Estimating Posterior Probabilities to Identify Breach Points during Cyber-attacks

Sriram Raghavan
Department of Computing,
University of Melbourne
Parkville, VIC 2010 Australia
Email: sriram.raghavan@unimelb.edu.au

S V Raghavan
Department of Computer Science & Engineering
Indian Institute of Technology Madras
Chennai, INDIA
Email: sv.raghavan@gov.in

***Abstract*—When a security incident is reported, identifying the point-of-breach is critical to figure out what an attacker might do next. Besides, the information related to the resources and levels of privilege acquired as a result of the breach, is essential to re-create the process. Such information is also crucial to planning a defense in real-time. To this end, we need to determine the following parameters associated with the system under attack, viz., a *breach probability* determined from the success of past cyber-attacks, the *attack matrix* which is dependent on the system design that decides the types of attacks and the associated modalities that the system may be susceptible to under standard conditions, and the *access matrix* which is determined by the access privileges that get granted when an attack succeeds.**

In this paper, we introduce the *breach-point detection problem* and present a model based on Bayes' conditional probability to identify the point-of-breach on a system impacted by a cyber-attack. We evaluate the probability associated with likely points-of-breach in the rest of the system based on the knowledge of its design and estimate the posterior probabilities associated with the points so identified based on the system parameters outlined earlier. We also determine the most probable point-of-breach and its posterior probability using our model. We demonstrate the use of our model by way of a numerical example. Finally, we illustrate the generalization of this approach to an arbitrary system and outline a method to compute its system parameters, viz., *the breach probability, the attack matrix and the access matrix.*

***Index Terms*—*Attack matrix, Access matrix, Breach probability, Bayes posterior probability, point-of-breach, Breach-point detection problem*.**

I. Introduction

In recent times, the security awareness is definitely on the rise; as a result more investments are going towards protecting the infrastructure and ensuring integrity of the data that is stored in e-infrastructure. Besides, with security solutions such as SIEMs (Security Information & Event Management), log event managers and log correlation engines, enable logging of more events across an organization effectively for monitoring and later analysis. However, due to the multiplicity of security policies and security solutions, it still remains a challenge to pin-point the point of breach, when a cyber attack penetrates through an organization's defenses. To identify the fault and gather evidence of the breach, the security & forensic analysts must analyze very large volumes of logs across different systems and correlate the data, which is time-consuming. Typically, the detection of a breach occur post-hoc when the logs across multiple connected subsystems are correlated for analysis. However, it doesn't often provide evidence of how one might have perpetrated. One often wonders whether it is possible to do so in real-time by predicting breach-points in a system *a priori* or during its operation.

A. The emergence of the Breach-point detection problem

Increasingly, a lot of intelligence is being built into security devices and monitoring systems, including log engines, log correlation devices such as SIEMs, UTMs (Unified Threat Management) or NGFWs (Next-generation firewalls). The NIST [9] outlines the principles underlining the computer log event management and event correlation to detect anomalous activities on a system. The intelligence includes network packet signatures, intrusion detection signatures, log correlation sequences, firewall pre-sets (network port configurations), and many more [2, 4, 7, 14, 15]. Liao et al. [8] discuss how the latest technologies are evolving in adapting to such attacks. In such an evolving scene, we are not far away from achieving near real-time attack detection. In fact, the challenge in cybersecurity is gradually shifting from one of detection to one of determining where and how a perpetration occurred, to pave the way for near real-time response to attacks: we call this the *Breach-point detection problem*. When the breach-detection is to be done in near real-time, the breach-point detection problem can be posed as a *Bayesian conditional inference* problem [13]. This would allow us to postulate as to how an attacker might have breached the system, evaluate different scenarios based on the system design (and possibly its configuration) and determine the most probable scenario describing the sequence of events that transpired, thereby enabling real-time defense.

B. Attack Types & Modalities

Largely, the access to an organization may be gained through one of four methods, viz., *access control systems*, *software vulnerabilities*, *brute force attacks* and *malware exploits*. The access control systems method is where the authentication and authorization stages in granting access are exploited. The software vulnerabilities method is one of most popular methods in conventional day security exploits where some inherent system design flaw grants access to an unauthorized access request. The brute force attack method is one where a cyber attack (on any part of the access control sub-system) renders the access control system ineffective to perform regular operations, thereby allowing unrestricted access to an unauthenticated access request. The malware exploits method is yet another popular method, which inserts itself into a regular user's system in order to snoop on private and personal information and uses the same for gaining legal access to the system. As an example, the OPM breach (https://www.opm.gov/cybersecurity) in early 2015 was perpetrated using attack method 4 (malware exploits) and used an attack modality involving malicious code exploits to gain access to the federal database of the personnel management.

In literature, there are many modalities [5, 7, 12] for each of the methods discussed here and their specific implementation may often impact their effectiveness to gain authorized or unauthorized access to the system. The natural question that arises, "Is it possible to determine the point of breach and the modality?"

In this paper, we propose a model and a method to identify the point of access breach to determine the most probable access method/modality adopted. Besides, we apply a Bayesian inference method and predict the next action of the attacker so that one can possibly intercept and quarantine the attacker. While doing so, we also record relevant evidence against the attacker, which may be used during a digital forensic investigation of the incident. We illustrate our algorithm using a probabilistic distribution across different access breach methods. While it may appear that near real-time attack detection is still in our future, we do so because, NIST and CERT announcements [9, 21, 22] indicate that the latency in detecting a breach/cyber-attack and being able to respond (or patch the system vulnerability) can be significant. In some case cases, this can be a matter of days, if not months. In fact, the Heartbleed and Shellshock are two examples where the latency in launching a response was 5-7 days.

The rest of this paper is organized as follows. In Section II, we review related work to highlight the growing importance of the breach-point detection problem. In Section III, we develop a model to detect breach-points during an attack using Bayesian conditional probability theory. In Section IV, we explore possible breaches from then on, using the attack matrix and the access matrix and illustrate the process with a numerical example. In Section V, we attempt to generalize the model to an arbitrary system and discuss the process of determining the three system parameters. In Section VI, we summarize our work and identify scope for further research in this emerging area.

II. Related Work

Bagchi and Udo [1] study the growth rates of different types of computer and Internet-related crimes using a modified Gompertz model to analyze sparse data. The Gompertz model is an appropriate diffusion model because it is capable of modeling two opposite behaviors: (1) acts of attacks and imitation of attacks and (2) deterrence acts to prevent such attacks. In addition, this model can handle sparse data adequately. Their study concluded that growth patterns of computer and Internet crimes differ in growth patterns and that a relationship exists between occurrences of such security breaches and uses of certain security technologies. Liu and Cheng [6] present an overview of the cause of cybersecurity problems and analyze the challenges associated with it. They also outline what constitutes a cyber-attacker profile, discuss cyber-attack patterns, and summarize recent cyber-attack trends. Kotenko and Chechulin [11] present a framework for attack modeling and security evaluation in Security Information and Event Management (SIEM) systems based on modeling of a malefactor's behavior, generating a common attack graph, calculating different security metrics and risk analysis procedures. Key elements of their suggested are using a comprehensive security repository, effective attack graph (tree) generation techniques, taking into account known and new attacks based on zero-day vulnerabilities, stochastic analytical modeling, and interactive decision support to choose preferred security solutions. Attack graphs depict ways in which an adversary exploits system vulnerabilities to achieve a desired state [19]. System administrators use attack graphs to determine how vulnerable their systems are and to determine what security measures to deploy to defend their systems.

Baumgartner et al. [2] present a new approach for reactive security monitoring in a virtualized computer environment based on minimally intrusive dynamic sensors deployed vertically across virtualization layers and horizontally within a virtual machine instance. The sensor streams are analyzed using a federation of complex event processing engines to maximize the performance of continuous queries, and the results of the analysis are used to trigger appropriate actions on different virtualization layers in response to detected security anomalies. As indicated in Section I, the work by Baumgartner et al. highlights the recent advances in attack detection aligning towards near real-time detection with advanced analytics.

There has been a fair amount of reported literature in the art of detecting network intrusions. Largely such works have focused on activity based abnormality detection in the network traffic. Garcia-Teodoro et al. [7] provide a review of the most well known anomaly-based intrusion detection techniques and study the available platforms, systems under development and research projects in the area. They also outline the main challenges in wide-scale deployment of anomaly-based intrusion detectors. Marpaung et al. [14] discuss some of the techniques adopted by intrusion and malware attackers in adaptive packet signatures and malware embedding techniques on network packets. Liao et al [8] note that current intrusion detection systems (IDS) pose challenges on not only capricious intrusion categories, but also huge computational power. They provide a taxonomy to modern IDS designs and the methods adopted for detection. Myers et al. [15] present a distributed event correlation system, which performs security event detection, with experimental evaluation, taking into account network bandwidth utilization, detection capability and query efficiency, in comparison with a centralized alternative. They conclude that distributed processing environment can provide up to 99% reduction of network syslog traffic allowing for near real-time detection. Kim et al. [10] present an abnormal network traffic detecting method by aggregating packets that belong to the identical flow and a detection algorithm using changes in traffic patterns that appear during attacks. Their algorithm is resistant to mutant attacks which use alternate port numbers or payloads.

Carl et al. [5] present a survey of detection techniques and testing results providing insight into the ability to successfully identify Denial-of-Service flooding attacks. Their survey observes that while each detector has shown promise, none completely solve the detection problem. In such a scenario, combining various approaches was most likely to produce the best results. Zimmer et al. [20] present three mechanisms for time-based intrusion detection, and more specifically, they detect the execution of unauthorized instructions in real-time CPS environments utilizing information obtained by static timing analysis.

Notwithstanding multiple security policies and security solutions that are in place, it remains a challenge to pinpoint the point of breach when a cyber attack penetrates an organization's defenses. However, we can pose this problem as a *Bayesian conditional inference* problem, if we assume the ability to detect a breach immediately. To the best of the authors' knowledge, there is no work that has been directed at detecting points-of-breach in cyber-attacks, particularly using a Bayesian inference approach. Our work attempts to provide some insight in developing such an approach as it naturally leads to building better defenses dynamically.

III. Modeling an Access breach using Conditional Probability

Consider some system S as the subject of our study which contains an access control system. The access control system can be breached using all the four methods outlined in Section I and each method contains an arbitrary number of implementations, each of which have a known probability of a breach attempt succeeding. We further assume that a breach can be detected immediately; as soon as a breach occurs, it sets off an alarm in the access control system (although how it gets set each time is the subject of breach detection techniques that is beyond the scope of this work). When an attacker breaches the security, the attacker will find oneself at a known location within the organization's network infrastructure. For instance, if the attacker targets the web server, the breach will lead the attacker to access to the public contents of the WWW server. If on the other hand, an attacker gains access using an SQL injection or code injection, the attacker may find oneself in the Database server with access to the public folders within that system. Once an attacker breaches the security, one may choose to do one of three things: (*i*) the attacker may choose to remain there; or (*ii*) proceed to attack further levels of access control to penetrate deeper (access neighbouring systems each having similar access control implementations); or (*iii*) access files for copying/deleting at the current level.

A. The Attack Matrix

We represent the access control system as a rectangular matrix (called the *access matrix*) of four rows, one for each method of breach, and each column representing the specific implementation that is used. To re-iterate our position, attack types correspond to one of the four classes (*Ref. Section I B*) of cyber-attacks a system may be subjected to while the attack modalities correspond to the specification of how a particular attack is carried out against the system. As an example, if a cyber-attack is conducted using the Heartbleed bug (CVE-2014-0160), it is based on attack type 2 (software vulnerability) that uses the attack modality involving memory overflow to dump system's memory contents on the network.

The values themselves will contain the probabilities of a breach attempt succeeding and this is normalized across all methods and implementations to the one-sided interval [0, 1.0). The attack mode or vector which causes the breach is also represented as a rectangular matrix whose only value will correspond to the probability of that attack succeeding. For a given column and row entry, if the attack vector value is greater than or equal to the value in the corresponding access control system, the attack is deemed to have succeeded. As we can discern from this discussion, the attack matrix is a system parameter and is fixed for a particular configuration of the system S, based on the nature of defences set in place against cyber-attacks.

Where an attacker may conduct two or more simultaneous modes of attack, each attack vector is independently represented as a rectangular matrix and compared against the corresponding entry on the access control matrix. If two or more attacks simultaneously succeed, the attacker is granted access to the system with the lower index on the *access matrix*. The attack matrix is shown in Table I where the rows correspond to one of four attack methods outlined in Section I and the columns correspond to the attack modalities for each attack method that is chosen to breach the system S.

TABLE I. The attack matrix

Attack Method	Attack Modality				
	1	**2**	**3**	**…**	**n**
1	p_{11}	p_{12}	p_{13}	…	p_{1n}
2	p_{21}	p_{22}	p_{23}	…	p_{2n}
3	p_{31}	p_{32}	p_{33}	…	p_{3n}
4	p_{41}	p_{42}	p_{43}	…	p_{4n}

Each entry in a particular row and column indicates the probability of the system S *giving in* to an attack conducted using the method from the corresponding row and the attack modality from the corresponding column, p_{13} indicates the probability with which an attack on the access control system using some modality as described by column 3 would penetrate the system's defence. One possible method to determine the probabilities is the use of penetration testing, which is common in the certification process.

Each entry is a probability value in the one-sided closed interval [0.0, 1.0). A probability entry having a value zero indicates that that particular attack modality is not supported under the attack method indicated by the respective row entry. No individual attack modality, irrespective of the attack method chosen has a success probability of 1.0, and hence the open interval. This explanation for the probability entries is further substantiated in the numerical example discussed in Section V. The following relationships hold true for the entries listed in Table I.

$$\begin{aligned} p_{11}+p_{12}+p_{13}+\ldots+p_{1n} &= p_1 \\ p_{21}+p_{22}+p_{23}+\ldots+p_{2n} &= p_2 \\ p_{31}+p_{32}+p_{33}+\ldots+p_{3n} &= p_3 \\ p_{41}+p_{42}+p_{43}+\ldots+p_{4n} &= p_4 \end{aligned} \quad \ldots (1)$$

$$p_1+p_2+p_3+p_4 = p \leq 1.0 \quad \ldots (2)$$

Although individual entries in the matrix can have a zero, each individual attack method has a non-zero probability in practice, i.e., none of method probabilities p_1, p_2, p_3 or p_4 have a value zero. As a consequence, the relationships described in eqn. (3) hold good.

$$\begin{aligned} p_1 &< 1.0 \\ p_2 &< 1.0 \\ p_3 &< 1.0 \\ p_4 &< 1.0 \end{aligned} \quad \ldots (3)$$

Once an attack succeeds and the alarm is raised, we have two tasks, one to determine how the attacker gained entry and another to predict based on this entry condition, where the attacker may be headed. For the first case, we compute the Bayesian conditional probability using the attack breach probabilities as reference values and rank the breach points in descending order of success probabilities.

B. Estimating Posterior Probabilities

In *Bayesian conditional probability*, we treat the parameter θ as a random variable, with a given probability density function *h(θ)* for $\theta \in \Theta$. The corresponding distribution is called the *prior distribution* of θ and is intended to reflect our knowledge (if any) of the parameter, *before* we gather data. After observing $x \in$ S, we use the Bayes' theorem, to compute the conditional probability density function of θ given X = x. Recall that the joint probability density function of (X, θ) is the mapping on S × Θ given by

$$\left(x,\ \theta\right)\mapsto \qquad \text{˙}(x\mid\theta)$$

Next recall that the (marginal) probability density function f of X is given by

$$f\left(x\right)=\sum_{\theta\in\Theta} h\left(\theta\right) f\left(x\mid\theta\right),\ x\in \mathrm{S} \quad \ldots (4)$$

…if the parameter has a discrete distribution, or

$$f\left(x\right)=\int_{\theta\in\Theta} h\left(\theta\right) f(x\mid\theta)\, d\theta,\ x\in \mathrm{S} \quad \ldots (5)$$

…if the parameter has a continuous distribution. Finally, the conditional probability density function of θ given X = x is

$$h(\theta\mid x) \ = \ h\left(\theta\right) f(x\mid\theta) f\left(x\right);\ \theta\in\Theta,\ x\in S \ldots (6)$$

The conditional distribution of θ given X=x is called the *posterior distribution*, and is an updated distribution, given the information in the data. Note that *f(x)* is simply the *normalizing constant* for the function θ ↦ *h(θ) f(x|θ)*. It may not be necessary to explicitly compute *f(x)*, if one can recognize the functional form of *θ*↦ *h(θ) f(x|θ)* as that of a known distribution [3].

Finally, if θ is a real parameter, the conditional expected value E(θ|X) is the *Bayes' estimator* of θ. Recall that E(θ|X) is a function of X and, among all functions of X, is closest to θ in the mean square sense. Of course, once we collect the data and observe X = x, the *estimate* of θ is E(θ|X = x).

C. Setting up Attack Priors

Let θ be the parameter representing the method used in gaining unauthorized access to the system S. By definition, $\theta \in \Theta$, where Θ corresponds to the four methods of gaining unauthorized access on to system S. We treat θ as a random variable over Θ and it is distributed according to probability density function $h(\theta)$.

Let δ be the parameter representing the mode of attack that is used to gain the unauthorized access. Let $\delta \in \Delta$ where Δ corresponds to the set of all modalities available to an attacker to exploit system S. We treat δ as another random variable over Δ and it is distributed according to probability density function $k(\delta)$.

Now, *h*(*θ)* and *k*(*δ*) represent the priors for the random variables θ and δ respectively. When an attack succeeds and the event is observed, the modality of the attack is what is perceived at the event-level [13]. Based on this observation, the modality is mapped to corresponding value(s) in parameter θ. Since *h(θ)* is the method prior on θ across the entire distribution for the parameter $\delta \in \Delta$, we can express,

$$h(\theta)=\sum_{\delta\in\Delta} k\left(\delta\right) f\left(\theta\mid\delta\right),\ \theta\in\Theta \quad \ldots (7)$$

…where *f(θ|δ)* is the conditional probability function for variable θ given δ was observed. Note that in order to compute the joint probability on an observer event x using attack modality δ is given by the expression,

$$\left(x,\ \delta\right)\mapsto \qquad \text{˙}(x\mid\delta)$$

This can be interpreted as the joint probability of event where the system S was breached in a cyber-attack and it occurred using the attack modality δ. Notwithstanding the dependence of θ on δ, the joint probability p (θ, δ) spread over *h(θ)* and *k*(*δ*) provides us with the probability of system being breached under an attack, irrespective of the nature of the attack itself.

Therefore in order to discern the specifics of the attack, we need additional information to distinguish the breached system from one type of attack to another. This necessitates that we classify the system operation into different states of operation and identify the state the system is in, when it is breached so that this information can be used to isolate the attack modality from the attack method used by an attacker to breach the system defences. This is discussed further in the sequel.

IV. TRACKING SYSTEM STATES & BREACH DETECTION

The nature of breach and the level of access that is granted, especially when under attack, guide the states of operation of a system, to an attacker by virtue of the successful attack. When a system is attacked, it internally affects the state of system operation and that needs to be tracked. Naturally, some methods provide certain type of system access (e.g., *administrative access on server*) while others are precluded. We designate states of operation for the system – a system shall have a minimum of two states, i.e., one in standard operation and the other denotes the system that has been breached. It is important to know that under our definition, it is not necessary that a system that has been breached is exploited once the attack has succeeded, although that is not an essential pre-condition for the tracking to work accurately. No matter how insignificant, an attack intrinsically changes the system in operation, and that is reflected by the change in the system's state. Any additional states included in the system's operation correspond to alternate methods of breaching the system's defenses.

A. System States & Cyber-attacks

To ascertain the current state of the system, one must carefully analyze the changes that can be effected on a system during a cyber-attack. In order to ascertain the current state, one should be able to describe the different ways (corresponding to changes to system operating states) in which an attack can affect a system.

Broadly, there are three types of attack impacts to a system. One of the most popular goals of a cyber-attack is to gain unauthorized access to the system or any of its resources. Contemporary attacks that execute binaries (or malicious code) without the knowledge of the owner of a system that write or transmit data to or from the system belong to this type. Another popular type of attack is employed to gain elevated access privileges. Contemporary attacks that result in a regular system user gaining administrative privileges belong to this type. A third type of attack is used to completely isolate the system and flood its resources so it becomes inoperable. Contemporary DDoS type attacks and those that render the system resources unresponsive to administrative commands are classified under this type. We illustrate the application of our method with a numerical example in the sequel.

B. Estimating the Posterior Probabilities to identify the Breach point: Numerical example

Consider the following system S that consists of three major subsystems, i.e., a web server, a file and a database. The system S can be breached using two methods and three modalities, of which modality one does not apply to method two. The attack matrix for the system S is shown in Table II.

Let S have an overall breach probability of 0.12. This indicates that out of any given 100 attempts to breach the system, 12 attempts are likely to succeed. For the system S, based on the value of entry in row 1 and column 1, there is a 10% chance that a cyber-attack using attack method 1 and attack modality 1 may result in a breach.

TABLE II. ATTACK MATRIX FOR SYSTEM S

ATTACK METHOD	ATTACK MODALITY		
	1	**2**	**3**
1	0.10	0.05	0.05
2	0.0	0.10	0.05

Similarly, based on the value in row 3 and column 3, there is 5% chance that a cyber-attack using attack method 2 and attack modality 3 may result in a breach. The values shown in Table II reflect the general upkeep of the system with regard to its system design that results in one or more vulnerabilities during its implementation. It is these vulnerabilities that can be exploited during an attempted perpetration. However, it is generally expected that in a matured cyber environment, the system design and its implementation are clearly documented which in turn identify the vulnerabilities a system may be susceptible to [4, 6, 9, 16]. The NIST [6] outlines some of the requirements with regard to event logging, management and event correlation that may be implemented on federal and mission-critical systems.

TABLE III. ACCESS MATRIX FOR SYSTEM S

ATTACK METHOD	ATTACK MODALITY		
	Web	**File**	**Data base**
1	*root*	*user*	*user*
2	-	*root*	*user*

In order to maintain the values low, it is expected that such systems be regularly updated including the installation of security patches for the system as well as the applications that are installed. Additionally, it may require that one or more security devices such as those mentioned in Section I are deployed in order to protect the access to the system S and is periodically monitored using one or more logging engines and correlation tools [16, 18].

In accordance to the attack matrix, the access granted when a breach is successful is tabulated in the *access matrix* shown in Table III where **Web**, **File** and **Database** correspond to the different subsystems where an attacker may gain access when an attack is successful and the entry *root* or *user* corresponds to the level of access the attacker is likely to receive on the breached system. The entry "*root*" corresponds to administrative privileges on the subsystem while the entry "*user*" corresponds to normal user privileges on the subsystem.

Based on the constraints outlined from Tables II and III, one may deduce that an attacker who has breached the system and has gained administrative privileges to the Web server had used attack method 1 and modality 1. Any counter measures that need to be deployed can be in concurrence with the system policy to limit exposure to web data or other system based on this information. This can be modeled using Bayesian inference in the following manner,

Pr(S breached) = 0.12

Pr(attack method θ & modality δ used in attack attempt = $p_{\theta,\delta}$ (as indicated in Table II)

Pr {attack method θ & modality δ} succeeded | S breached) =

$$\frac{\Pr(\{\text{attack used } \theta,\delta\} \cap \text{ S breached})}{\Pr(\text{S breached})}$$

…where Pr (e) denotes probability of event e in some known sample space **U**. The probability value computed corresponds to the posterior probability for an attack conducted on system S. The Bayesian Posterior probability for the given scenario corresponds to the joint probability of using the attack pair {θ, δ}, S breached and the attacker landing on the Web subsystem with administrative privileges under the condition that S breached and Web root access are observed. The argand plane for this particular state space is examined for determining the exact values for the pair {θ, δ}, and is given by the expression,

$$\underset{\theta,\delta}{\arg\max}\{\text{Probability }\{\text{method} = \theta \,\&\, \text{ modality} = \delta \mid \text{ S breached } \& \text{ root on}\textbf{Web}\}\}$$

…where $\theta \in \{1, 2\}$ and $\delta \in \{1, 2, 3\}$. The probability value computed attains its maximum value for {$\theta = 1$, $\delta = 1$}. We generalize this approach in the sequel.

V. Generalizing the Bayesian inference model & Determining system parameters

In this section, we develop a framework to generalize the model described to any standard system S. To do this, given any system S, we must be able to identify the system parameters, such as the *breach probability*, the *attack matrix* and the corresponding *access matrix*. Once these parameters are known, suitable measures should be set in place to detect a breach and the attacker's relative position within S. Based on this information, we can then compute the conditional probability of the joint occurrence which can be used in turn to determine the posterior probability. We address the identification of the three system parameters in turn.

A. Determining the Breach probability

Typically, the system breach probability is informed from statistical system history based on past attacks on the system across an equal spread of attack types and their respective modalities. There are two main reasons why this is a very objective measure of the system's security. Firstly, it purely focusses on the fraction of attacks that succeeded irrespective of the incremental security countermeasures in place and assesses the risk for the system as whole rather than adopting a differential protection mechanism for the different subsystems. Secondly, if additional countermeasures were in play, then over time, the overall breach probability will saturate to a level that normalizes incremental deployment of these countermeasures which incorporates it into the formula to compute this breach probability.

Based on the system history, we identify the number of attack attempts on the system in a given period of time and the number of attempts that were successful in the same period. The ratio of the number of successful attacks to the total number of attempts within that same period provides us with a gross estimate of the system breach probability. The gross is used here to indicate that by no means is this value meant to be static and must regularly be updated using the running average of the average success rate based on the system history – but it does provide us with a starting point! We compute the running average of the breach probability at the (n+1)th stage of a system using eqn. (8) shown below.

$$b_{n+1} = \frac{n.b_n + r_{n+1}}{(n+1)} \quad \text{… (8)}$$

… where b_n is the breach probability after n iterations, r_{n+1} is the breach probability computed individually at the (n+1)th iteration.

B. Determining the Attack matrix

For a given configuration of a system S, the CVE database maintained by the MITRE Corporation (*http://www.mitre.org*) can provide one with a list of system vulnerabilities classified into one of four attack types as outlined in Section I of this paper. An exhaustive vulnerability assessment (as is carried out by vulnerability assessment specialists using a variety of tools like Nmap, Metasploit, Nessus, and so on) across all the types and different modalities to which the system is vulnerable is used to develop the attack matrix. In order to determine the specific values, the system is subjected to series of penetration tests covering the spread of attack types and modalities and the ratio of the number of successful attempts to breach the system using a particular combination of attack type and modality with respect to the total number of attempts made against the system provides us with the respective entries in the attack matrix.

Considering that the system is subject to continuous flux in the event of an attack, the system behavior is divided into epochs – each epoch targeted at a specific subsystem during which it is engaged in activity interacting with the other subsystems or outside world (i.e., the Internet). After each epoch, the elements corresponding to that subsystem are reassessed and the probabilities are re-computed for the corresponding elements in the matrix. The benefits of this approach are evident; it simplifies the number of computations necessary to maintain the currency of the matrix while evaluating an attack. Sheyner and Wing's work on attack graph generation techniques [19] to determine how vulnerable their systems are and to determine what security measures to deploy to defend their systems is also a useful approach; however, it is meant to identify the security countermeasures and apply it under steady-state, rather than when reacting to an attack.

C. Determining the Access Matrix

The task of determining the access matrix is closely coupled with the task of completing the attack matrix. For each vulnerability test conducted when exhausting listing the attack matrix, we identify the nature of access granted to an attacker when the system is successfully breached. The highest form of privileges the granted to the attacker across all types of system penetration attempted for each attack type and modality combination is listed in the access matrix.

VI. Conclusions & Future Work

In this paper, we introduced the breach-point detection problem and developed a Bayesian approach to determine the attack method and modality of a "*cyber-attack*". We evaluated the probability associated with likely points of entry into some given system S based on system parameters that include the breach probability, attack matrix and the access matrix. We estimated the posterior probabilities associated with potentially exploited vulnerabilities in the system being analyzed using Bayes' approach to identify evidence relevant to the investigation. This was demonstrated using a worked example on a system which was breached to determine the most likely method and attack modality used by an attacker to breach the system defenses. We also provided a discussion on generalizing this approach to an arbitrary system and the method to compute its various system parameters, viz., the breach probability, the attack matrix and the access matrix. The approach is intended for isolating compromised resource and identifying the most suitable course of action to the taken depending on how the attacker may have breached the system's defenses.

Detecting attacks immediately after they have succeeded has remained a challenge for several years now. The cyber-attackers employ a diverse range of tactics that allow them to operate as covertly as possible often leaving false trails. Our method assumes the ability to detect such breaches immediately on occurrence and being able to precisely pinpoint the sub-system and the level of access gained by virtue of such a breach. However, adopting some heuristics that can monitor of active users on each individual subsystem and the nature of commands executed can provide some indications of where an attacker might have gained access, if at all. It must be said when Rootkits and Trojans are inserted, this becomes a harder problem still. Our efforts are focused on making headway into these challenges. It also remains to be seen how this approach fares against a subjective scenario, which may often involve insiders or when zombie computers are used. The authors are currently investigating this problem.

References

[1] Bagchi, Kallol and Udo, Godwin (2003)., "An Analysis of the Growth of Computer and Internet Security Breaches". *Communications of the Association for Information Systems*: Vol. 12, Article 46, pp. 684-700.

[2] Baumgärtner L., Strack C., Hoßbach B., Seidemann M., Seeger B., and Freisleben B. (2015)., "Complex event processing for reactive security monitoring in virtualized computer systems", *In Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems* (DEBS '15). ACM, New York, NY, USA, 22-33.

[3] Beck, J. and Au, S., (2002)., "Bayesian updating of Structural Models and reliability using Markov Chain Monte-Carlo Simulation", *In the Journal of Engineering Mechanics*, *J. Eng. Mech., American Society of Civil Engineers*, Vol. 128(4), pp. 380-391.

[4] Butler M., (2009)., "Benchmarking the Security Information & Event Management", SANS Institute Whitepaper, available online at http://www.sans.org/reading-room/whitepapers/analyst/benchmarking-security-information-event-management-siem-34755, last accessed July 25, 2015

[5] Carl, G., Kesidis, G., Brooks, R.R., and Rai, S., (2006)., "Denia-of-service attack detection techniques", *In IEEE Transactions on Internet Computing*, IEEE publishers, Vol 10(1)., pp. 82-89.

[6] Demsey K., Chawla, N.S., Johnston A., Johnston R., Johns A.C., Orebaugh A., Scholl M. and Stine K., (2011)., "Information Security Continuous Monitoring for Federal Systems & Organizations", NIST US Dept. of Commerce, NIST Special Publication SP800-137, available online at http://csrc.nist.gov/publications/nistpubs/800-137/SP800-137-Final.pdf, last accessed June 25, 2015

[7] Garcia-Teodoro, P., Diaz-Verdejo, J., Marcia-Fernandez, G., and Vazquez, E., (2009)., "Anomaly-based network intrusion detection:Techniques, Systems & Challenges", *In Elsevier Transactions on Computers & Security*, Vol. 28(1-2), pp.18-28.

[8] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, Kuang-Yuan Tung (2013)., "Intrusion detection system: A comprehensive review", *Journal of Network and Computer Applications*, Vol. 36(1), January 2013, pp. 16-24.

[9] Kent K. and Souppaya M. (2006)., "Guide to Computer Security Log Management", NIST US Dept. of Commerce, NIST Special Publication SP800-92, available online at http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf, last accessed June 25, 2015

[10] Kim, Myung-Sup., Kong, Hun-Jeong, Hong, Seong-Cheol.,Chung, Seung-Hwa., and Hong, J.W., (2004)., "A Flow-based method for abnormal network traffic detection", In Proceedings of the Network Operations & Management Symposium (IEEE/IFIP NOMS 2004)., ISBN: 0-7803-8230-7, IEEE Publishers, Vol. 1, pp. 599-612.

[11] Kotenko, I.; Chechulin, A., "Common Framework for Attack Modeling and Security Evaluation in SIEM Systems," *In Proceedings of the 2012 IEEE International Conference on Green Computing and Communications (GreenCom),* pp.94-101.

[12] Liu, S., Cheng, B., (2009)., "Cyberattacks: Why, What, Who, and How", *IT Professional*, vol.11, no. 3, pp. 14-21, May/June 2009, doi:10.1109/MITP.2009.46

[13] Lou, Z., Loparo, K.A., (2004)., "Bearing fault diagnosis based on wavelet transform and fuzzy inference", *In Elsevier Transactions on Mechanical Systems and Signal Processing*, Vol 18(5), pp. 1077-1095.

[14] Marpaung, J.A.P.; Sain, M.; Hoon-Jae Lee (2012)., "Survey on malware evasion techniques: State of the art and challenges," *In Proceedings of the 14th IEEE International Conference on*

Advanced Communication Technology (ICACT 2012), Print ISBN: 978-1-4673-0150-3, pp.744-749.

[15] Myers, J.; Grimaila, M.R.; Mills, R.F., "Log-Based Distributed Security Event Detection Using Simple Event Correlator," *In Proceedings of the 44th Hawaii International Conference on System Sciences (HICSS 2011)*, pp.1-7.

[16] Nicolett M. and Kavanaugh K. M. (2011)., "Magic Quadrant for Security Information & Event Management", *Gartner Research Note G00212454, Gartner Publications* pp. 1-32, May 2011, available online at http://jameskaskade.com/wp-content/uploads/2011/06/Magic-Quadrant-for-Security-Information-and-Event-Management.pdf, last accessed online June 25, 2015

[17] Pérez-Castillo R., Weber B., García-Rodríguez de Guzmán I., Piattini M., Pinggera J. (2014)., "Accessing event correlation in non-process aware information systems", *Springer Series on Systems and Software Modeling*, Vol. 13(3), pp. 1117-1139., July 2014.

[18] SC Awards 2015, SC Magazine Security Awards 2015, available online at http://media.scmagazine.com/documents/118/botn2015sm_29485.pdf, last accessed June 25, 2015

[19] Sheyner O., and Wing J. (2004). "Tools for Generating and Analyzing Attack Graphs", *Lecture Notes on Computer Science, Formal Methods on Components & Objects,* Vol. 3188, pp. 344-371.

[20] Zimmer. C, Bhat B., Mueller F., and Mohan, S., (2010)., "Time-based intrusion detection of cyber-physical systems", *In proceedings of the First ACM/IEEE Conference on Cyber-Physical Systems* (ICCPS 2010), ISBN: 978-1-4503-0066-7, ACM Publications, pp. 109-118.

[21] CERT Australia, "Cyber Crime & Security Survey Report" (2013)., *CERT Australia, Australian Govt.,* pp.1-51.

[22] ICS-CERT Monitor, Sept 2014-Feb 2015, *NCCIC, Dept. of Homeland Security,* pp.1-15.