Accepted Manuscript

Bayesian network based weighted APT attack paths modeling in cloud computing

Aaron Zimba, Hongsong Chen, Zhaoshun Wang

PII: S0167-739X(18)30876-8
DOI: https://doi.org/10.1016/j.future.2019.02.045
Reference: FUTURE 4798

To appear in: *Future Generation Computer Systems*

Received date : 12 April 2018
Revised date : 27 December 2018
Accepted date : 22 February 2019

Please cite this article as: A. Zimba, H. Chen and Z. Wang, Bayesian network based weighted APT attack paths modeling in cloud computing, *Future Generation Computer Systems* (2019), https://doi.org/10.1016/j.future.2019.02.045

Bayesian Network Based Weighted APT Attack Paths Modeling in Cloud Computing

Aaron Zimba [a], Hongsong Chen [a,*], Zhaoshun Wang [a]

[a] Department of Computer Science and Technology, University of Science and Technology Beijing, 100083, China

Abstract— Security vulnerabilities exhibited in cloud computing components and technologies not limited to hypervisors, virtual machines, and virtualization present a major security concern. The primary challenge has been to characterize interlinked attack paths generated by Advanced Persistent Thereat (APT) attackers upon exploitation of vulnerabilities exhibited in cloud components. We propose a Bayesian network based weighted attack paths modeling technique to model these attack paths. In our approach, we employ quantitative induction to express weighted attack paths. We chain marginal and conditional probabilities together to characterize multiple attack paths from the attack source to the target node. In so doing, we evaluate the likelihood of an APT occurring in a given path. Furthermore, we propose an optimized algorithm to find the shortest attack path from multiple sources based on key nodes and key edges. The algorithm not only finds the shortest path but also resolves any existing ties amongst paths of equal weights. We characterize the attack time expense of the APT attack by modeling the associated atomic attack events in a path as Poisson variables obeying the Erlang distribution. The attack time expense is classified into three different levels; *High*. *Medium* and *Low*. We use the WannaCry ransomware attack to evaluate our model.

Keywords—attack path, advanced persistent threats, cloud computing, Bayesian attack network, exploit, vulnerability

1 INTRODUCTION

Security presents a major concern echoed by many organizations migrating to cloud computing [1]. With the advent of e-governance different governments likewise are switching to cloud computing and this has inadvertently attracted Advanced Persistent Threat (APT) attackers who target big corporations and governments [2]. APT attackers possess high levels of technical skills and have extensive resources at their disposal and this has enabled them to effectuate sophisticated stealthy reconnaissance, surveillance and data exfiltration attacks with little traceability if any at all. This profile of attackers has come to exploit vulnerabilities exhibited in cloud computing components not limited to hypervisors, virtual machines, virtual routers etc, to reach the otherwise secured or unreachable resources. Virtualization, for example, which is the foundation of most cloud offerings [3], has a myriad of attack vectors targeting virtual machines whether at

rest in the cloud data centers or during migration on the network. Attacks on such a level of detail require highly skilled threat actors, hence APTs. Traversal of vulnerable cloud components during an attack generates virtual attack paths which depict dependencies shared amongst the exploited vulnerabilities. Attack paths have been widely studied [4 -7] in literature using different approaches. However, most of the studies apply to generic network environments with discrete network devices as opposed to virtualized cloud computing devices [8]. Bayesian networks have been employed to study attack paths but they suffer from attack cycles which typically occur in real-world scenarios due to the interleaving of reconnaissance and active APT attack stages.

In this paper, we propose a Bayesian network based weighted attack paths modeling technique for a cloud environment in which we employ quantitative induction for expression of weighted attack paths edges as opposed to the common approach of weighting nodes. We contrast the attacker's view of the vulnerabilities in the target system against the real system in terms of attack graphs. We reduce the attacker's view of the targeted system based on the limitations imposed by the attributes and characteristics of the attack profile in order to depict real world scenarios. Furthermore, we identify the key nodes and edges of each attack path in the resultant Directed Acyclic Graph (DAG) which are useful for mitigation strategies. We also propose an optimized shortest path algorithm for finding the shortest attack path from multiple sources based on key nodes and key edges. Unlike other shortest path algorithms which are single-source single-target based [10], our algorithm is multiple-source based entailing that it considers different attack sources for a given target and outputs the shortest attack path. The algorithm not only finds the shortest attack path but it's also capable of resolving ties amongst paths of equal weights. Thus, unlike greedy algorithms, our proposed algorithm does not treat two paths as identical if they only share same weight. And since APT attacks are characterized by varying durations of the whole attack process [11], we endeavor to characterize the time expense associated with each attack path. We evaluate our modeling approach with a WannaCry ransomware attack which is capable of exploiting cloud computing environments [9] in the presence of specific vulnerabilities.

The remainder of the paper is organized as: Section 2 presents the threat model and formulation of attack paths while Bayesian modeling and attacker's view are discussed in Section 3. Illustrative results and the use case are presented in Section 4 while the conclusion is drawn in Section 5.

2 THREAT MODEL AND ATTACK PATH FORMALIZATIONS

2.1 The Threat Model

We model cloud attacks based on the philosophy of conceptual attack units [12]. The overall attack process comprises discrete units serving as basic building blocks which when correctly implemented lead to the actualization of the attack. The *threat actor*, otherwise attacking agent, executes a series of coordinated *actions* to obtain a set of *assets* needed to reach the *goal(s)*.

***Threat actor*:** this is a subject entity of a given attack scenario whose actions are directed towards a specific object or goal. A threat actor can either be a human actor or APT malware capable of executing actions independently or remotely via a command and control (C&C). In our exposition, the actor is a highly skilled attacker with a considerable level of sophistication employing highly resilient APT malware.

***Actions*:** These are sequential steps of a particular order which return a value of "true" when an asset is acquired and "false" if not. The threat actor has a considerable level of confidence in the returned value of the action.

***Assets*:** This is any resource that the actor acquires for the realization of the attack. If no further attainment of assets is needed after an action returns a "true" value, then the asset is the final goal itself. Otherwise, it is a sub-goal employed as a pivot to the final goal.

Goals: These are actualized final assets when the returned action values in an execution stream are all "true". Thus a goal is a representation of a request to complete a given asset since each goal has associated assets. We distinguish two assets; pivot assets and critical assets. Pivot assets are not directly linked to the target as opposed to critical assets.

2.2 Attack Path Formalizations

A threat actor can gain access to the cloud from inside the cloud itself as an insider. They may have valid or compromised user credentials and seek to elevate privileges to launch an attack. Those that originate from outside the cloud likewise could reflect the same credential set. The last attack source for attack path initialization is an outsider actor who has no credentials to the cloud. His is to exploit weaknesses in the cloud infrastructure and infiltrate the network at the first found exploit or weakness. To make our paper more self-contained and save space, we reference a number of documented cloud attacks [13 - 37] and build a table of ingress attack paths as shown in Table 2, putting into consideration the attack classes on cloud layers. The abbreviations in Table 1 below refer to Table 2, the Bayesian network structure in Figure 1 and further. The end goal of the attack path is user data residing in the cloud data center or on a client's VM on the virtual layer. We tabulate the attack ingress table as follows: the Cloud Layer column specifies the layer at which the attack occurs while the Attack type entails which tenet of the CIA triad (Confidentiality, Integrity and Availability) is pursued as the end goal of the attack process. The Attack Name denotes the specific name of the attack as documented in literature [13 - 37] whilst the Key Node and Key Edge represent the mandatory node and edge in the respective class required to actualize the attack.

The possible attack paths incorporates nodes and edges other than the key nodes and edges thereby providing an alternative path through which the attack can be realized. Unlike routers which tend to only process header information of the packet being routed, the honest-but-curious server on the other hand processes the actual information and however encrypted the client data might be, it is more often than not decrypted before processing in the cloud thereby exposing the client's data to the need-to-know attack

TABLE 1: ABBREVIATIONS AND TERMS

	Cloud Layer Component	Abbreviation	Description
1	Hypervisor	HV	A virtual machine monitor for creating and running virtual machines
2	Virtual Machine	VM	An emulation of computing resources RAM, CPU and other devices
3	Local Storage	LS	A physical storage device e.g. SAN, NAS, RAID etc
4	Physical Machine	PM	A general purpose physical device connected to the cloud network
5	Physical Server	PS	A special purpose physical machine e.g. domain, DBMS, web server etc.
6	Virtual Device	SDN	Software-centric approach to managing networking of a system
7	Virtual Router	VR	A software replication of OSI layer 3 internetwork routing.
8	Virtual Server	VS	A special purpose virtual machine operating on the virtual layer
9	Virtual Switch	VSwt	A software replication of OSI layer 2 local network switching
10	BT	Botnet	A collection of interconnected compromised hosts on a network
11	CSP Enclave	CSP E	The service cloud provider's internal private network
12	CSP Public	CSP P	The cloud service provider's public network available to cloud users
13	CCS Internet	CSS I	The global Internet which cloud computing services interacts with
14	Trusted Third Party	TTP	A network of shared trust relationship with the CSP like federated cloud

Since it is very difficult for a cloud user to monitor this attack as the mostly likely perpetrator would be the provider, mitigation of this attacker tends to be costly, e.g. homomorphic encryption [38] which seeks to conceal information that the server is processing from itself.

3 BAYESIAN NETWORK AND ATTACKER'S BEHAVIOUR MODELING

We now employ the services of Bayesian network statistics to the aforementioned attack paths in Table 2 to model the target system state and the attacker's view.

TABLE 2. INGRESS ATTACK PATHS ABBREVIATIONS AND TERMS

Cloud Layer	Attack Type	Attack Name	Key Nodes	Key Edge	Possible Attack Paths
Cloud Layer I (Physical Layer)	Confidentiality	HV-jacking [13]	PM, HV, Host OS	VM – HV	PM – HV, VM – HV, PM – VM – HV
		Honest-but-curious server [14]	PS, LS, VM	PS – LS	PS – LS, PS – VM – LS
		Malicious Insider [15]	PM, HV, VM	PM - VM	PM – HV, PM – VM, PM – HV – VM, PM – VM – HV
	Integrity	CSP sys admin + physical access [16]	PM, HV, VM	PM - VM	PM – HV – VM, PM – VM
		Data Corruption [17]	Router, VM, PM, LS	PM – L.S	PM – Router – L.S, PM – VM – L.S
		Malicious Admin [16]	PM, Router, VM, HV	PM - VM	PM – HV – VM, PM – Router – VM
	Availability	Link aggregation DOS [18]	Router, VM, PM, VSwt	Router - VM	VM – Router, VM – Host
		DDOS flooding [19]	PS, Router, VM, BT	VM – PS	VM – PS, VM – Router – PS
		Software DOS [20]	PS, HV, VM	VM – HV	PS – HV, VM – HV
Cloud Layer II (Virtual Layer)	Confidentiality	VM Migration Attack [21]	VM, VR, HV	VM – VR	VM – VR, VM – HV – VR
		Side Channel Attack [22]	VM, Memory, PM	VM – Memory	VM – Memory, VM – VR – Memory
		Nested Virtualization [23]	HV, VSwt, VM, PM	VM - HV	VM – HV, PM - HV
	Integrity	VM image corruption [24]	VM, HV, LS	VM – L. S.	VM – L.S, VM – VR – L.S
		Hypervisor Rootkit [25]	HV, VM	VM - HV	VM – HV, VM – VR – HV, PM – HV
		VM escape [26]	VM, VMM, Host OS, HV	VM - HV	VM – HV – Host OS
	Availability	SDN Attack [27]	VM, VR, VSwt, VSvr	VM - VR	VM – VS – VR, VM - VR
		RFA Attack [28]	VM, VR, Server, VSvr	VM - PS	VM – Server, VM - VR
		VM hopping [29]	VM, VR, VS	VS - VM	VM – VS – VM, VM – VR – VM
Cloud Layer III (Application Layer)	Confidentiality	MITM attack [30]	VM, Client Device, Router	VM - Router	VM – Router, VM – Client Dev.
		Cookie Poisoning [31]	VM, LS, VR	VM – L.S.	VM – VR – L.S, VM – L.S
		Side Channel [32]	VM, App. Server, LS	VM – L.S.	VM – App. Server – L.S, VM – L.S.
	Integrity	MITC attack [33]	Client Device, VM, L.S	VM - Router	VM – Client Dev, VM – L.S. – Client Dev.
		Replay attack [34]	Client Device, VM, Router	VM - Router	VM – Router – Client Dev, VM – Router – L.S.
		SQL Injection [35]	Client Device, VM, L.S.	VM - VR	VM – VR – L.S, VM – Client Dev. – L.S.
	Availability	Link aggregation DOS [18]	Router, VM, PM	VM - VR	VM – VR, VM – PM – VR
		DDOS flooding [36]	VM, BT, VR	VM - BT	VM – VR – BT, VM - BT
		XML - HTTP DOS [37]	VM, VR, App. Server	VM – App. Server	VM – App. Server, VM – VR – App. Server

Such application is not uncommon in studying vulnerabilities and probabilistic measurement of enterprise information systems. Most attack modeling techniques consider attackers as omniscient threat actors with complete knowledge of a system, unlimited skill and resources as well as unlimited time. They are further frequently assumed to make only right decisions and no mistakes during an attack and to exploit only the best available way. However, the reality is different as attackers tend to make mistakes. Attackers have limitations and their view of vulnerabilities and exploitability of a system differs from the actual view of the system. In this regard, we model the cloud system's view and the attacker's view (belief about the system).

We apply the attack paths to cloud components spread across the three cloud layers to construct a Bayesian Network Structure (BNS). The resultant network structure is a multiply-connected graph shown in Figure 1 below.

We distinguish four sources of infiltration attacks namely: 1) CSP Enclave – the cloud provider's own private network; 2) Trusted Third Party – a network sharing trust relationship with the CSP like inter-cloud, federated cloud or cloud broker. The attacker from this network has limited access to the cloud network depending on the design pattern and trust agreement; 3) CCS Public – a group/domain of users with legitimate access to the cloud who would otherwise need privilege escalation to access other cloud components; 4) CCS Internet – attack sources in this domain denote entities without tenant accounts with the cloud, they denote the general user from the Internet. The goal of the attacker as constrained by the attack model is to breach any or all of the CIA tenets of client (tenant) data on the cloud residing in local storage e.g. SANs, NAS, RAID or residing in a data center accessible from the cloud network.

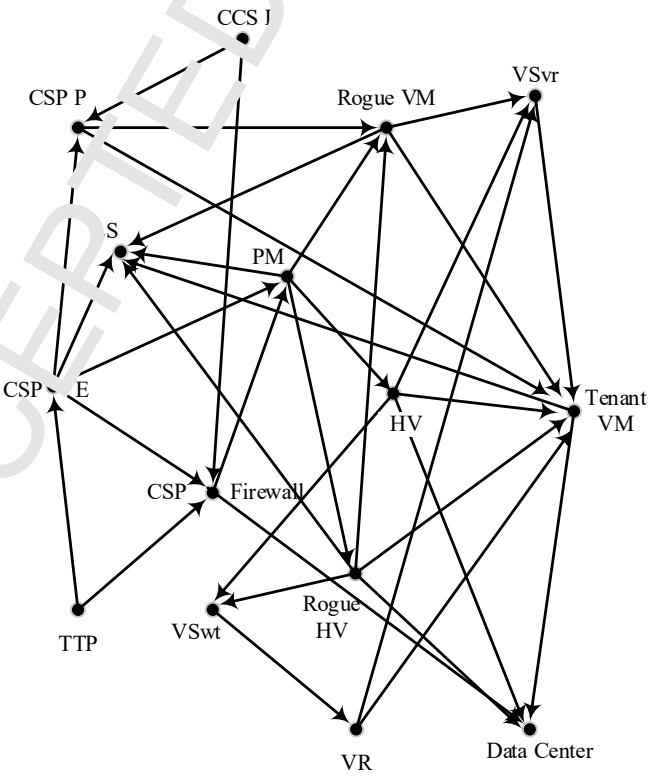

Fig. 1. Bayesian network structure of cloud components

3.1 Attacker's perception vs Actual system exploitability

The network structure above shows possible attack paths which an omniscient attacker can traverse to reach the goal provided the perceived vulnerabilities exist. At any given instance, the attacker resides at a node $\boldsymbol{N_i}$ and only transitions to the next node $\boldsymbol{N_j}$ if the vulnerability is present and exploitable. The probability of this *state* transition is given by:

$$\Pr(S_{ij}) = Pr^{V+}(S_{ij}) \cdot Pr_{exp}^{V+}(S_{ij}) \tag{1}$$

where $Pr^{V+}(S_{ij})$ is the probability presented by the vulnerability j in the cloud and $Pr_{exp}^{V+}(S_{ij})$ the conditional probability that such a vulnerability is exploitable if present. This implies that the attacker's view of the system only concides with that of the real system if he is considered omniscient. Nonetheless, in a real world scenario, the attacker's view does not concide with that of the real system. Rather, his view is determined by the set of vulnerabilities he perceives to be present in the system. This set of perceived vulnerabilities defines a new graph G_Γ which is a subset of the attack graph of the real system:

$$G_\Gamma = (V_\Gamma, A_\Gamma): V_\Gamma = V^+ \cup V^-, A_\Gamma = A^+ \cup A^- \tag{2}$$

where $V^+ \subseteq V$ and $A^+ \subseteq A$ are the subsets of vulnerabilities and attack steps that actually exist in the cloud sytem respectively and are believed so by the attacker. $V^- \subseteq V$ and $A^- \subseteq A$ are believed to exist by the attacker but actually do not exist in the system.

Since the threat actor of our model is an APT attacker, he dominates most of the parameters from the attack profile set $\xi = \{skill, resources, time, system\ knowledge\}$. However, the attacker's knowledge about the system in real scenarios tends to be limited even for an APT. This further reduces the attacker's view of the system to:

$$G_\xi^{att} = (V_\xi, A_\xi) = V_\xi \subseteq V_\Gamma, A_\xi \subseteq A_\Gamma \tag{3}$$

Since the knowledge about the system's vulnerability is a major determinant in an APT, the attacker increases this knowledge via discovery of the software running on the targeted cloud component and the associated vulnerability. Knowledge about the software alone is not enough as it is uncertain whether the software is patched or not. The probability of discovering the vulnerability is heavily dependent on the time elapsed since the vulnerability became known to the public. Assuming the probability of existence of this vulnerability from Equation (1) decreases linearly in the time elapsed, we have:

$$Pr^{V+}(S_{ij}) = \begin{cases} -\frac{1}{t_p} \cdot t + 1, & t_p \geq t \\ 0, & t_p < t \end{cases} \tag{4}$$

where t_p is time required to patch all vulnerable software, t is time elapsed from patch release and all software is patched for $t > t_p$. The conditional probability Pr_{exp}^{V+} of exploiting an existent vulnerability is computed using base scores based on expert knowledge as elaborated later in section 4.

3.2 The Bayesian Attack Network

Based on the attack network structure in Figure 1, we build the Bayesian Attack Network (BAN) from the attacker's view.

Definition: *a **Bayesian Attack Network** is an acyclic directed graph, with three arguements such that:*

$$BAN^{att} = (G_{\xi}^{att}, \Omega_i^{att}, ST^{att}), \text{ where}$$

- $G_{\xi}^{att} = (N_i, E_i)$ *is a graph of discrete random Bernoulli variables representative of existent exploitable vulnerabilities where the nodes* $N_i \in V_{\xi}$ *and the corresponding attack edges or arcs* $E_i \in A_{\xi}$.
- Ω_i^{att} *is a collective of quantitative network parameters* ω_i, *i.e.* $\omega_i \in \Omega_i^{att}$.
- ST^{att} *is an attack step denoting the feasibility of attack traversal from an ancestor node* N_i *to dependent node* N_j, *where* $(N_i, N_j) \in V_{\xi}$.

It is not uncommon for APT attacks to utilize multiple attack vectors by pursuing different exploits. The attack usually comprises a sequence of transitions over time traversing the cloud network from one node to the other depending on the attack structure. Moreover, some vulnerabilities, such as zero-day, appear with time meaning the BAN exhibits a property of dynamicity with respect to time implying the addition of new attack nodes to the BAN. In like manner, the cloud provider patches up some vulnerabilities upon detection which leads to elimination of the associated node from the BAN. This entails that attack nodes are prone to be added and deleted to the BAN with time. Therefore, in order to capture this dynamicity, the corresponding attack graph of the BAN can be expressed with respect to time as:

$$G_t^{att} = (N_t, E_t) \tag{5}$$

where $N_t = \{n_x \mid x = 1,2,3, \dots h_t\}$, is a set of nodes taking part in the attack up to the time t and $E_t = \{e_y \mid y = 1,2,3, \dots k_t\}$ is the set of the associated attack edges applicable to N_t. Considering the theory of network evolution, the addition and deletion of attack nodes at time $t+1$ can be expressed to satisfy the conditionality:

$$G_{t+1}^{att}(N,E) : \begin{cases} N_{t+1} = (N_t \cup \{n_+^{t+1}\}) - \{n_-^{t+1}\} \\ E_{t+1} = (E_t \cup \{e_+^{t+1}\}) - \{e_-^{t+1}\} \end{cases} \tag{6}$$

where $\{n_+^{t+1}\}$ denotes node addition with dynamic attack network growth and $\{n_-^{t+1}\}$ is the failure node

removed from the attack network upon failure of the preceding attack. Likewise, $\{e_{+}^{t+1}\}$ denotes appearance of an attack edge associated with the added attack node whereas $\{e_{-}^{t+1}\}$ denotes removal of the attack edge associated with the failure node.

Each node N_i of the BAN^{att} network casts a conditional probability distribution $\Pr\{N_i \mid parents(N_i)\}$ reflective of the quantified sample space restriction imposed by the parent nodes on child nodes. Since the nodes are discrete random variables (*cf.* definition), we henceforth employ conditional probability tables (CPT) to express the corresponding dependencies.

If we let the probability $Pr_{exp}^{V+}(S_{ij}) = Pr(N_i \mid N_j) = n_{ij}$ be the likelihood of exploiting a child node N_i given that parent node N_j has been accessed prior, via an attack action corresponding to the edge E_i, it follows from the acyclic property of the DAG $G_{\xi}^{att} = (N_i, E_i)$ that:

$$n_{ij}, n_{ji} = \begin{cases} Pr(N_i \mid N_j) \neq Pr(N_j \mid N_i) \\ n_{ij} = 0 \text{ or } n_{ji} = 0 \end{cases} \tag{7}$$

But even though the attacker is a highly skilled APT threat actor according to the threat model, who will avoid backtracking to maintain stealthiness and a long persistent undetected presence, it's only reasonable to assume that the attacker still might switch paths depending on the attack scenario. This creates multiple paths in the BAN.

3.3 Conditional Probabilities with detection

The probabilities discussed thus far do not factor in detection. As the attacker traverses the cloud network exploiting one node after the other, he generates noise in form of network traffic and system calls acting as seed for the IDS. Let the random variable denoting whether the IDS detects access to node N_i be denoted by D_i^+ and the absence of detection be denoted by D_i^- in the cloud network. We calculate the probability of detection as:

$$\Pr(D^+) = \frac{\Pr(N \mid D^-) - \Pr(N)}{\Pr(N)} \tag{8}$$

The above Equation (8) shows the effectiveness of the IDS in detecting infiltration attacks. If the attacker is detected in the network and his advances thwarted at this particular node, he seeks another path provided an exploitable vulnerability exists. Otherwise he is taken aback to the initial stages of the APT ttack chain. With detection in mind, we calculate the probability of undetected access as:

$$\Pr(N) = 1 - \prod_{i=1}^{n} [\,1 - \Pr(N \mid N_i) \cdot \Pr(N_i) \cdot [\,1 - \Pr(D_i^+)]] \tag{9}$$

Considering Equation (9), the probability of accessing the node N_i in the attack network via the parent nodes

N_j is thus given by:

$$\Pr(N_i) = 1 - \prod_{j=1, i \neq j}^{n} \left[1 - \Pr(N_i \mid N_j) \cdot \Pr(N_j)\right] \quad (10)$$

To find the initial conditional probabilities of accessing key nodes depicted in Table 2, we employ the use of CVSS and CPT as further elaborated in the next section.

And in light of detection from the IDS, we contend that the attacker does not backtrack to revisit an already accessed node N_j from current node N_i because not only does it not bring any new information nor add any value, but also that the APT actor will want to maintain a stealthy presence by not generating unnecessary traffic through backtracking to alarm the IDS.

4 Path Derivation and Illustrative Results

There are different paths of reaching target nodes, and these paths are linked by vulnerabilities exhibited in the nodes exploited by earlier discussed attacks [13 - 37]. Thus the relationships between the nodes are correlated by the corresponding vulnerabilities in the cloud components. However, existing derivation techniques assign individual numerical values to vulnerabilities on individual basis [11] [12]. By limiting our scope to the Base Score (BS), we employ CVSS V3.0 to assign scores to individual vulnerabilities and chain these vulnerabilities to capture the corresponding paths. The BS quantifies the intrinsic properties of a vulnerability immune to perturbation over time. We refer the reader to the official CVSS V3.0 for the details on the BS specifications. Expert knowledge in the knowledge domain is used for score assignment for exploits. Since we are handling conditional probabilities and the BS severity ranges from 0 – 10, we convert the BS into discrete node probability (P_{dn}) in the following way:

$$Pr_d(N_i \mid \forall\, c \in R_i) = {}^{BS_i}/_{10} \quad (11)$$

where R_i are characteristics features for actualization of the vulnerability exhibited by the node for conditions c from the base score parameters. Equation (11) assumes the condition that the vulnerability in the node N_i has satisfied all the requirements stipulated in the base score and thus the exploit holds true.

It's worth noting that a node can exhibit more than one vulnerability and this might lead to multiple edges between a node pair. These resultant arcs are handled by node fusion using applicable Equations (4) – (7).

A vulnerability v_j on node N_j is exploitable from node N_i, i.e. instantiation of exploit e_j, if the attack action transitions the attacker to a new state with new privileges. The two share a dependancy if they are reachable one from the other as stipulated by the parameters in the BS. To determine node reachability in the BN, we construct a node connectivity matrix representative of the adjacency matrix of the BN. To elaborate this and

derive CPs, we consider an illustrative attack graph depicted in Figure 4 and compute the corresponding connectivity matrix.

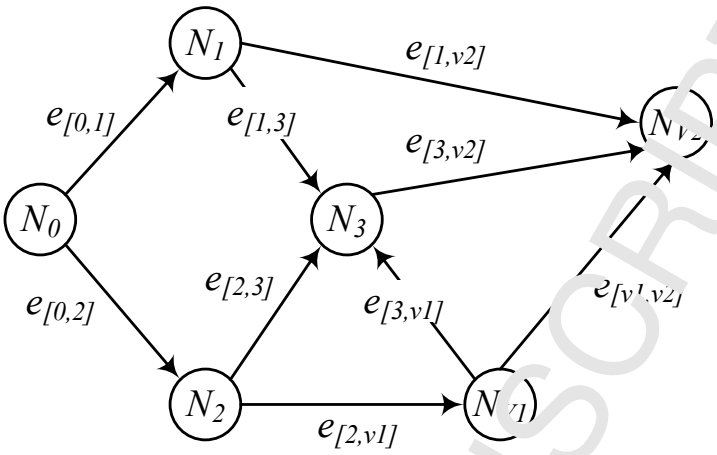

Fig. 4. Attack network with one target node.

The connectivity matrix *(CM)* is denoted by a |N| X |N| matrix $CM = (a_{ij})$ where the matrix element is a binary variable $a_{ij} = 1\ if\ (i,j) \in e_{[i,j]}$, otherwise $a_{ij} = 0$ denoting the absence of an edge between the node pair. For Figure 4, *CM* is a square matrix of the *6-th* order given by:

$$CM = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where each entry in the matrix is given as:

$$a_{ij} = \begin{cases} 1, if\ \exists\ edge\ from\ i\ to\ j \\ 0, \quad otherwise \end{cases}$$

It's apparent from the connectivity matrix that node N_3 has the highest vertex degree and does not self-loop. To find discrete node probability (P_{dn}) of each node in Figure 4, we use the NVD database for BS_i score derivation and further employ Equation (11) for probability conversion. From the aforesaid, we compute the Conditional Probability Tables (CPTs) indicative of the different conditions under which a respective node can be attacked. Table 3 below shows general CPT semantics of a child victim node N_v, conditioned on two incoming arcs from parents $N_i\ and\ N_j$ for disjunction and conjunction scenarios upon which we base all our CPTs computations henceforth. The probability $Pr_v(c = 1 \mid T)$ is the likelihood of exploiting the vulnerability v of node N_v via exploit e_i given that the conditions c stipulated in the base score have been satisfied. For vulnerabilities that do not require exploits, we assign mean discrete levels analogous to CVSS metrics in the

qualitative severity range :$Critical = 0.95$; $high = 0.795$; $medium = 0.545$; $low = 0.20$. Likewise for root nodes that do not have any parents as those depicted in the BAN in Figure 1, the four sources (root nodes) are assigned the following values on the subjective belief of the security administrator regarding infiltration from the attack sources: $CSP\ P = 0.90$; $CCS\ I = 0.65$; $TTP = 0.25$; $CSP\ E = 0.10$. We assign a higher weighting to CSP Public than CSS Internet because as elaborated from Table 2, most of cloud attacks reside on the virtual layer and are VM related thereby requiring the attacker to have had access to the cloud network, e.g. by having a legitimate account with the service provider or owning valid credential set for an authorized user.

TABLE 3 CPT FOR NODE N_v

Parents		*Union of Events*		*Intersection of Events*	
N_i	N_j	$Pr_v(c = 0 \mid F)$	$Pr_v(c = 1 \mid T)$	$Pr_v(c = 0 \mid F)$	$Pr_v(c = 1 \mid T)$
0	0	1	0	1	0
1	0	$1 - Pr(N_i)$	$Pr(N_i)$	1	0
0	1	$1 - Pr(N_j)$	$Pr(N_j)$	1	0
1	1	$1 - Pr(N_i \cup N_j)$	$Pr(N_i \cup N_j)$	$1 - Pr(N_i \cap N_j)$	$Pr(N_i \cap N_j)$

4.1 Conditional Probability Assignments

Using CVEs and CVSS BS scores from the NVD database, we apply Equation (11) to find the discrete node probability (P_{dn}) for the nodes in Figure 4 for vulnerabilities associated with selected cloud attacks from Table 2. This is illustrated in Table 4 below. The attacker, being an APT threat actor, uses a variety of reconnaissance techniques to surveil the network for such vulnerabilities. By inspecting his subnet mask, he is able to gather the number of hosts in his broadcast domain. The attacker can likewise infer adjacent networks by consideration of his subnet address and broadcast address. He further can have the topological overview of the associated subnets should he get hold of the corresponding routing tables. Armed with information collected from the aforementioned vectors, the attacker can use information gathering tools not limited to Armitage, Nessus, Nmap, Metasploit Framework etc to find exploits of vulnerabilities in the components residing in the surveilled subnets. To find the local conditional probability distributions (LCPD) of the nodes, we employ the use of CPTs of which the semantics are stipulated in Table 3. We first compute CPTs for nodes conditioned on a maximum of one incoming arc from the parent. The resultant CPTs for nodes N_0, N_1, N_2 *and* N_{v1} are shown in Figure 5 below.

TABLE 4. DISCRETE NODE PROBABILITY (P_{dn}) ASSIGNMENT

Node (N_i)	Attack Name	Cloud Layer	Associated CVE ID	(P_{dn})
N_0	CSP P Infiltration	Virtual	Subjective	0.90
N_1	SDN Attack	Virtual	CVE-2017-0181 [39]	0.76
N_2	VM Escape	Virtual	CVE-2017-0075 [40]	0.76
N_3	SQL Injection	Application	CVE-2016-8230 [41]	0.65
	XML DOS	Application	CVE-2017-8056 [42]	0.53
	Side Channel	Virtual	CVE-2017-5681 [43]	0.75
N_{v1}	VM Escape	Virtual	CVE-2016-3456 [44]	0.77
N_{v2}	Data Corruption	Application	CVE-2016-8931 [45]	0.88
	HV-jacking	Virtual	CVE-2017-3623 [46]	0.99
	Software DOS	Application	CVE-2015-1647 [47]	0.21

We now turn to compute the CPTs of the two nodes N_{v1} and N_{v2} whose conditional probability is each conditioned on three separate incoming arcs. Incoming arcs to a node can share a disjunction or conjunction relationship. The attacker in the case of former needs to pursue only one exploit unto completion to actualize the intended breach whereas the latter requires that all exploits be pursued unto completion to materialize the attack. This is so because a given vulnerability might only breach one tenet of CIA and the attacker will need to additionally actualize the other two exploits as well if he intends to breach the whole CIA tenet set.

N_0

c	$Pr_{N_0}(c = 0 \mid F)$	$Pr_{N_0}(c = 1 \mid T)$
1		
0	1	0

	$Pr_{N_1}(c = 0 \mid F)$	$Pr_{N_1}(c = 1 \mid T)$

	$Pr_{N_2}(c = 0 \mid F)$	$Pr_{N_2}(c = 1 \mid T)$
		0.76

	$Pr_{N_2}(c = 0 \mid F)$	$Pr_{N_2}(c = 1 \mid T)$

Fig. 5. CPTs assignments for nodes N_0, N_1, N_2 and N_{v1}

We assign a prior probability of 0.90 as an external attribute to N_0 corresponding to the CSP P infiltration source in the cloud network and as a subjective value relative to the security administrator's belief. The unconditional probability of the four nodes, shown below the corresponding CPTs, is calculated by consideration of the emanating corresponding sub-networks of the incoming arcs.

The case of a disjunction of arcs where at least one vulnerability needs to be exploited presents a probability of union of involved events and we calculate it as:

$$\Pr(N_i) = \Pr\left(N_j \cup N_k \cup N_l\right), where\ j,k,l \in \mathbb{R}$$

$$= \Pr(N_j) + \Pr(N_k) + \Pr(N_l) - \Pr(N_j) \cdot \Pr(N_k) - \Pr(N_j) \cdot \Pr(N_l) - \Pr(N_k) \cdot \Pr(N_l) + \Pr(N_j) \cdot \Pr(N_k) \cdot \Pr(N_l) \quad (12)$$

If all the vulnerabilities need to be exploited to achieve the attack, the corresponding conjunction probability of intersection of events is calculated as the product of the individual probabilities:

$$\Pr(N_i) = \Pr\left(N_j \cap N_k \cap N_l\right), where\ j,k,l \in \mathbb{R}$$

$$= \Pr(N_j) \cdot \Pr(N_k) \cdot \Pr(N_l) \quad (13)$$

We employ Equation (12) and (13) in computing CPT assignments for node N_3 and N_{v2} as shown in Table 5 and 6 respectively.

TABLE 5. CPT SPECIFICATIONS FOR TARGET NODE N_3

Nodes			Disjunction		Conjunction	
N_1	N_2	N_{v1}	$Pr(N_3 = F)$	$Pr(N_3 = T)$	$Pr(N_3 = F)$	$Pr(N_3 = T)$
0	0	0	1	0	1	0
1	0	0	0.24	0.760	1	0
0	1	0	0.24	0.760	1	0
0	0	1	0.23	0.77	1	0
1	0	1	0.055	0.945	0.415	0.585
1	1	0	0.058	0.942	0.422	0.578
0	1	1	0.055	0.945	0.415	0.585
1	1	1	0.027	0.973	0.555	0.445

The attacker at the root node can reach the victim node by traversing the edge $e_{[0,1]}$ or $e_{[0,2]}$. Even though these two arcs have the equal probabilities, it's worth noting that successful pursuance of the respective edge

does not result in the same security state of the victim nodes. CVE-2017-0181 associated with edge $e_{[0,1]}$ facilitates an SDN attack whereas CVE-2017-0075 associated with edge $e_{[0,2]}$ actualizes a VM escape attack. Though the two vulnerabilities affect instances of Hyper-V exploitable with an AV value of *Adjacent*, the former affects the network switch whereas the latter Remote Code Execution (RCE) on the component. The attacker therefore chooses the appropriate edge depending on whether he seeks privilege escalation via RCE or network traversal via the switch. It's worth noting that these two vulnerabilities might be exhibited by a single host and we would thus employ macro nodes as elaborated earlier, whether the case is disjunction or conjunction of events.

The attacker at node N_1 can either directly attack the victim node N_{v2} via edge $e_{[1,v2]}$ (since the Access Vector value is *Network*) by exploiting the vulnerability CVE-2016-8931 to breach confidentiality via directory traversal with a resultant probability of 0.667. Or if he chooses to breach integrity, he can first attack node N_3 via the edge $e_{[1,3]}$ by exploiting CVE-2016-8930 with a resultant probability of 0.494 and further seek to reach the target node N_{v2} via Hyperjacking through CVE-2017-3623 with a probability of 0.482. Since Hyperjacking gives the attacker root access, he is able to breach all CIA tenets by pursuing this route via the edge $e_{[3,v2]}$. Even though the attacker can breach node N_3 via edge $e_{[2,3]}$ with a probability of 0.401, it would be of less value to him since CVE-2017-8456 fosters an XML DOS attack on node N_3 which does not advance his prospects of reaching victim node N_{v2}. In order to exploit the VENOM vulnerability on node N_{v1} via the edge $e_{[2,v1]}$, he first needs to establish local subnet residency since the Access Vector value for CVE-2015-3456 is *Local Network Exploitable* with a resultant probability of 0.585.

TABLE6. CPT SPECIFICATIONS FOR VICTIM NODE N_{v2}

Nodes			Disjunction		Conjunction	
N_1	N_3	N_{v1}	$Pr(N_{v2}=F)$	$Pr(N_{v2}=T)$	$Pr(N_{v2}=F)$	$Pr(N_{v2}=T)$
0	0	0	1	0	1	0
1	0	0	0.350	0.880	1	0
0	1	0	0.001	0.990	1	0
0	0	1	0.790	0.210	1	0
1	0	1	0.095	0.905	0.815	0.185
1	1	0	0.002	0.998	0.129	0.871
1	1	1	0.008	0.992	0.792	0.208
1	1	1	0.001	0.999	0.817	0.183

Since utilization of the VENOM exploit facilitates VM escape the attacker can reach node N_3 via the edge $e_{[v1,3]}$ or opt to reach the targeted by exploiting CVE-2015-1647 with a resultant probability of 0.161 via the

edge $e_{[v1,v2]}$ to breach availability by initiating a DOS attack after having established local subnet co-residency. However, if he intends to breach integrity or confidentiality, he pursues the edge $e_{[v1,3]}$ exploiting CVE-2017-5681 with a resultant probability of 0.578 to reach node N_3.

Given a fused macro node $N_{mc} \rightarrow \{N_1, N_3, N_{v1}\}$, the probability of reaching node N_{v2} given N_{mc} in a disjunction scenario, i.e. the attack emanating from the constituents of the macro node, is equal to 99.9%. Decomposing the macro node to constitute only the first two nodes lowers the probability insignificantly to 99.8%. If the macro node were to contain the last two nodes, the probability lowers further to 99.2%. However, if the macro node constituted the first and last nodes in the macro node set, the probability further lowers to 90.5%. This implies that node N_3 exerts more influence with respect to the other nodes. On the other hand, the probability of reaching the target node N_{v2} in a conjunction case given N_{mc} is 18.3% which is ~ 20% that of the disjunction case. This is logically sound given the complexity of carrying out three independent events successfully to satisfying a single condition. It's worth noting that eliminating N_{v1} from the macro node set raises the probability significantly to 87.1%. The influence exerted by node N_3 is felt both in the disjunction and conjunction case. In light of the above, the security analyst could prioritize turning node N_3 of the macro node set into a failure node for security mitigation. This does not just lower the conditional probabilities but also eliminates a set of four edges $\{e_{[1,3]}, e_{[2,3]}, e_{[v1,3]}, e_{[3,v2]}\}$ as arguments of the dynamic function $G_{t+1}^{att}(N, E)$ at time $t+1$ from subsection 4.1. Turning N_3 into a failure node implies that the attacks to target node N_{v2} can only come from N_1 or N_{v1} and are limited to breach either confidentiality or availability.

4.2 The Optimized Shortest Path Algorithm and Edge Weighting

We now present the proposed algorithm for finding the shortest attack path in a given graph from the attacker's view. The input to the algorithm is a weight matrix whilst the output is the shortest path represented as a three-tuple comprising the effective distance, the cardinality of atomic attack steps in that path and the attack time expense. We define weight matrix $W = (e_{ij})$ as a mapping of the probability of a successfully exploiting a vulnerability $e_{i,j}$ present in the network to the corresponding element in the connectivity matrix *CM*:

$$W = \begin{bmatrix} e_{01} & e_{02} & . & . & . & e_{0n} \\ e_{11} & e_{12} & . & . & . & e_{1n} \\ e_{21} & e_{22} & . & . & . & e_{2n} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ e_{n1} & e_{n2} & . & . & . & e_{nn} \end{bmatrix}$$

Considering that we have 4 attack sources, the input matrices to the algorithm are thus W_0, W_1, W_2 and W_3. The corresponding attack sources for these matrices are w_0, w_1, w_2 and w_3 respectively. The output consists of a maximum from the products of attack paths (D_{eff}) with the minimum number of atomic attack steps ($St_{\sum P}$). Figure 6 shows the algorithm for generating the shortest path given the input matrices.

Algorithm: Shortest Attack Path Generation Algorithm

Input: Matrix Stream W_0, W_1, W_2, W_3
Output: Shortest path $\{D_{eff}, St_{\sum P}, \{TE\}_i\}$

1: Initialize and Read Matrix Stream
2: **for** $W_0 != Null$
3: **if** $\exists\, e_{0n}: e_{0n} \neq 0$ then
4: Shortest path $\rightarrow \{D_{eff} = e_{0n}, St_{\sum P} = 1\}$ from w_0
5: **else**
6: Extract the pivot element where $e_{ij} \neq 0$ && $i = 0$
7: Extract all non-zero elements in j-th row
8: **if** $\exists j: j = n$ then
9: Shortest path $\rightarrow \{D_{eff} = (e_{0j}) \cdot (e_{jn}),\ St_{\sum P} = 2\}$
10. Break this loop and go to next pivot element $e_{ij} \neq 0$ && $i++$
11: **else**
12: Extract all non-zero elements in k-th row where $k < n$
13: **if** $\exists k: k = n$ then
14: Shortest path $\rightarrow \{D_{eff} = (e_{0j}) \cdot (e_{jn}) \cdot (e_{kn}),\ St_{\sum P} = 3\}$
15. Break this loop and go to next pivot element $e_{ij} \neq 0$ && $i++$
16: **else**
17: Repeat extraction of non-zero elements until extracted $row = n$
18: **end if**
19: **end if**
20: **end if**
21: **end for**
22: Repeat step 2 for W_1, W_2 and W_3
23: Given w_0, w_1, w_2 and w_3;

$$Shortest\ Path\ (SP) = \begin{cases} D_{eff} = max \prod_{i,j=0}^{n} (e_{ij}) \\ St_{\sum P} = min \left| e_{ij} \rightarrow e_{nn} \right| \\ \{TE\}_i \end{cases}$$

Fig. 6. Algorithm for generation of Shortest Attack Path

The algorithm in figure 6 extracts different attack paths from each input matrix and computes the weightiest path with the least number of attack steps. For illustration purposes, we consider a single matrix as input to the algorithm in figure 6. The resultant weight matrix derived from mapping probabilities of exploitating the present vulnerabilities to the connectivity matrix of the attack graph (fig. 4) is expressed as:

$$W = \begin{bmatrix} 0 & 0.76 & 0.76 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.65 & 0 & 0.88 \\ 0 & 0 & 0 & 0.53 & 0.77 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.90 \\ 0 & 0 & 0 & 0.75 & 0 & 0.21 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We define the weight function by mapping discrete node probabilities to edges in the weight matrix $W = (e_{ij})$ for the range $\{0, 1\}$ in following way:

$$\xi: E(G_i) \rightarrow \mathbb{R}^+ \cup \{0\} \rightarrow 1, where\ G_i \subset G_\xi^{att} \tag{14}$$

Following from Equation (14), we express the weight of the edge between two nodes N_i and N_j as $\xi(e_{[i,j]})$. The cumulative edge weight of an edge stream (path) is given as the product:

$$\xi(P) = \prod_{e \subset P} \xi(e) \tag{15}$$

Further, we define the cardinality of the path consisting multiple edges between N_i and N_n as $|\xi(e_{[i,n]})|$ to denote the number of edges between those nodes. This corresponds to the number of atomic attack actions required of the attacker to reach the goal node. From $W = (e_{ij})$, we deduce five directed feasible paths from N_0 to N_{v2} listed as follows:

$$\begin{aligned} &(1)\ \ P_1: e_{[0,1]} \rightarrow e_{[1,v2]} \\ &(2)\ \ P_2: e_{[0,1]} \rightarrow e_{[1,3]} \rightarrow e_{[3,v2]} \\ &(3)\ \ P_3: e_{[0,2]} \rightarrow e_{[2,3]} \rightarrow e_{[3,v2]} \\ &(4)\ \ P_4: e_{[0,2]} \rightarrow e_{[2,v1]} \rightarrow e_{[v1,v2]} \\ &(5)\ \ P_5: e_{[0,2]} \rightarrow e_{[2,v1]} \rightarrow e_{[v1,3]} \rightarrow e_{[3,v2]} \end{aligned} \tag{16}$$

Therefore, attacker has different paths of reaching the target node N_{v2} from the source N_0 and his options increase after successfully implementing the first level exploit either via N_1 or N_2. In order to measure the overall likelihood of reaching the destination, we evaluate all possible paths available to the attacker from

source to destination. This is equivalent to the probability of reaching the target node after traversing and exploiting discrete nodes which otherwise isn't captured in the discrete node assignment.

Path (3) is an infeasible path in as far as reaching N_{v2} is concerned since the vulnerability CVE-2017-8056 exploited via the edge $e_{[2,3]}$ does not further efforts to reach the target but rather ensues a DOS attack on N_3. This entails that we have four paths to the destination. Following from Equation (14) and (15), we compute the shortest attack path as the maximum effective distance (D_{eff}) from attack source N_0 to the target N_{v2} with the least number of attack action steps:

$$\left[D_{eff}\right]_{G_{i,j}}^{\prod}\left(e_{[i,j]}\right) = \max\{\xi(P) \mid P: e_i \rightarrow^* e_j\}$$

$$St_{\sum P} = min\left|\xi(e_{[i,n]})\right| \tag{17}$$

where $St_{\sum P} \in \mathbb{N}$ (*cf.* definition) is the number of atomic attack steps in a given path P_i. Applying the conditions from (16) to the list of feasible paths (17), we have:

$$(1)P_1: \left[D_{eff}\right]_{G_{i,j}}^{\prod}\left(e_{[i,j]}\right) = 0.667, \quad St_{\sum P} = 2$$

$$(2)P_2: \left[D_{eff}\right]_{G_{i,j}}^{\prod}\left(e_{[i,j]}\right) = 0.489, \quad St_{\sum P} = 3$$

$$(4)P_4: \left[D_{eff}\right]_{G_{i,j}}^{\prod}\left(e_{[i,j]}\right) = 0.123, \quad St_{\sum P} = 3$$

$$(5)P_5: \left[D_{eff}\right]_{G_{i,j}}^{\prod}\left(e_{[i,j]}\right) = 0.435, \quad St_{\sum P} = 4$$

It's clear from the above that the path P_1 chosen is shortest attack path since it carries heaviest path weight with the least number of atomic attack steps. This implies that the attacker has a higher probability of reaching the target node while performing the least number of attack actions via this path. The tie in number of attack steps occurring between path P_2 and P_4 can be broken by considering their respective path weights meaning path P_2 is the better option. The path P_5 has the highest required number of attack actions implying the attacker will have to exploit more vulnerabilities in order to reach the target if he were to choose this path. It's also worth noting that path P_1 and P_4 lead to the breach of only one CIA tenet, i.e. confidentiality and availability respectively, entailing that the resultant security of the breached system after attacks pursued via these two separate paths is different. In light of this, path P_2 then should actually be compared against P_5 because both paths lead to the same breached security state of the target system by way of sharing the same end attack edge $e_{[3,v2]}$, i.e. a full breach of all the three CIA tenets. Therefore, path P_2 is a better path than P_5 by having a weightier path by 25% and attack step actions by 9.94%.

All the end arcs connected directly to the targeted victim are the key edges for the respective path under consideration. The key nodes are those nodes linked directly to the victim node via the key edge. The key

edges in our case, applicable to Table II, are $e_{[1,v2]}$, $e_{[3,v2]}$ and $e_{[v1,v2]}$ whilst the corresponding key nodes are N_1, N_3 and N_{v1} respectively.

Since the resultant breached security state via the key edge $e_{[1,v2]}$ is a subset of that via key edge $e_{[3,v2]}$, the security analyst might need to mitigate attacks pursued via the latter end attack edge. This means turning key node N_3 into a failure node and this implies that the system is categorically protected against integrity and availability attacks via path P_1 unlike if the failure node was created at N_1, the system's confidentiality, integrity and availability could be breached via path P_5 even though it has the worst path metrics, i.e. 60% more attack actions and a lesser path weight by 34.8%.

It's worth noting that when there's a tie in the cumulative edge weight $\xi(P)$ of two given paths, the tie is broken by selecting the path with the lowest $St_{\sum P}$. Similarly, when there's a tie in the number of atomic steps $St_{\sum P}$ for two given attack paths, e.g. in the case of P_2 and P_4, the tie is broken by selecting the path with the highest cumulative edge weight $\xi(P)$. In this manner, the proposed algorithm effectively finds the shortest path unlike in the Dijkstra algorithm where a tie is resolved arbitrarily.

4.3 Attack Time Expense

APT attacks exhibit a time characteristic where the actor seeks to maintain a long undetected presence from the time of infiltration and reconnaissance to the actual attack. Following from this, we characterize the attack time expense associated with the attack paths in Equation (16). The time expense of a given attack path is the cumulative time expenses of the associated atomic attacks whose discrete nature is given by an exponential distribution:

$$f(t{:}\lambda) = \lambda \cdot e^{-\lambda t} \tag{18}$$

where the rate of success of the attack is denoted by λ.

Since the cumulative edge weight $\xi(P)$ of an attack path is a conjunction of attack events (*cf.* Equation 16), the probabilities in that path are independent events hence the product. Furthermore, the probability value associated with an attack edge is derived from the base score which does not change over time (unlike temporal scores that change with time). Following from the above, the atomic attack event t from Equation (18) can be expressed as a Poisson variable with a distribution given as:

$$F(t{:}\lambda, k) = \sum_{n=0}^{n-1} \frac{(\lambda)^n \cdot e^{-\lambda}}{n!} \tag{19}$$

where $\lambda = \mu = \sigma^2 \in \{\mathbb{R}^+ \cup \{0\} \leq 1\}$ is the success rate of the attack (μ is the mean, σ^2 is the variance) and $k \in \mathbb{N}$ is the attack complexity dependent on $St_{\sum P}$. The sum of these distribution functions representative of the attack time expense of the corresponding attack path is a cumulative Erlang distribution

expressed as:

$$F(t{:}\lambda,k) = 1 - \sum_{n=0}^{k-1} \frac{(\lambda t)^n \cdot e^{-\lambda t}}{n!} \tag{20}$$

Since the number of events in an attack path is given by k, similarly $k = St_{\sum P}$. Thus, the mean of a given attack path is computed as:

$$\lambda = \frac{1}{St_{\sum P}} \cdot \prod_{e \subset P} \xi(e) \tag{21}$$

Given the values of k and λ from Equation (21), and that the cumulative attack time resulting from chaining of the atomic attacks obeys Erlang's distribution, we calculate, with respect to time t, the probability density function for different values of k and λ as:

$$f(t{:}\lambda,k) = \frac{(\lambda)^k}{(k-1)!} \cdot t^{k-1} \cdot e^{-\lambda t} \text{ for } t, \lambda \geq 0 \tag{22}$$

Using Equation (22) and the values of k and λ from Equation (21), we generate the probability density curves for the corresponding attack paths in Equation (16). The resultant density graph is illustrate below in Figure 7.

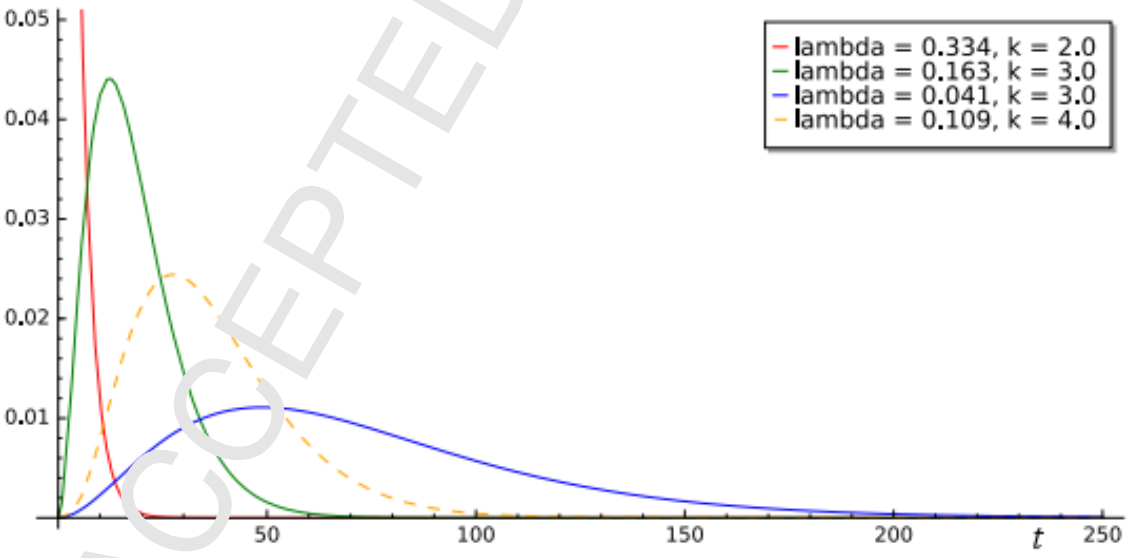

Fig. 7. Probability density functions for attack paths

The positive skew of $f(t{:}\lambda,k)$ is a right-skewed distribution where the mass thereof is concentrated on the left and an increment in k perturbates a decrease in the average variance of the distribution. Clearly, λ *and* k exert an influence over the mode and mean likewise. We acknowledge the fact that the overall time

expense of the attack would include the time spent on surveillance during reconnaissance of potential victims. Notwithstanding the aforementioned, we contend that the attacker will first initiate a full reconnaissance attack on the whole network before launching any exploits. Therefore, this initial timeframe of the APT attack is considered common to all attack paths pursued thereafter and we thus contend that it's an unpredictable parameter in the case of further repeated reconnaissance for discovery of the new nodes discussed in earlier.

Therefore, we use a parameter, Time Expense ($\{TE\}_i$), to characterize the time expense of the APT attack. We first divide the range of the $density\ curve$ values from all the attack paths into 3 ranges denoted by Δt_i for High, Medium and Low as shown below:

$$\Delta t_{High} = t_{max} - t_{mean}$$
$$\Delta t_{Medium} = t_{mean} - t_{min}$$
$$\Delta t_{Low} = t_{min} - t_0 \tag{23}$$

where $|\Delta t_{High}| = |\Delta t_{Medium}| = |\Delta t_{Low}|$ are equivalent ranges and t_0 is an attack step denoting direct access from the attack node to target without any pivoting nodes. We thus characterize Time Expense using Equation (23) from the above as:

$$\{TE\}_i = \begin{cases} High, & t_{mean} < t_i \leq t_{max} \\ Medium, & t_{min} < t_i \leq t_{mean} \\ Low, & t_0 \leq t_i \leq t_{min} \end{cases} \tag{24}$$

A $\{TE\}_i$ of High entails that an APT has had the highest persistence presence relative to that of Medium while a $\{TE\}_i$ of Low entails the least persistence presence. The effectiveness of the attacks in these parameters are deduced from λ in the corresponding attack path. Therefore, APT attacks observed at the target node which belong to the *High* category indicate a longer persistence presence whilst those belonging to the *Low* category insinuate the opposite. The time expense for attacks exploiting newly discovered vulnerabilities and zero-days is expected to be extremely short.

4.4 WannaCry ransomware attack use case

Many governments and big corporations, which are essential targets of APTs, have shifted to provision various service via cloud computing. This has consequently led to the emergency of APT attacks in cloud computing environments. Though APT attacks are conventionally known to seek to exfiltrate data, recent trends have seen a shift towards incapacitating the operations of the victim by making production data inaccessible. This has been achieved via the use of crypto ransomware. Although crypto ransomware is

traditionally known to seek a ransom before data accessibility is made available via decryption, the malware variants used in APTs do not seek to decrypt the data whatsoever. This was the case Petya ransomware attacks in June 2017 on Ukrainian government ministries, banks, electricity and other utility companies. It's worth noting that though government agencies and ministries tend to favor the use private or federated cloud computing as opposed to other APT target entities such as corporations that might implement hybrid or public cloud computing, the attack structure generally remains the same due to the fundamental cloud partitions (see Table 2). We consider WannaCry ransomware which uses the Eternal Blue exploit, the same exploit used by Petya ransomware to exploit vulnerabilities in the Windows operating system and subsequent cloud networks. It leverages multiple exploits in the Windows SMB network sharing resource. It's known that WannaCry begins encrypting victim files the moment it infects a host and does the same to any locally attached storage devices. The infection and subsequent attack process is summarized in figure 8 below.

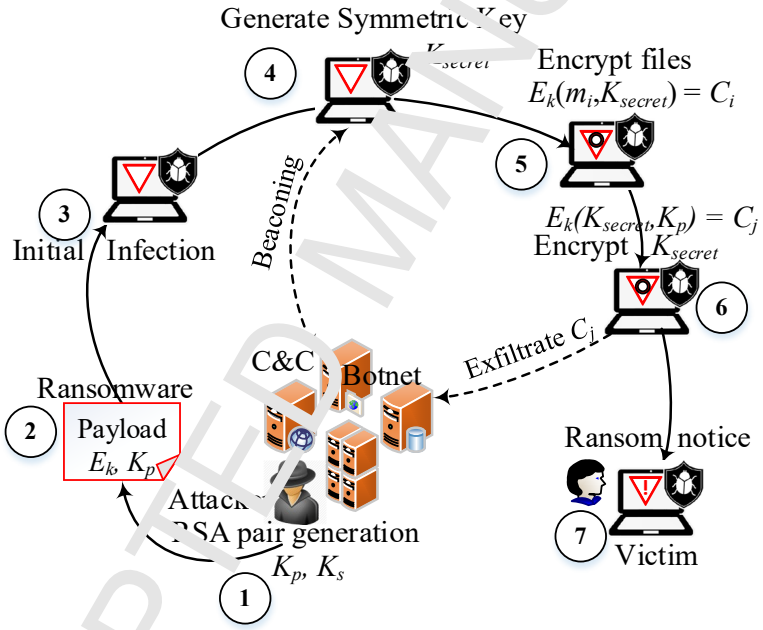

Fig. 8. Summarized WannaCry infection and attack process

Being a hybrid crypto ransomware, the malware comes with an embedded public key (RSA) K_p in its payload which is used to encrypt the symmetric key (AES) K_{secret} to produce the ciphertext C_j. The symmetric key is used to encrypt the user files which is the end-goal on this particular victim with an output ciphertext C_i. Access to encrypted user files is only possible upon decryption but with the master private key K_s from the RSA pair generated by the attacker on the C2 servers or botnet. This essentially is an attack on availability in the CIA security principles. Since the *Acces Vector* for the CVE-2017-0144 (exploited by WannaCry) is Network, the attack is feasible from within and outside the targeted cloud subnet. Therefore, we partion to attack scenarios generating two different attack paths:

(1) when the attacker resides within the internal subnet

(2) when the attacker resides in an external subnet and thus requires to reach the target network across OSI layer 3 bounderies before launching the attack.

We simulate these two attack scenario (1 and 2) on VMs in a virtualized sandbox environment as illustrated in figure 9 below using hypervisor type II.

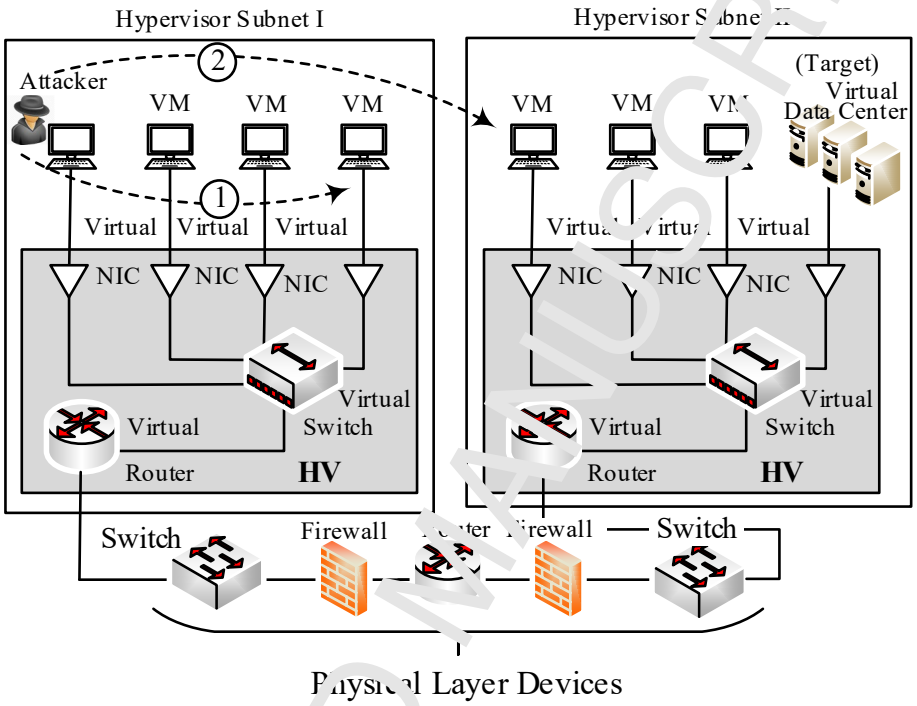

Fig. 9. Attack scenarios from two different subnets

Internal cloud subnet

Upon infection on a local subnet, WannaCry spawns two threads which scan the local and external subnets. The first thread uses the GetAdapterInfo() function to retrieve local subnet details such as subnet mask and network range. Local subnet scan is mutltithreaded and limited to 10 IP addresses per scan. The thread seeks to establish a connection on port 445 [49] to exploit CVE-2017-0143 if the SMB vulnerability is present on any host in the scanned IP addresses. This attack scenario corresponds to the first attack path of figure 4; $P_1: e_{[0,1]} \rightarrow e_{[1,v2]}$. In our use case, we infect a vulnerable host in the subnet with RDP backdoor vulnerability [48]. This corresponds to the attack edge $e_{[0,1]}$. Whereupon a vulnerable host is found, the malware infects the host and launches encryption of the files. This denoted by the attack edge $e_{[1,v2]}$. Thefore, this attack scenario generates the values:

$$[D_{eff}]_{G_{i,j}}^{\Pi}(e_{[i,j]}) = 0.430$$

$$St_{\sum P} = k = 2$$

Despite the limit on the number of IP addresses scanned at an instance in the first thread, the ransomware generates network traffic which we capture via Wireshark. The earlier traffic before SMB scans is general traffic related to name lookups, ARP, DHCP requests etc. The diagram below in figure 10 shows network traffic capture with SMB traffic denoted in red. Clearly, the presence of huge amounts of SMBv1 traffic on port 445 is an Indicator of Compromise (IOC) from the Eternal Blue seeking to exploit the vulnerability in SMB version 1 through which the malware propagates to other hosts on the network.

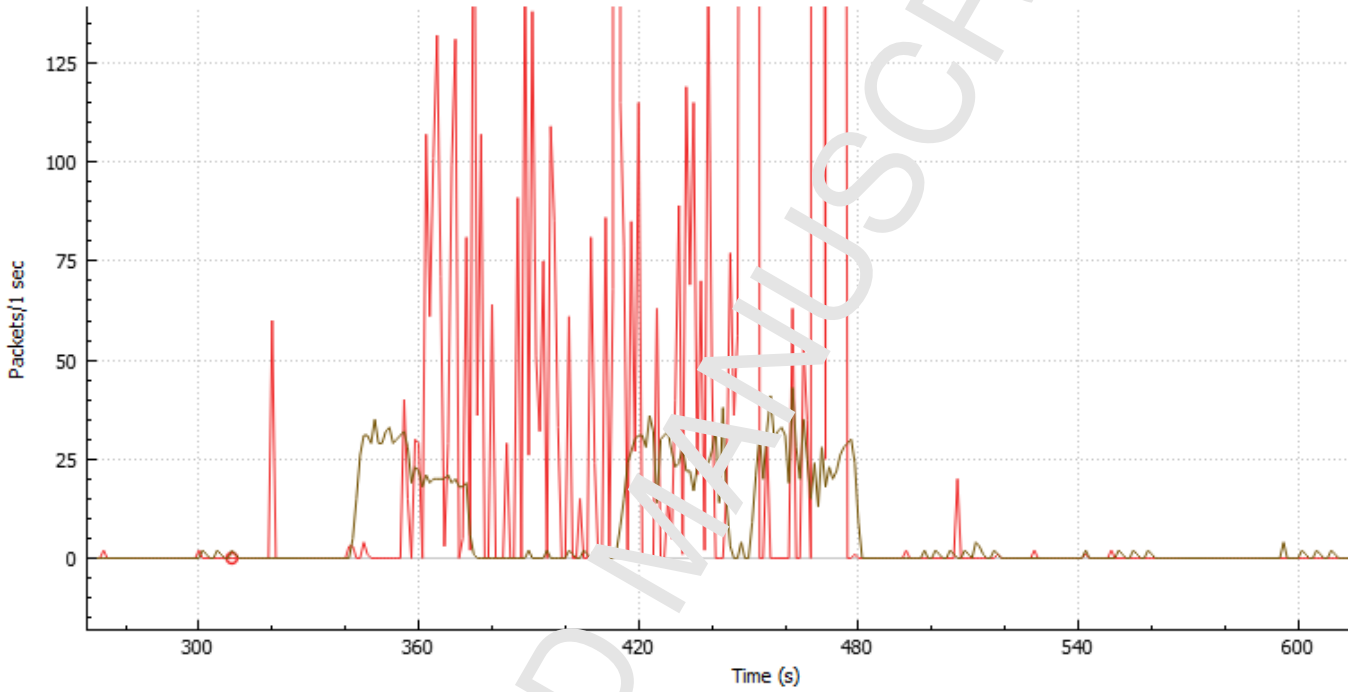

Fig. 10. Network activity capture of the ransomware with SMB scans on port 445

External cloud subnet

In the second attack scenario where the target resides in a different subnet, the second thread generates a list of external IP addresses ranges to be scanned and probes for a connection on port 445. The overall generated attack path is equivalent to attack path; P_4: $e_{[0,2]} \rightarrow e_{[2,v1]} \rightarrow e_{[v1,v2]}$. The attack paths for this scenario generates the following values:

$$[D_{eff}]_{G_{i,j}}^{\prod}(e_{[i,j]}) = 0.348$$

$$St_{\sum P} = k = 3$$

Another attack scenario utilizing a path identical to the above seeks to reach the target by exploiting another SMB vulnerability CVE-2017-0148. This particular attack path generates the values:

$$[D_{eff}]_{G_{i,j}}^{\prod}(e_{[i,j]}) = 0.400$$

$$St_{\sum P} = k = 3$$

The corresponding probability density curves for the above scenarios are shown in figure 11 below.

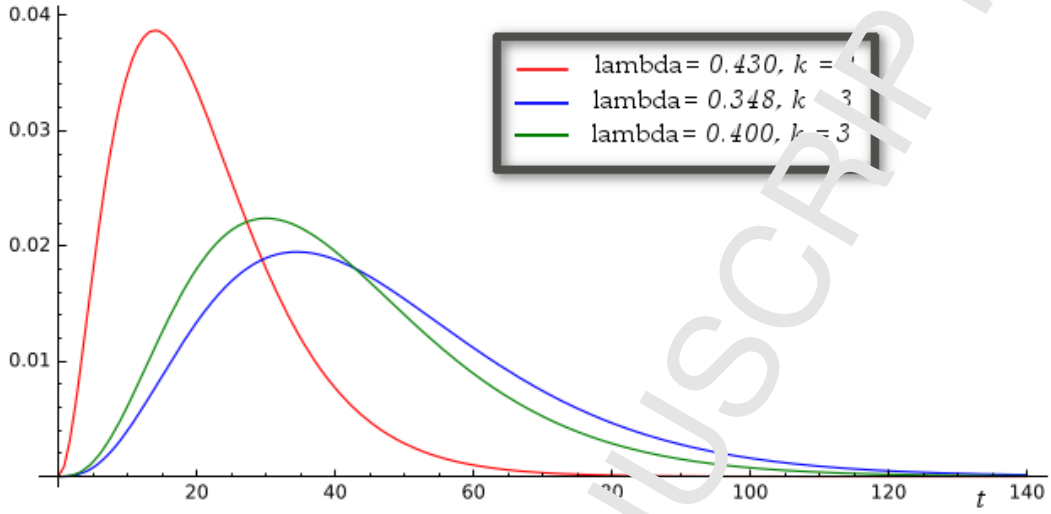

Fig. 11. Probability density curves for the 3 attack instances

All the three density curves are positively skewed denoting intensified attack activities in the early stages of the attack. This is a typical feature of the WannaCry ransomware as earlier explained. The first attack scenario corresponds to the red curve where the mean and median of the distribution lie after the mode. The mode falls in the last category of the time expense partition in Equation (24) to indicate a $\{TE\}_1$ of low. The second attack scenario comprising two attack instances generates the green and blue curve. Both curves are positively skewed where the mode comes before the mean and the median. Though the two graphs have the same k value (implying a tie) and a $\{TE\}_1 = low$, it's visible from the resultant graphs that the second attack path has the shortest route considering the location of the mode. It's worth noting that though these two paths exhibit $\{TE\}_2 = low$, the first attack has lower metrics both in terms of attack time expense $\{TE\}_1$ and parameteric value k, hence the shortest path.

5 Conclusion

In this paper, we have illustrated a quantitative way of characterizing APTs by chaining vulnerabilities of Bayesian attack network nodes derived from a cloud structure partitioned into three layers; application, virtual and physical layers. The main contributions of this paper are threefold and summarized as follows: (1) Quantitative characterization of possible attack paths from an attacker's view which is generically expressed as a parameter of two arguments – the cumulative edge weight, and summation of atomic attack steps. Unlike the popular approach of assigning probabilities to nodes, the probabilities of exploitation are mapped to a weight matrix which depicts all paths to the target node reflective of the existent vulnerabilities in the real system. The parameters express effectively the overall likelihood of an APT reaching the target in

a multiple attack sources cloud environment. (2) A proposed shortest path algorithm for finding the shortest attack path from multiple sources based on key nodes and key edges. The algorithm not only finds the shortest attack path but it's also capable of resolving ties amongst paths of equal weights by selecting the path with the largest effective distance. Our proposed algorithm thus does not treat two paths as identical if they only share same weight. (3) Characterization of the attack time expense of the APT by modeling the associated atomic attack events of a path as Poisson variables obeying the Erlang distribution partitioned into three levels: *High*, *Medium* and *Low*. Though APT attacks generally extend for longer periods of time, ransomware attacks on cloud production environment exhibits very short attack duration. Paths considered in our modeling can only be compared for evaluation of the shortest path if they share the same key edges and key nodes because the combination of these two parameters brings the breached security status of the victim to a defined state, i.e. breaches of specific CIA tenets or the combination thereof, regardless of the traversed path in the Bayesian attack network.

ACKNOWLEDGMENT

This work is supported by the Fundamental Research Funds for the Central Universities (TW201706) from the Ministry of Education of China.

REFERENCES

[1] F. Zafar, A. Khan, S.R. Malik, M. Ahmed, A. Anjum, M.I. Khan, N. Javed, M. Alam, and F. Jamil. "A survey of cloud computing data integrity schemes: Design challenges, taxonomy and future trends." Computers & Security 65 (2017): pp. 29-49.

[2] I. Friedberg, F. Skopik, G. Settanni, and R. Fiedler. "Combating advanced persistent threats: From network event correlation to incident detection." Computers & Security 48 (2015): pp. 35-57.

[3] R. Jain and S. Paul. "Network virtualization and software defined networking for cloud computing: a survey." IEEE Communications Magazine 51, no. 11 (2013): pp. 24-31.

[4] M. Frigault, et. al., "Measuring network security using dynamic bayesian network," in Proceedings of the 4th ACM workshop on Quality of protection, 2008, pp. 23–30.

[5] A. Singhal and X. Ou, Security Risk Analysis Of Enterprise Networks Using Probabilistic Attack Graphs, National Institute of Science and Technology Interagency Report 7788, Gaithersburg, MD, August 2011.

[6] R. Kumar, E. Ruijters, and M. Stoelinga. "Quantitative attack tree analysis via priced timed automata." In International Conference on Formal Modeling and Analysis of Timed Systems, pp. 156-171. Springer, Cham, 2015.

[7] M. Angelini, N. Prigent, and G. Santucci. "Percival: proactive and reactive attack and response assessment for cyber incidents using visual analytics." In Visualization for Cyber Security (VizSec), 2015 IEEE Symposium on, pp. 1-8. IEEE, 2015.

[8] K. Ingols, R. Lippmann, and K. Piwowarski. "Practical attack graph generation for network defense." In Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual, pp. 121-130. IEEE, 2006.

[9] N Perlroth, M Scott and S Frenkel. Cyberattack hits Ukraine then spreads internationally. (June 27, 2017) [Online] Available: https://www.nytimes.com/2017/06/27/technology/ransomware-hackers.html [Accessed February 17, 2018]

[10] S.E. Yusuf, M. Ge, J. B. Hong, H. Alzaid, and D.S. Kim. "Evaluating the Effectiveness of Security Metrics for Dynamic Networks." In Trustcom/BigDataSE/ICESS, 2017 IEEE, pp. 277-284. IEEE, 2017.

[11] X. Yang, T. Zhang, L. Yang, L. Wen, and Y.Y. Tang. "Maximizing the effectiveness of an advanced persistent threat." arXiv preprint arXiv:1707.02437 (2017).

[12] I. Arce and G. Richarte. "State of the art security from attacker's viewpoint." In PacSec Conference, Tokyo, Japan, 2003.

[13] Y.L. Huang, C. Borting , W.S. Ming, and Y.L. Chien. "Security impacts of virtualization on a network testbed." In Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference on, pp. 71-77. IEEE, 2012.

[14] L.E. Olson, M.J. Rosulek, and M.Winslett. "Harvesting credentials in trust negotiation as an honest-but-curious adversary." In Proceedings of the 2007 ACM workshop on Privacy in electronic society, pp. 64-67. ACM, 2007.

[15] B.M. Salem, S. Hershkop, and S. J. Stolfo. "A survey of insider attack detection research." In Insider Attack and Cyber Security, pp. 69-90. Springer US, 2008.

[16] D. Gonzales, J. Kaplan, E. Saltzman, Z. Winkelman, and D. Woods. "Cloud-trust-a security assessment model for infrastructure as a service (IaaS) clouds." IEEE Transactions on Cloud Computing. 2015.

[17] M. Sachin, E. Daniel, and N. A. Vasanthi. "Survey on various data integrity attacks in cloud environment and the solutions." In Circuits, Power and Computing Technologies (ICCPCT), 2013 International Conference on, pp. 1076-1081. IEEE, 2013.

[18] L. Huan. "A new form of DOS attack in cloud and its avoidance mechanism." In Proceedings of the 2010 ACM workshop on Cloud computing security workshop, pp. 65-76. ACM, 2010.

[19] J. J. Sah and L.J Malik. "Impact of DDOS attacks on cloud environment." IJRCCT 2, no. 7, pp.362-365. 2013.

[20] F. Massimo, and R. Massimiliano. "Stealthy denial of service strategy in cloud computing." IEEE Transactions on Cloud Computing 3, no. 1, pp. 80-94. 2015.

[21] M.I. Gofman, R. Luo, P. Yang, and K. Gopalan. "Sparc: a security and privacy aware virtual machinecheckpointing mechanism." In Proceedings of the 10th annual ACM workshop on Privacy in the electronic society, pp. 115-124. ACM, 2011.

[22] T. Ristenpart, E.Tromer, H. Shacham, and S. Savage. "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds." In Proceedings of the 16th ACM conference on Computer and communications security, pp. 199-212. ACM, 2009.

[23] J. Rutkowska, and A. Tereshkin. "Bluepilling the xen hypervisor." Black Hat USA (2008).

[24] D. Goodin. 1st August 2016. "New cloud attack takes full control of virtual machines with little effort" [Online] Available https://arstechnica.com/security/2016/08/new-attack-steals-private-crypto-keys-by-corrupting-data-in-computer-memory/ [Accessed 20th December 2017]

[25] C. Gebhardt, C. I. Dalton, and R. Brown. "Preventing hypervisor-based rootkits with trusted execution technology." Network Security 2008, no. 11 (2008): 7-12.

[26] J. Sahoo, S. Mohapatra, and R. Lath. "Virtualization: A survey on concepts, taxonomy and associated security issues." In Computer and Network Technology (ICCNT), 2010 Second International Conference on, pp. 222-226. IEEE, 2010.

[27] B. Wang, Y. Zheng, W. Lou, and Y. Thomas Hou. "DDoS attack protection in the era of cloud computing and software-defined networking." Computer Networks 81 (2015): 308-319.

[28] V. Varadarajan, T. Kooburat, B. Farley, T. Ristenpart, and M. M. Swift. "Resource-freeing attacks: improve your cloud performance (at your neighbor's expense)." In Proceedings of the 2012 ACM conference on Computer and communications security, pp. 281-292. ACM, 2012.

[29] S.N. Brohi, M.A. Bamiah, M.N. Brohi, and R. Kamran. "Identifying and analyzing security threats to Virtualized Cloud Computing Infrastructures." In Cloud Computing Technologies, Applications and Management (ICCCTAM), 2012 International Conference on, pp. 151-155. IEEE, 2012.

[30] B. Schneier. (8th March, 2012.) "Cloud Computing is a Man-in-the-Middle Attack." [Online] Available: https://www.schneier.com/blog/archives/2012/03/cloud_computing_1.html [Accessed 23rd December 2017]

[31] V. Ashktorab, and S. R. Taghizadeh. "Security threats and countermeasures in cloud computing." International Journal of Application or Innovation in Engineering & Management (IJAIEM) 1, no. 2 (2012): 234-245.

[32] Y. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart. "Cross-tenant side-channel attacks in PaaS clouds." In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. s990-1003. ACM, 2014.

[33] C.H. Kao, J.H. Dai, R.K , Y.T. Kuang, C.P. Lai, and C.H. Mao. "MITC Viz: Visual Analytics for Man-in-the-Cloud Threats Awareness." In Computer Symposium (ICS), 2016 International, pp. 306-311. IEEE, 2016.

[34] "Replay Attacks." Microsoft Security. [Online] Available : https://msdn.microsoft.com/en-us/library/aa738652(v=vs.110).aspx [Accessed 14th January 2018]

[35] N. Singh, A. Jangra, U. Lakhina, and R. Sharma. "SQL Injection Attack Detection & Prevention over Cloud Services." International Journal of Computer Science and Information Security 14, no. 4 (2016): 256.

[36] B. Joshi, A. S. Vijayan and B.K. Joshi. "Securing cloud computing environment against DDoS attacks." In Computer Communication and Informatics (ICCCI), 2012 International Conference on, pp. 1-5. IEEE, 2012.

[37] A. Chonka, Y. Yang, W. Zhou, and A. Bonti. "Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks." Journal of Network and Computer Applications 34, no. 4, pp. 1097-1107. 2011.

[38] B. Wang, W. Song, W. Lou, and Y. T. Hou. "Inverted index based multi-keyword public-key searchable encryption with strong privacy guarantee." In Computer Communications (INFOCOM), 2015 IEEE Conference on, pp. 2092-2100. IEEE,2015.

[39] CVE-2017-0181. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0181

[40] CVE-2017-0075. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0075

[41] CVE-2016-8930. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-8930

[42] CVE-2017-8056. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-8056

[43] CVE-2017-5681. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-5681

[44] CVE-2015-3456. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-3456

[45] CVE-2016-8931. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-8931

[46] CVE-2017-3623. (2017) [Online] Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-3623

[47] CVE-2015-1647.(2017)[Online]Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-1647

[48] A. Zimba, Z. Wang,"Malware-Free Intrusions: Exploitation of Built-in Pre-Authentication Services for APT Attack Vectors", International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.7, pp.1-10, 2017.DOI: 10.5815/ijcnis.2017.07.01

[49] WannaCry Update: Vulnerable SMB Shares Are Widely Deployed And People Are Scanning For Them. (May 2017). [Online] Available: https://community.rapid7.com/community/infosec/blog/2017/05/16/update-on-wannacry-vulnerable-smb-shares-are-widely-deployed-and-people-are-scanning-for-them

Author Bibliography

Zhaoshun Wang is a Professor and the Associate Head of the Department of Computer Science and Technology at the University of Science and Technology Beijing. He graduated from Department of Mathematics at Beijing Normal University in 1993. He received his PhD from Beijing University of Science and Technology in 2002. He completed postdoctoral research work at the Graduate School of the Chinese Academy of Sciences in 2006. He holds patents and has many awards to his name. His main research areas include Information Security, Computer Architecture and Software Engineering.

Hongsong Chen received his PhD degree in Department of Computer Science from Harbin Institute of Technology, China, in 2006. He was a visiting scholar in Purdue University from 2013-2014. He is currently an associate professor in Department of Computer Science, University of Science and Technology Beijing, China. His current research interests include wireless network security, attack and detection models, and cloud computing security.

Aaron Zimba is lecturer at Mulungushi University and he is currently pursuing PhD studies at the University of Science and Technology Beijing in the Department of Computer Science and Technology. He received his Master and Bachelor of Science degrees from the St. Petersburg Electrotechnical University in St. Petersburg in 2009 and 2007 respectively. He is also a member of the IEEE. His main research interests include Network and Information Security, Network Security Models, Cloud Computing Security and Malware Analysis.

- Proposition of a Bayesian network based weighted attack paths modeling technique for cyber-attack paths in cloud computing
- An optimized algorithm that evaluates the shortest attack path from multiple attack sources based on key nodes and key edges
- Characterization of attack time expense of the APT attack by modeling the associated atomic attack events in a path as Poisson variables obeying the Erlang distribution.
- Generated probability density distributions for various attack paths depicting the associated success rates and attack complexity