyluk, H. (2001). Learning Bayesian network parameters from small data sets: Application of Noisy-OR gates. *International Journal of Approximate Reasoning*, *27*(2), 165–182. doi:10.1016/S0888-613X(01)00039-1

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems - Networks of Plausible Inference*. Morgan Kaufmann., doi:10.1016/C2009-0-27609-4

Pék, G., Buttyán, L., & Bencsáth, B. (2013). A survey of security issues in hardware virtualization. *ACM Computing Surveys*, *45*(3), 40. doi:10.1145/2480741.2480757

Pék, G., Lanzi, A., Srivastava, A., & Balzarotti, D. (2014). On the feasibility of software attacks on commodity virtual machine monitors via direct device assignment. *Proceedings of the 9th ACM symposium on Information, computer and communications security,* 305-316. doi:10.1145/2590296.2590299

Pollino, C. A., Woodberry, O., Nicholson, A., Korb, K., & Hart, B. T. (2007). Parameterisation and evaluation of a Bayesian network for use in an ecological risk assessment. *Environmental Modelling & Software*, *22*(8), 1140–1152. doi:10.1016/j.envsoft.2006.03.006

Poolsappasit, N., Dewri, R., & Ray, I. (2012). Dynamic security risk management using bayesian attack graphs. *IEEE Transactions on Dependable and Secure Computing*, *9*(1), 61–74. doi:10.1109/TDSC.2011.34

Razavi, S., & Gupta, H. V. (2019). A multi-method Generalized Global Sensitivity Matrix approach to accounting for the dynamical nature of earth and environmental systems models. *Environmental Modelling & Software*, *114*, 1–11. doi:10.1016/j.envsoft.2018.12.002

Richter, A., Herber, C., Rauchfuss, H., & Wild, T. (2014). Performance isolation exposure in virtualized platforms with pci passthrough i/o sharing. *International Conference on Architecture of Computing Systems,* 171-182.

Rutkowska, J. (2004). Red pill. Or How to Detect VMM Using (almost) One CPU Instruction. *Internet Archive.* Retrieved from http://web. archive. org/web/20110726182809/http://invisiblethings. org/papers/redpill.html

Salim, F., Ioannidis, M., Penlidis, A., & Górecki, T. (2019). Modelling permeation passive sampling: Intra-particle resistance to mass transfer and comprehensive sensitivity analysis. *Environmental Science. Processes & Impacts*, *21*(3), 469–484. doi:10.1039/C8EM00565F PMID:30724928

Salmerón, A., Rumí, R., Langseth, H., Nielsen, T. D., & Madsen, A. L. (2018). A review of inference algorithms for hybrid Bayesian networks. *Journal of Artificial Intelligence Research*, *62*, 799–828. doi:10.1613/jair.1.11228

Scarfone, K., Souppaya, M., & Hoffman, P. (2011). *Guide to Security for Full Virtualization Technologies - Recommendations of the National Institute of Standards and Technology*. NIST.

Sharkey, J. (2016). Breaking Hardware - Enforced Security with Hypervisors. Blackhat USA.

Shenoy, P. F., & Shafer, G. (1990). Axioms for probability and belief-function propagation. *Uncertainty in Artificial Intelligence*, *4*(9), 169–198. doi:10.1016/B978-0-444-88650-7.50019-6

Smith, J. E., & Nair, R. (2005). The architecture of virtual machines. *IEEE Computer*, *38*(5), 32–38. doi:10.1109/MC.2005.173

Swiler, L. P., & Phillips, C. (1998). *A graph-based system for network-vulnerability analysis.* Sandia National Labs No. SAND-98-1127C; CONF-980914. 10.2172/573291

Tsai, H. Y., Siebenhaar, M., Miede, A., Huang, Y., & Steinmetz, R. (2011). Threat as a service?: Virtualization's impact on cloud security. *IT Professional*, *14*(1), 32–37. doi:10.1109/MITP.2011.117

Wang, L., Islam, T., Long, T., Singhal, A., & Jajodia, S. (2008). An attack graph-based probabilistic security metric. In *IFIP Annual Conference on Data and Applications Security and Privacy* (pp. 283-296). Springer.

Wang, L., Singhal, A., & Jajodia, S. (2007). Toward measuring network security using attack graphs. In *Proceedings of the 2007 ACM workshop on Quality of protection* (pp. 49-54). ACM. doi:10.1145/1314257.1314273

Xie, P., Li, J. H., Ou, X., Liu, P., & Levy, R. (2010). *Using Bayesian networks for cyber security analysis. In Dependable Systems and Networks (DSN), 2010 IEEE/IFIP international conference on*. IEEE. doi:10.1109/DSN.2010.5544924

Xie, X., Schenkendorf, R., & Krewer, U. (2019). Efficient sensitivity analysis and interpretation of parameter correlations in chemical engineering. *Reliability Engineering & System Safety*, *187*(July), 159–173. doi:10.1016/j.ress.2018.06.010

Yarom, Y., & Falkner, K. (2014). FLUSH+ RELOAD: A High Resolution, Low Noise, L3 Cache Side-Channel Attack. *USENIX Security*, *2014*(August), 719–732.

Yuan, C., & Lu, T. C. (2007). Finding explanations in Bayesian networks. *18th International Workshop on Principles of Diagnosis*, 414-419.

APPENDIX

Identified Vulnerabilities for Deriving the CPT Values

Table 3. Critical and high vulnerabilities reported in commodity hypervisors during Jan 2016- Mar 2020 (Asvija, Eswari, & Bjioy, 2019)

CVE ID	Hypervisor	Score	Reason	Possible attack type	CWE categories [3]	AV[1]	AC[2]
CVE-2017-1000407	KVM	7.4	Allowing direct access to host I/O port 0x80	DoS, Host OS crash	ICUEC	A	Low
CVE-2017-12188	KVM	7.8	Improper traversal of guest page table entries	DoS, Host OS crash	PT	L	High
CVE-2017-12154	KVM	7.1	Improper access control on host CR8 register	Obtain sensitive information	IAC	L	Low
CVE-2017-2583	KVM	8.4	Improper Instruction emulation	Guest OS crash, Gain guest OS privileges	PPAC	L	Low
CVE-2016-10150	KVM	9.8	Vulnerability in ioctl_create_ device function	Host OS crash, Gain privileges	PPAC, UAF	N	Low
CVE-2017-2584	KVM	7.1	Instruction emulation for fxrstor, fxsave, sgdt, and sidt.	DoS, Obtain sensitive info from kernel memory	ILD, UAF	L	Low
CVE-2016-9777	KVM	7.8	Faulty bounds check for the VCPU index in I/O APIC mode	DoS, Gain host OS privileges	OOB - Read	L	High
CVE-2016-4440	KVM	7.8	Mishandling of the APICv on/ off state	DoS, Execute arbitrary code on the host OS	PPAC	L	Low
CVE-2016-3713	KVM	7.1	Improper handling of MTRR support	DoS, Obtain sensitive information	IAC	L	Low
CVE-2016-0749	KVM	9.8	Memory allocation flaw in SPICE remote display protocol	Execute arbitrary code, Buffer Overflow	BE	N	Low
CVE-2019-14821	KVM	8.8	Bug in implementation of Coalesced MMIO write operation	DoS, Host OS crash, Gain privileges	OOB-Write	L	Low
CVE-2019-7221	KVM	7.8	Use after free bugs	Obtain sensitive information	BE	L	Low
CVE-2018-16882	KVM	8.8	Use after free bugs	Obtain sensitive information	BE	L	Low
CVE-2018-7541	Xen	8.8	Inconsistencies in grant table transitions	DoS, Gain privileges	PPAC	L	Low
CVE-2017-17564	Xen	7.8	Incorrect error handling for memory page reference counts	DoS, Gain privileges	EH	L	High
CVE-2017-17563	Xen	7.8	Incorrect overflow check for memory page reference counts	DoS, Gain privileges	BE	L	High
CVE-2017-17045	Xen	8.8	Incorrect handling of Populate on Demand, Physical-to-Machine errors	DoS, Obtain sensitive info from kernel memory, Gain privileges	PPAC	L	Low
CVE-2017-15597	Xen	9.1	Incorrect assumption in pin count / page reference race in grant table code	Host OS crash, Gain privileges, DoS	PPAC	N	Low
CVE-2017-15592	Xen	8.8	Mishandling of self linear shadow mappings	DoS, Gain privileges	PPAC	L	Low
CVE-2017-15590	Xen	8.8	Mishandling of MSI mapping	DoS, Gain privileges	PPAC	L	Low
CVE-2017-14316	Xen	8.8	Out of bound access to an internal array	Execute arbitrary code	OOB - Read	L	Low
CVE-2017-12136	Xen	7.8	Race condition in the grant table code	DoS, Gain privileges	RC	L	High
CVE-2017-12135	Xen	8.8	Problems in handling transitive agents	DoS, Gain privileges	PPAC	L	Low
CVE-2017-12134	Xen	8.8	Errors in handling block I/O	DoS, Obtain sensitive info from kernel memory, Gain privileges	PPAC	L	Low
CVE-2016-9637	Xen	7.5	Flaw in range check for ioport read/write operations	Gain host privilege	PPAC	L	High
CVE-2016-10013	Xen	7.8	Mishandling of SYSCALL single step during emulation	Gain guest OS privileges	PPAC	L	Low
CVE-2016-9381	Xen	7.5	Double fetch vulnerability – Race condition in QEMU	Gain host privilege	PPAC, RC	L	High
CVE-2016-9386	Xen	7.8	Improper treatment of x86 NULL segments during memory access	Gain guest OS privileges	PPAC	L	Low
CVE-2016-9382	Xen	7.8	Mishandling x86 task switches to VM86 mode	Guest OS crash , Gain host privilege	PPAC	L	Low

Table 4. Critical and high vulnerabilities reported in commodity hypervisors during Jan 2016- Mar 2020 (Asvija, Eswari, & Bjioy, 2019)

CVE ID	Hypervisor	Score	Reason	Possible attack type	CWE categories [3]	AV[1]	AC[2]
CVE-2016-7093	Xen	8.2	Mishandling of instruction pointer truncation during emulation	Gain host OS privileges	PPAC	L	Low
CVE-2019-19578	Xen	8.8	Exploiting linear pagetables implementation by PV guests	DoS, Gain privileges	IV	L	Low
CVE-2019-19577	Xen	7.2	Exploiting pagetable height updates on AMD hosts with IOMMU	DoS, Memory leaks	IV	L	Low
CVE-2019-18425	Xen	9.8	Exploiting descriptor table limits in 32 bit PV emulations	Gain Privileges	PPAC	N	Low
CVE-2017-4941	ESXi	7.5	Vulnerability in handling VNC sessions	Execute remote code	BE	N	High
CVE-2017-4933	ESXi	7.5	Vulnerability in handling VNC sessions	Execute remote code	BE	N	High
CVE-2017-4924	ESXi		Out-of-bounds write vulnerability in SVGA device	Execute arbitrary code	OOB - Write	L	Low
CVE-2016-5330	ESXi	7.8	DLL side loading vulnerability	Gain Privileges	USP	L	Low
CVE-2019-5544	ESXi	9.8	Heap overwrite issue	Gain Privileges	OOB - Write	N	Low
CVE-2019-5521	ESXi	9.6	Vulnerability in pixel shader	DoS, Gain Privileges	OOB – Read	N	Low
CVE-2017-8714	Hyper-V	7.8	Improper validation of input from authenticated user	Execute remote code	IV	L	High
CVE-2017-8664	Hyper-V	8.8	Improper validation of input from privileged user	Execute remote code	IV	L	Low
CVE-2017-0212	Hyper-V	7.6	Improper validation of vSMB packets	Gain privileges	PPAC	A	High
CVE-2017-0181, 0180, 0163, 0162	Hyper-V	7.6	Improper input validation in network switch	Execute remote code	IV	A	High
CVE-2017-0109	Hyper-V	7.6	Input Validation vulnerability in Hyper-V	Execute arbitrary code	IV	A	High
CVE-2017-0075	Hyper-V	7.6	Access control vulnerability	Execute arbitrary code	IAC	A	High
CVE-2017-0021	Hyper-V	9.0	Improper validation of vSMB packets	Execute arbitrary code	IAC	A	Low
CVE-2016-0090	Hyper-V	7.1	Information disclosure vulnerability	Obtain sensitive information	ILD	L	Low
CVE-2016-0088	Hyper-V	9.3	Remote Code Execution Vulnerability	Execute arbitrary code	IAC	L	Low
CVE-2018-8489	Hyper-V	8.4	Remote Code Execution Vulnerability	Execute arbitrary code	IV	N	Low
CVE-2019-0719	Hyper-V	9.1	Remote Code Execution Vulnerability	Execute arbitrary code	IV	N	Low
CVE-2019-0721	Hyper-V	9.1	Remote Code Execution Vulnerability	Execute arbitrary code	IV	N	Low

1. AV = Attack Vector L = Local N = Network A = Adjacent Network 2. AC = Attack Complexity

3. CWE Categories: PPAC = Permissions, Privileges, and Access Control, UAF = Use After Free, ILD = Information Leak/Disclosure, OOB = Out of Bounds, IAC = Improper Access Control, BE = Buffer Errors, RC = Race Conditions, USP = Untrusted Search Path, IV = Input Validation, EH = Error Handling, ICUEC = Improper Check for Unusual or Exceptional Conditions, , PT = Path Traversal

B. Asvija is serving as a Joint Director in Centre for Development of Advanced Computing (C-DAC), Bengaluru, India. His current research is focused on building defenses to security threats in the virtualization and hypervisor technologies. Prior to this, he has carried out research in grid computing and parallel debugging technologies.

R. Eswari is working as Assistant Professor in the Department of Computer Applications, National Institute of Technology, Tiruchirappalli, India. She obtained her Bachelor of Engineering in Computer Science and Engineering from Bharathidasan University, Tiruchirappalli and Master of Engineering in Computer and Communication from Anna University, Chennai. She started her research in optimization techniques and awarded Ph.D. in Scheduling algorithms for distributed systems by National Institute of Technology, Tiruchirappalli. She has over 10 years of teaching and research experience and published more than 25 research papers. Her research interests include Computational Intelligence, Optimization, Multi-objective optimization, Cloud Computing and Wireless Sensor Networks.

Bijoy M. B. holds a Master of Technology degree in Computer and Information Sciences and is currently serving as a Principal Technical Officer in Centre for Development of Advanced Computing (C-DAC), Bengaluru, India. He is involved in designing and developing cloud computing infrastructures and security tools. He has been instrumental in setting up and operating Megha, a scientific cloud service for Indian researchers and scientists. His research interests include Cloud Computing and Grid computing technologies.