

# Modeling and Quantification of Security Attributes of Software Systems\*

Bharat B. Madan<sup>1†</sup>, Katerina Goševa-Popstojanova<sup>2</sup>, Kalyanaraman Vaidyanathan<sup>1</sup>  
and Kishor S. Trivedi<sup>1</sup>

<sup>1</sup>*Department of Electrical and Computer Engineering  
Duke University, Durham, NC 27708  
{bbm,kv,kst}@ee.duke.edu*

<sup>2</sup>*Lane Department of Computer Science and Electrical Engineering  
West Virginia University, Morgantown, WV 26506  
katerina@csee.wvu.edu*

## Abstract

*Quite often failures in network based services and server systems may not be accidental, but rather caused by deliberate security intrusions. We would like such systems to either completely preclude the possibility of a security intrusion or design them to be robust enough to continue functioning despite security attacks. Not only is it important to prevent or tolerate security intrusions, it is equally important to treat security as a QoS attribute at par with, if not more important than other QoS attributes such as availability and performance. This paper deals with various issues related to quantifying the security attribute of an intrusion tolerant system, such as the SITAR system. A security intrusion and the response of an intrusion tolerant system to the attack is modeled as a random process. This facilitates the use of stochastic modeling techniques to capture the attacker behavior as well as the system's response to a security intrusion. This model is used to analyze and quantify the security attributes of the system. The security quantification analysis is first carried out for steady-state behavior leading to measures like steady-state availability. By transforming this model to a model with absorbing states, we compute a security measure called the "mean time (or effort) to security failure" and also compute probabilities of security failure due to violations of different security attributes.*

---

\*This work is sponsored by the U.S. Department of Defense Advanced Research Projects Agency (DARPA) under contract N66001-00-C-8057 from the Space and Naval Warfare Systems Center - San Diego (SPAWARSYSCEN). The views, opinions and findings contained in this paper are those of the authors and should not be construed as official DARPA or SPAWARSYSCEN's positions, policy or decision.

<sup>†</sup>Contact author

## 1. Introduction

It is imperative for well designed software systems to meet certain Quality-of-Service (QoS) requirements, such as reliability, availability and performability. Increasingly, such systems are being put to use in mission critical applications in military, aerospace, e-commerce, governmental and health care applications. At the same time, most such software systems are network accessible through public networks, such as the Internet. As a result, these applications and systems have become prone to security intrusions. The range of security intrusions may vary from minor mischief for pleasure, denial of service, and criminal intent for stealing or destroying assets controlled by such systems. This has brought security attribute of a software to the forefront of QoS specifications. As is the case with other common QoS measures, (reliability, availability etc.), qualitative evaluation of security attributes may no longer be acceptable. Instead, we need to quantify security so that a software system may be able to meet contracted levels of security.

### 1.1. Related work

As previously stated, the security of computing and information systems has been mostly assessed from a qualitative point of view. A system is assigned a given security level with respect to the presence or absence of certain functional characteristics and the use of certain development techniques. Only a few studies have considered the quantitative assessment of security. A discussion of the similarities between reliability and security with the intention of working towards measures of operational security appeared

in [12]. This paper identified open questions that need to be answered before the quantitative approach can be taken further. Work also exists on modeling the known Unix security vulnerabilities as a privilege graph [16]. The privilege graph is transformed into a Markov chain based on the assumption that the probability of success in a given elementary attack before an amount of effort  $e$  is spent is described by an exponential distribution  $P(e) = 1 - \exp(-\lambda e)$ , where  $1/\lambda$  is the mean effort to succeed in a given elementary attack. This model allows the evaluation of the proposed measure of operational security *mean effort to security failure*, analogous to mean time to failure. A quantitative analysis of attacker behavior based on empirical data collected from intrusion experiments was presented in [9]. A typical attacker behavior comprises of three phases: the learning phase, the standard attack phase, and the innovative attack phase. The probability for successful attacks, although for different reasons, is small during the learning and the innovative phases. During the standard attack phase the probability of successful attacks is considerably higher; the collected data indicated that the time between breaches in this phase are exponentially distributed.

In this paper we propose a model for quantitative assessment of security attributes for intrusion tolerant systems based on stochastic models. This is a generic model that considers intrusions with different impacts (e.g., compromise of confidentiality, compromise of data integrity, and denial of service attacks) and captures the dynamic behavior of the system in response to these intrusions.

## 1.2. Intrusion tolerance versus fault tolerance

In some ways, intrusion tolerance is similar to fault tolerance. However, despite some similarities, there are also a few differences as enumerated below:

- Hardware or software failures experienced by a system are almost invariably accidental in nature caused either by physical wear and tear, environmental conditions or by a peculiar set of inputs/excitations given to the system. In contrast, security intrusions are caused by deliberate user actions. It is however quite possible that a security intrusion may manifest itself as a failure. For example, stack overflow may either crash a system resulting in denial of service or it may be used to invoke a piece of hostile code.
- As mentioned in the previous point, there is an active attacker who causes a security intrusion unlike a failure that occurs accidentally. As a result, an attacker has to spend time and effort in order to be able to cause a security intrusion. In general, these attacks could arrive at a random point in time, just as a failure may occur randomly. In either case, this randomness can

be described by suitable arrival processes (e.g., Poisson, MMPP, NHPP etc.) [18]. Similarly, the amount of time or effort that an attacker has to spend in injecting an attack can be modeled as a random variable that can be described by chosen distribution functions.

- Before injecting an attack into a system, the attacker has to actively identify vulnerabilities present in the system that could be exploited to subsequently cause a security intrusion. This contrasts with the fault tolerance situation in which a system is always assumed to be vulnerable to failures. Intermittent and latent faults are exceptions to this.
- Once a system has been subjected to a security attack, an intrusion tolerant system responds to this security threat in a manner similar to the actions initiated by a fault tolerant system, though the exact details of such actions will vary. This similarity allows us to adopt some of the well established stochastic modeling and analysis techniques (e.g., Markov chains, semi-Markov processes etc.) [18] that have been extensively used in the field of fault tolerance for modeling and analyzing the security intrusion behavior of a system.

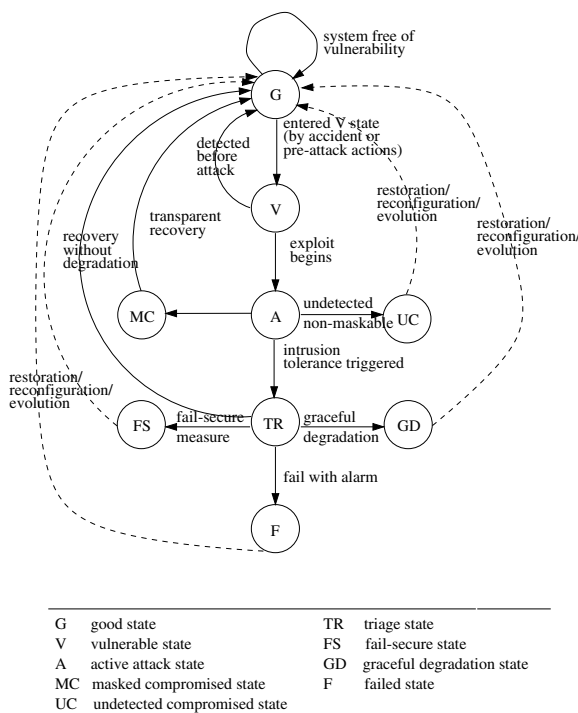
Based on the above discussion, in this paper, we use the state transition model of the SITAR intrusion tolerant system described in [8]. From the security quantification point of view, since some of the sojourn time distribution functions may be non-exponential, the underlying stochastic model needs to be formulated in terms of a semi-Markov process (SMP). Next, we analyze this SMP model to compute the following quantities for the purposes of quantifying the security measures. After computing steady-state probabilities of all the states, we can compute the steady-state availabilities. By making system failure states as *absorbing states* [18], the effort or time required to reach such absorbing states is computed to yield the *MTTSF* in a manner similar to the notion of MTTF. Computing the eventual probabilities of reaching each of the absorbing states, we can separate out the causes of different types of security violations.

The rest of the paper is organized as follows. In Section 2, we develop a semi-Markov model for an intrusion tolerant system like the SITAR system [20] from the security quantification view point. This model is used to find the steady-state probabilities which lead to the computation of the system availability. *MTTSF* analysis and the eventual probabilities of absorption are then described. Numerical results of the analysis performed on the models are presented in Section 5. Final conclusions are presented in Section 6 along with some future directions pertaining to estimating the parameters of the models used in this paper.

## 2. SMP Model for Security Quantification

A software system that is security intrusion tolerant has to be capable of reorganizing itself, preferably automatically, in order to mitigate the effects of a security intrusion. To analyze and quantify the security attributes of such a system, we have to consider not only the system's response to a security attack, but also the actions taken by an attacker to cause such an attack. This would require a composite security model that incorporates the behavior of both these elements.

### 2.1. Generic state transition model



**Figure 1. A state transition diagram for intrusion tolerant system**

Figure 1 depicts the state transition model which we proposed in [8] as a framework for describing dynamic behavior of an intrusion tolerant system. This is a generic model that enables multiple intrusion tolerance strategies to exist and supports tolerance of intrusions with different impacts (e.g, compromise of confidentiality, compromise of data integrity, and denial of service attacks). Case studies that map several known vulnerabilities to this model are presented in [8]. Here, we briefly describe the basic concepts.

Traditional computer security leads to the design of systems that rely on resistance to attacks, that is, hardening

for protection using strategies such as authentication, access control, encryption, firewalls, proxy servers, strong configuration management, and upgrades for known vulnerabilities. If the strategies for resistance fail, the system is brought from good state  $G$  into the vulnerable state  $V$  during the penetration and exploration phases of an attack. If the vulnerability is exploited successfully, the system enters the active attack state  $A$  and damage may follow. Intrusion tolerance picks up where intrusion resistance leaves off. The four phases that form the basis for all fault tolerance techniques are error detection, damage assessment, error recovery, and fault treatment [11]. These can and should be the basis for the design and implementation of an intrusion tolerant system.

Strategies for detecting attacks and assessment of damage include intrusion detection (i.e., anomaly based and signature based detection), logging, and auditing. If the probing that precedes the attack is detected, the system will stay in the good state. The other possibility is to detect the penetration and exploration phases of an attack and bring the system from the vulnerable state back to the good state. So far, the resistance and detection of attacks have received most of the attention, and once active attack state is entered damage may follow with little to stand in the way. Therefore, it is critical to use strategies for recovery which include the use of redundancy for critical information and services, incorporation of backup systems in isolation from network, isolation of damage, ability to operate with reduced services or reduced user community.

The best possible case is when there is enough redundancy to enable the delivery of error-free service and bring the system back to the good state by masking the attack's impact ( $MC$ ). The worst possible case is when the intrusion tolerance strategies fail to recognize the active attack state and limit the damage, leading to an undetected compromised state  $UC$ , without any service assurance.

When an active attack in exploitation phase is detected, the system will enter the triage state  $TR$  attempting to recover or limit the damage. Ideally, of course, the system should have in place some measures for eliminating the impacts produced by an attack, providing successful restoration to the good state. However, when restoration to the good state is not feasible, the system could attempt to limit the extent of damage while maintaining the essential services. Essential services are defined as the functions of the system that must be maintained to meet the system requirements even when the environment is hostile, or when failures or accidents occur that threaten the system [7]. Of course, there is no "one size fits all" solution. In intrusion tolerance the impacts are more important than the causes. If the aim is to protect the system from denial of service (DoS) attack, the system should enter the graceful degradation state  $GD$ , maintaining only essential services. How-

ever, if the aim is to protect confidentiality or data integrity the system must be made to stop functioning. This is called fail-secure state  $FS$ , analogous to fail-safe state in fault tolerance. If all of the above strategies fail then the system enters the failed state,  $F$ , and signals an alarm.

Recovering the full services after an attack and returning to the good state by manual intervention is represented by transitions denoted with dashed lines. In order to reduce the effectiveness of future attacks it may be required to use techniques such as reconfiguration or evolution of the system. This phase can be considered analogous to fault treatment phase in fault tolerance.

Next, we develop the stochastic model of intrusion tolerant systems appreciating that the uncertainty in security arises from the incompleteness of our knowledge. To an attacker with an incomplete knowledge of the system, there is uncertainty as to the effects of the attack. To the system designer/owner/operator, there is uncertainty as to the type, frequency, intensity and the duration of the attack, and even as to whether a particular attack would result in a security breach. In developing such a theory we need to describe the events that trigger transitions among states in terms of probabilities and cumulative distribution functions (CDF).

## 2.2. Attacker's behavior and system's response

In order to analyze the security attributes of an intrusion tolerant system, we need to consider the actions undertaken by an attacker as well as the system's response to an attack. An attacker always tries to eventually send such a system into a security-failed state. Obviously, this requires the attacker to spend time or effort. In general, this time or effort<sup>1</sup> is best modeled as a random variable. Depending on the nature of an attack, this random variable may follow one of several distribution functions. In this paper, we borrow some of the common distribution functions used in the field of reliability theory. Deterministic, exponential, hyper-exponential, hypo-exponential, Weibull, gamma and log-logistic etc. are some of the distribution function that make sense in the context of security analysis [18]. The hypo-exponential distributions may be used to model transitions that may involve multi-stage activities. For example, the *Code-Red* worm [17] has to first cause the parameter stack buffer to overflow by sending a long URL to the web server that is to be attacked. In the next stage, this is followed by causing the normal return address (already stored on this stack) to be over-written with a bad return address placed in this URL. In the final stage, this bad return address points to a rogue piece of *Code-Red* code (also supplied as a part of the long URL) that gets invoked next time the return from a call is executed. Thus the above discussion suggests that we need to consider non-exponential type

<sup>1</sup>Henceforth, we will use time to represent both time and/or effort

of distributions. The hypo-exponential distribution may be used to model threat situations that can cause monotonically increasing failure rate (IFR) of security. Similarly, hyper-exponential distribution may be used to model threats that have can cause monotonically decreasing failure rate (DFR). Weibull distribution function may be used to model constant failure rate (CFR), DFR or IFR type of threats by suitably choosing its parameters. For more complex attack scenarios, that are characterized by having a decreasing rate of success initially, followed by an increasing rate of success (or vice-a-versa), we can use the log-logistic type of distribution function. It should also be noted that an attacker may not always be successful in causing a security failure, i.e., probability of success  $\leq 1$ . In relation to the state transition diagram described in Figure 1, an attackers actions are modeled by the states  $\{G, V, A\}$ .

An intrusion tolerant has to constantly evaluate the presence of any security penetration. Once a security attack is detected, the system needs to initiate suitable remedial actions. After detecting an attack, a SITAR like system can respond in a variety of ways. However, the basic nature of this response would be to try to make the system move back to a secure state from a security-compromised state. Obviously this movement will require time or effort on the part of the system. As before, this time or effort is best modeled as a random variable that is described by a suitable probability distribution function. It should again be remarked here that it is not guaranteed that the system will be able to detect all attacks, i.e., probability of detection of an attack is  $\leq 1$  in general. Thus system's response may be parameterized by a set of distribution functions and a set of probabilities. For a SITAR like system, the system's response to a security intrusion may be described by the states  $\{MC, UC, TR, FS, GD, F\}$  and the transitions between these states. A system's response to a security attack is fairly automated and could be quite similar to how it may respond to accidental faults. Let  $\{X(t) : t \geq 0\}$  be the underlying stochastic process with a discrete state space  $X_s = \{G, V, A, TR, MC, UC, FS, GD, F\}$ . To analyze an SMP, we need to deal with two sets of parameters [18, 5] - (i) mean sojourn time  $h_i$  in state  $i \in X_s$ , and (ii) the transition probabilities  $p_{ij}$  between different states  $i \in X_s$  and  $j \in X_s$ . We note that the analysis carried out in this paper depends only on the mean sojourn time and is independent of the actual sojourn time distributions for the SMP states. If we were to carry out a transient analysis of the SMP, this will no longer be true.

When analyzing security, we may also be interested in computing the calendar time it takes to cause such a transition. In such cases, we have to establish a relationship between effort and time. In general, effort is a random function of time since time required to cause a transition depends on several randomly behaving entities, e.g., attacker's

level of expertise and background, robustness of the system, type of pre-existing system vulnerabilities etc. This will result in a doubly-stochastic model. However, due to space limitations, we will not deal with the doubly stochastic model here. Instead, we will ignore the difference between time and effort and use these terms interchangeably.

### 3 Model Analysis

In this section we discuss and derive several security related quality attributes based on the SMP model presented in the previous section. Security related quality attributes are considered by some researchers as part of dependability [6]. Dependability is defined as the property of computer system such that reliance can justifiably be placed on the service it delivers [10]. Dependability attributes include:

- *Reliability*, continuity of service
- *Safety*, non-occurrence of catastrophic consequences
- *Maintainability*, ability to undergo repairs and evolutions.
- *Availability*, readiness for usage
- *Integrity*, data and programs are modified or destroyed only in a specified and authorized manner
- *Confidentiality*, sensitive information is not disclosed to unauthorized recipients

The present work is concerned primarily with evaluating the last three attributes. Associating integrity and availability with respect to authorized actions, together with confidentiality, leads to *security* [2]. The degree to which each of these properties is needed varies from application to application. For instance, the defense industry is primarily interested in confidentiality. In contrast, the banking industry is primarily interested in integrity, and the telephony industry may value availability most. The exact requirements that are needed for a particular system or application are expressed in the security policy for that system or application.

While the methods for quantitative assessment of dependability attributes such as reliability, availability, and safety are well established, so far the security attributes have been mostly assessed from the qualitative point of view. In this paper we derive and evaluate dependability attributes that are relevant to security.

*Instantaneous availability*  $\mathcal{A}(t)$  of a system is defined as the probability that the system is properly functioning at time  $t$ . We are interested in the *steady state availability*  $\mathcal{A}$  as the limiting value of  $\mathcal{A}(t)$  as  $t$  approaches infinity. For our model, the system is unavailable in states  $FS$ ,  $F$ , and  $UC$ , that is, the availability  $\mathcal{A}$  is given by  $\mathcal{A} = 1 - (\pi_{FS} + \pi_F + \pi_{UC})$ , where,  $\pi_i, i \in \{FS, F, UC\}$  denotes the steady state probability that the SMP is in state  $i$ . Note that for some applications and types of attacks the system may be considered available in the state  $UC$ .

Availability is an appropriate measure for the compro-

mise of data integrity and DoS attacks. It should be pointed out that in the case of DoS attacks which are aimed at disrupting normal services by consuming large amounts of service resources, states  $MC$  and  $FS$  do not make sense. Thus, it is not possible to mask DoS attack by using redundancy. Also, intentionally making the system to stop functioning, i.e., bringing it to the fail-secure state  $FS$  will accomplish the goal of DoS attack. Therefore, the states  $MC$  and  $FS$  will not be part of the state diagram describing DoS attacks. It follows that for the DoS attacks the system availability reduces to  $\mathcal{A}_{DoS} = 1 - (\pi_F + \pi_{UC})$ .

In a similar manner, confidentiality and integrity measures can be computed in the context of specific security attacks. For example, Microsoft IIS 4.0 suffered from the so-called ASP vulnerability as documented in the Bugtraq ID 167 [1]. Exploitation of this vulnerability allows an attacker to traverse the entire web server file system, thus compromising confidentiality. Therefore, in the context of this attack, states  $FS$  and  $F$  are identified with the loss of confidentiality. Similarly, if the well known *Codered* worm is modified to inject a piece of code into a vulnerable IIS server to browse unauthorized files, states  $FS$  and  $F$  will imply loss of confidentiality. Therefore, the confidentiality measure can then be computed as:  $\mathcal{C}_{ASP} = 1 - (\pi_F + \pi_{UC})$ .

The integrity measure in the presence of integrity compromising attacks can be computed in a similar manner. Take for example the CGI vulnerability present in the Sambar server as reported in the Bugtraq ID 1002 [1]. Exploitation of this vulnerability permits an attacker to execute any MS-DOS command including deletion and modification of files in an unauthorized manner, thus compromising the integrity of the system. Once again, states  $FS$  and  $F$  signal compromise of the integrity measure  $\mathcal{I}_{CGI}$  which can be computed as in equations for  $\mathcal{A}_{DoS}$  and  $\mathcal{C}_{ASP}$ .

Another measure of interest is the *Mean time to security failure* (MTTSF). For the purpose of deriving MTTSF, failed or compromised states are made absorbing by deleting all outgoing arcs from these states. For our model of intrusion tolerant system, states  $FS$  (if it applies),  $F$ ,  $UC$ , and  $GD$  are made absorbing states. Section 4 describes the details of the method used for computing the *MTTSF* for a generic model that can be specialized easily for specific security attacks.

#### 3.1. DTMC steady-state probability computations

It was explained earlier that to carry out the security quantification analysis, we need to analyze the SMP model of the system that was described by its state transition diagram. The SMP corresponding to Figure 1 can be described in terms of its embedded DTMC shown in Figure 2. As stated in Section 2 complete description of this SMP model requires the knowledge of various parameters, viz. mean

sojourn time in each state and the branching probabilities. Some of the parameters of the SMP model are summarized here:

- $h_G$ , Mean time for the system to resist becoming vulnerable to attacks
- $h_V$ , Mean time for the system to resist attacks when vulnerable
- $h_A$ , Mean time taken by the system to detect an attack and initiate triage actions
- $h_{MC}$ , Mean time the system keeps the effects of an attack masked
- $h_{UC}$ , Mean time that an attack remains undetected while doing the damage
- $h_{TR}$ , Mean time the system takes to evaluate how best to handle an attack
- $h_{FS}$ , Mean time the system operates in a fail secure mode in the presence of an attack
- $h_{GD}$ , Mean time the system is in the degraded state in the presence of an attack
- $h_F$ , Mean time the system is in the fail state despite detecting an attack
- $p_a$ , Prob. of injecting a successful attack, given that the system was vulnerable
- $p_u$ , Prob. that a successful attack has remained undetected
- $p_m$ , Prob. that the system successfully masks an attack
- $p_g$ , Prob. that the system resists an attack by gracefully degradation
- $p_s$ , Prob. that the system responds to an attack in a fail secure manner

Clearly for the model to be accurate, it is important to estimate the model parameters accurately. In this paper, our focus is more on developing a methodology for analyzing quantitatively the security attributes of an intrusion tolerant system rather than model parameterization. In Section 6, we briefly discuss methods that may be used to estimate these parameters and present the results of this study in a future paper. In the absence of exact values of model parameters, it will, however be meaningful to evaluate the sensitivity of security related attributes to variations in model parameters. In Section 5 we present some numerical results to evaluate the sensitivity of the *MTTSF* and the steady-state availability *A* to various model parameters.

In order to compute availability measure, we need to first compute the steady-state probabilities  $\{\pi_i, i \in X_s\}$  of the SMP states.  $\pi_i$ 's in turn can be computed in terms of the embedded DTMC steady-state probabilities  $\nu_i$ 's and the mean sojourn times  $h_i$ 's [18]:

$$\pi_i = \frac{\nu_i h_i}{\sum_j \nu_j h_j}, \quad i, j \in X_s. \quad (1)$$

The DTMC steady-state probabilities  $\nu_i$ 's can be computed

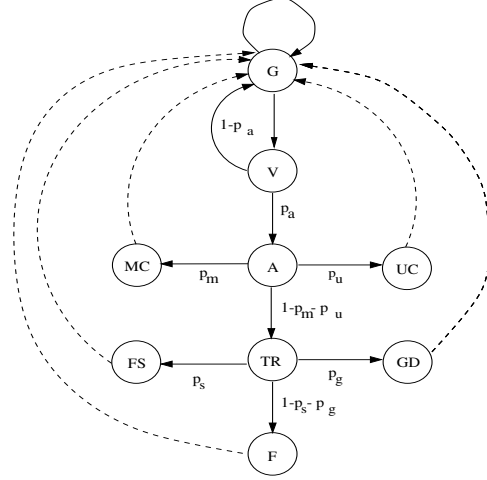


Figure 2. Embedded DTMC for the SMP model

as,

$$\bar{\nu} = \bar{\nu} \cdot P \quad (2)$$

where,  $\bar{\nu} = [\nu_G \ \nu_V \ \nu_A \ \nu_{MC} \ \nu_{UC} \ \nu_{TR} \ \nu_{FS} \ \nu_{GD} \ \nu_F]$  and  $P$  is the DTMC transition probability matrix which can be written as:  $P =$

$$\begin{array}{l} G \\ V \\ A \\ MC \\ UC \\ TR \\ FS \\ GD \\ F \end{array} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ p_1 & 0 & p_a & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_m & p_u & p_2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_s & p_g & p_3 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where,  $p_1 = 1 - p_a$ ,  $p_2 = 1 - p_m - p_u$  and  $p_3 = 1 - p_s - p_g$ . In addition,

$$\sum_i \nu_i = 1, \quad i \in \{G, V, A, MC, UC, TR, FS, GD, F\}. \quad (3)$$

The  $P$  matrix describes the DTMC state transition probabilities that are used to label the transitions between the DTMC states as shown in Figure 2. Knowledge of these transition probabilities would be essential to completely analyze the SMP security model. Section 6 briefly touches upon the issue of how to estimate various model parameters based on an a-priori knowledge and intrusion injection experiments as suggested in [13, 14]. Our focus in this paper, however, is mostly on developing security analysis techniques, given various model parameters i.e., mean sojourn times and the DTMC transition probabilities. Towards this end, the first component of security analysis requires us to find DTMC steady-state probabilities. We can derive expressions for  $\nu$ 's by solving equations (2) and (3).

### 3.2. Semi-Markov model analysis

The mean sojourn time  $h_i$  in a particular state  $i$  is the other quantity that is needed to compute  $\pi_i$ 's.  $h_i$  obviously is determined by the random time that a process spends in a particular state. In the computer security domain, there is a wide variety of attackers ranging from amateur hackers to cyber criminal syndicates to inimical intelligence agencies possessing a wide spectrum of expertise and resources. DARPA has recently initiated the Information Assurance (IA) program [13, 14] that aims to characterize a wide range of attacks. While many more studies need to be carried out to get a more complete understanding of the attacker behavior, DRAPA's IA studies point to the fact that in order to capture the attacker behavior, we need to consider a variety of attacks ranging from trivial to highly sophisticated. In the model being considered in this paper, the transitions  $G \rightarrow V$  and  $V \rightarrow A$  describe the attacker's behavior. Keeping in mind a wide range of attacks, we need to consider a variety of probability distribution functions describing attacker related transitions. A system's response on the other hand, is more algorithmic and automated that is not very different from how a system responds to conventional failures due to accidental faults. The important advantage of the approach developed in this paper to analyze and quantify security lies in its simplicity. Starting with the SMP model used to capture the security related behavior of a system, we can derive the embedded DTMC that involves only the transition probabilities. Given this DTMC model, the steady-state DTMC probabilities  $\nu_i$ 's can be easily computed as shown in the previous Subsection. Therefore, it suffices to know just the mean sojourn times  $h_i$ 's, in order to compute SMP steady-state probabilities  $\pi_i$ 's. As an example, if we assume the sojourn time distributions for two of the states, viz.  $G$  and  $V$  to be HypoEXP( $\lambda_{g1}, \lambda_{g2}$ ) and Weibull( $\lambda_v, \alpha_v$ ) respectively, then  $h_G = (1/\lambda_{g1} + 1/\lambda_{g2})$  and  $h_V = (1/\lambda_v)^{1/\alpha_v} \Gamma(1 + 1/\alpha_v)$ . Similarly, remaining states  $\{A, MC, UC, TR, FS, GD, F\}$  have mean sojourn times  $\{h_A, h_{MC}, h_{GD}, h_{TR}, h_{FS}, h_{UC}, h_F\}$ , respectively. The SMP steady-state probabilities  $\pi_i$ 's can now be easily computed by using equations (1), (2) and (3) as:

$$\begin{aligned}\pi_G &= (h_G)(h_G + h_V + p_a[h_A + p_m h_{MC} + p_u h_{GD} \\ &\quad + (1 - p_m - p_u)[h_{TR} + p_s h_{FS} + p_g h_{UC}] \\ &\quad + (1 - p_g - p_s)h_F])^{-1} \\ \pi_V &= \frac{h_V \pi_G}{h_G}; \quad \pi_A = p_a \frac{h_A \pi_G}{h_G}; \\ \pi_{UC} &= p_a p_u \frac{h_{GD} \pi_G}{h_G} \\ \pi_{MC} &= p_a p_s (1 - p_m - p_u) \frac{h_{FS} \pi_G}{h_G} \\ \pi_{FS} &= p_a p_u \frac{h_{MC} \pi_G}{h_G}\end{aligned}$$

$$\begin{aligned}\pi_{TR} &= p_a p_s (1 - p_m - p_u) \frac{h_{TR} \pi_G}{h_G} \\ \pi_{GD} &= p_a p_g (1 - p_m - p_u) \frac{h_{UC} \pi_G}{h_G} \\ \pi_F &= p_a (1 - p_s - p_g) (1 - p_m - p_u) \frac{h_F \pi_G}{h_G}. \quad (4)\end{aligned}$$

Given the steady-state probabilities, various measures, such as, availability, confidentiality and integrity may be computed via equations for  $\mathcal{A}_{DoS}$  or  $\mathcal{C}_{ASP}$ .

#### 3.2.1 Model of a SYN-flood DoS attack

A significant advantage of the SMP model described so far is its generic nature that is easy to specialize for specific security attacks. For example, when a system is being subjected to a SYN-flood DoS attack, the model reduces to states  $(G, V, A, UC, TR, GD, F)$ . The resulting SMP with reduced number of states is as shown in Figure 3. Solution of this SMP based on the approach outlined earlier yields the following steady-state probabilities,  $\pi_i$ 's, as a special case of (4).

$$\begin{aligned}\pi_G &= (h_G)(h_G + h_V + p_a[h_A + p_u h_{GD} + (1 - p_u) \\ &\quad h_{TR} + p_g (1 - p_u) h_{UC} + (1 - p_u)(1 - p_g) h_F])^{-1} \\ \pi_V &= \frac{h_V \pi_G}{h_G}; \quad \pi_A = p_a \frac{h_A \pi_G}{h_G} \\ \pi_{UC} &= p_a p_u \frac{h_{GD} \pi_G}{h_G}; \quad \pi_{TR} = p_a p_u \frac{h_{TR} \pi_G}{h_G}; \\ \pi_{GD} &= p_a p_g (1 - p_u) \frac{h_{UC} \pi_G}{h_G}; \\ \pi_F &= p_a (1 - p_g) (1 - p_u) \frac{h_F \pi_G}{h_G}. \quad (5)\end{aligned}$$

In the context of a DoS attack, availability is the only meaningful security attribute that can be computed using the equation for  $\mathcal{A}_{DoS}$ .

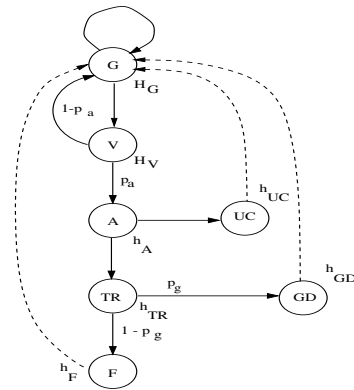


Figure 3. DoS attack - SMP model

#### 4. MTTSF analysis

For quantifying the reliability of a software system, *Mean time to failure* (MTTF) is a commonly used reliability measure. MTTF provides the mean time it takes for the system to reach one of the designated failure states, given that the system starts in a good or working state. The failed states are made absorbing states. Using the MTTF analogy, we define *Mean time to security failure* (MTTSF) as the measure for quantifying the security of an intrusion tolerant system. MTTF or MTTSF can be evaluated by making those states of the embedded DTMC that are deemed to be failed or security-compromised states as the absorbing states. Classification of the SMP states into absorbing and transient categories depends on the actual nature of the security intrusion. For example, if the model is describing the *Sun web server bulletin board vulnerability (Bugtraq ID 1600)* [1], the states  $X_a = \{UC, FS, GD, F\}$  will form the set of absorbing states, while  $X_t = \{G, V, A, MC, TR\}$  will be the set of transient states. In contrast, for the SYN-flood DoS attack,  $X_a = \{UC, GD, F\}$  and  $X_t = \{G, V, A, TR\}$ . It is clear that once the system reaches one of the absorbing states, the probability of moving out of such a state is 0, i.e., outgoing arcs from such states are deleted. The resulting transition probability matrix  $P$  then has the general form,

$$P = \left[ \begin{array}{c|c} Q & C \\ \hline 0 & I \end{array} \right]$$

submatrices  $Q$  and  $C$  consists of the transition probabilities between transient states and from transient states to absorbing states respectively. Matrices  $Q$  and  $C$  are given by,

$$Q = \begin{array}{c} \begin{array}{ccccc} & G & V & A & MC & TR \\ \begin{array}{c} G \\ V \\ A \\ MC \\ TR \end{array} & \left[ \begin{array}{ccccc} 0 & 1 & 0 & 0 & 0 \\ (1-p_a) & 0 & p_a & 0 & 0 \\ 0 & 0 & 0 & p_m & (1-p_m-p_u) \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array} \right] \end{array} \\ \\ C = \begin{array}{c} \begin{array}{ccccc} & UC & FS & GD & F \\ \begin{array}{c} G \\ V \\ A \\ MC \\ TR \end{array} & \left[ \begin{array}{ccccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ p_u & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & p_s & p_g & 1-p_s-p_g \end{array} \right] \end{array} \end{array}$$

To find the MTTSF we can now use the approach outlined in [18, 3]. Using this approach,

$$\text{MTTSF} = \sum_{i \in X_t} V_i h_i \quad (6)$$

where  $V_i$  denotes the the average number of times the state  $i \in X_t$  is visited before the DTMC reaches one of the absorbing states and  $h_i$  is the mean sojourn time in state  $i$ . The visit count elements  $V_i$  can be written as,

$$V_i = q_i + \sum_j V_j Q_{ji}, \quad i, j \in X_t \quad (7)$$

where,  $q_i$  is the probability that the DTMC starts in state  $i$ . In our case, we assume that  $G$  is the initial state. This gives,

$$\bar{q} = [q_i] = [1 \ 0 \ 0 \ 0 \ 0].$$

Solving (7) for the visit counts  $V_i$ 's gives,

$$\begin{aligned} V_G &= \frac{1}{p_a(1-p_m)} & V_V &= V_G & V_A &= \frac{1}{1-p_m} \\ V_{MC} &= \frac{p_m}{1-p_m} & V_{TR} &= \frac{1-p_m-p_u}{1-p_m}. \end{aligned}$$

With the knowledge of the mean sojourn times  $h_i$ 's in various states  $\{i \in X_t\}$ , we can use (6) to compute the MTTSF as,  $\text{MTTSF} = [h_G p_a + h_V p_a + h_A + p_m h_{MC} + (1-p_m) h_{TR}] [1-p_m]^{-1}$ . In the next Section, we choose specific parameters for our SMP model that will allow us to compute some numerical results.

When a system fails in the context of security (on an average, after MTTSF interval of time from the start time has elapsed) the system will find itself in one of the absorbing states. For example, for the Bugtraq 1600, this state will  $\in \{UC, FS, GD, F\}$ . Any security intrusion can have many security implication. Depending on the actual code inserted by intruder by exploiting the Bugtraq 1600 vulnerability, the intrusion may result in the compromise of user authentication and confidentiality in case the system finds itself in the  $UC$  state. Alternately, if the system reaches the  $FS$  or  $F$ , it may imply non-availability of some or all services. It is therefore important to be determine the final absorption state in probabilistic terms. In computational terms, this would require finding the SMP probabilities for the states  $\in X_a$  after absorption. We now define a matrix  $B = [b_{ij}]$ , where,  $b_{ij}$  denotes the probability that the DTMC starting in a state  $\{i \in X_t\}$  will eventually get absorbed in an absorbing state  $\{j \in X_a\}$ . It has been shown earlier in [15] that,  $B = (I - Q)^{-1}C$ . The first row elements of  $B$  can then be written as,

$$b_{1j} = \sum_{i \in X_t} V_i C_{ij} \quad j \in X_a$$

Therefore,

$$\begin{aligned} b_{1F} &= \frac{(1-p_s-p_g)(1-p_m-p_u)}{1-p_m} \\ b_{1FS} &= \frac{p_s(1-p_m-p_u)}{1-p_m} \\ b_{1GD} &= \frac{p_g(1-p_m-p_u)}{1-p_m}; & b_{1UC} &= \frac{p_u}{1-p_m}. \end{aligned}$$



In other words, once a security attack succeeds in causing a security failure, elements  $\{b_{1j}, j \in X_a\}$  give us the probability that after system has failed, it would reach the absorbing state  $j$  given that the system started in the  $G$  state.

## 5. Numerical Results

One can also obtain a numerical solution to the DTMC and the SMP described in Figure 2. We use the following set of model parameters:

### Transition probabilities:

We assume that a successful injection of an attack is less likely as compared to an unsuccessful injection of attack. The probability of injecting a successful attack from the vulnerable state,  $p_a = 0.4$ . The probability that the system can successfully mask the attack inherently,  $p_m = 0.3$ . The probability of an undetected attack,  $p_u = 0.2$ . Hence the probability of attack detection (the system enters the triage state),  $1 - p_m - p_u = 0.5$ . The probability that the system can resist an attack by gracefully degrading itself,  $p_g = 0.6$  and the probability that the system enters the fail-secure state,  $p_s = 0.3$ . Hence the probability of an attack going undetected from the triage state,  $1 - p_g - p_s = 0.1$ .

### Mean sojourn times:

We assume that the time spent in the attack state,  $A$ , is less than the times spent in each of the states,  $V$  and  $G$ . The mean time spent in the good state ( $G$ ),  $h_g = 1/2$  time units and the mean time spent in the vulnerable state ( $V$ ),  $h_v = 1/3$  time units. The mean time the attacker spends in the attack state ( $A$ ),  $h_A = 0.25$  time units. The mean time the system masks an attack before it is brought back to the good state,  $h_{MC} = 0.25$  time units. The mean time the attack remains undetected, until a manual intervention to bring it back to the good state,  $h_{UC} = 0.5$  time units. In the triage state, the mean time spent by the system,  $h_{TR} = 1/6$  time units. Once an attack is detected and if the system enters the fail-secure mode, the mean time the system spends in this state,  $h_{FS} = 1$  time units. In case the system enters the graceful degradation mode, the system spends a mean time,  $h_{GD} = 4$  time units. The mean time spent in the failed and attack detected state,  $h_F = 2$  time units.

### • DTMC steady-state probabilities:

The steady-state DTMC probabilities are:

$$\begin{aligned} \nu_G &= 0.3333; & \nu_V &= 0.3333; & \nu_A &= 0.1333; \\ \nu_{MC} &= 0.04; & \nu_{UC} &= 0.0267; & \nu_{TR} &= 0.06667; \\ \nu_{FS} &= 0.02; & \nu_{GD} &= 0.04; & \nu_F &= 0.0067. \end{aligned}$$

Assuming states  $FS$ ,  $F$  and  $UC$  are the unavailable states, steady-state availability = 0.9466

### • SMP steady-state probabilities:

$$\begin{aligned} \pi_G &= 0.3386; & \pi_V &= 0.2257; & \pi_A &= 0.0677; \\ \pi_{MC} &= 0.0203; & \pi_{UC} &= 0.2167; & \pi_{TR} &= 0.0226; \\ \pi_{FS} &= 0.0406; & \pi_{GD} &= 0.0406; & \pi_F &= 0.0271. \end{aligned}$$

Assuming states  $FS$ ,  $F$  and  $UC$  are the unavailable states, steady-state availability  $\mathcal{A} = 0.7156$ .

### • MTTSF: MTTSF = 3.5595 time units.

### • Probability of eventual absorption:

$$\begin{aligned} b_{1F} &= 0.071429; & b_{1FS} &= 0.21429; \\ b_{1GD} &= 0.42857; & b_{1UC} &= 0.2857. \end{aligned}$$

### • Sensitivity Analysis:

Sensitivity analysis is often performed on models so that the system can be optimized, parts of the system model sensitive to error can be identified and bottlenecks in the system can be found [4]. We perform parametric sensitivity analysis on the SMP model and examine the sensitivity of the availability and the MTTSF. We first compute the derivative of a measure,  $M$ , with respect to various system parameters,  $\lambda_i$ . Performing more detailed analysis or taking additional measurements in a system involves cost or time. This additional cost or time due to the change in  $\Delta\lambda_i$  could be assumed to be proportional to  $\Delta\lambda_i/\lambda_i$ . Let  $I_i = \left| \lambda_i \frac{\partial M}{\partial \lambda_i} \right|$ . Refining the parameter  $\lambda_i$  that results in the maximum value of  $I_i$  is the most cost effective way to improve the accuracy of the model.

### MTTSF Sensitivity

$$\begin{aligned} I_{h_G} &= 0.3333 & I_{h_V} &= 0.2222 & I_{h_A} &= 0.3571 \\ I_{h_{MC}} &= 0.1071 & I_{h_{TR}} &= 0.1667 & I_{p_a} &= 0.4762 \\ I_{p_m} &= 0.1531. \end{aligned}$$

These numerical results suggest that the  $MTTSF$  is sensitive to various model parameters in the following order of decreasing sensitivity -  $\{p_a, h_A, h_G, h_V, h_{TR}, p_m, h_{MC}\}$ .

### Availability Sensitivity

$$\begin{aligned} I_{h_G} &= 0.2844 & I_{h_{GD}} &= 0.2167 & I_{h_{FS}} &= 0.0406 \\ I_{h_F} &= 0.0271 & I_{p_a} &= 0.2844 & I_{p_u} &= 0.1896 \\ I_{p_m} &= 0.0406 & I_{p_s} &= 0.0406 & I_{p_g} &= 0.1625. \end{aligned}$$

From the above numerical results we can infer that the *Availability's* sensitivity to various model parameters exhibits the following order of decreasing sensitivity -  $\{(h_G, p_a), h_{GD}, p_u, p_g, (h_{FS}, p_m, p_s), h_F\}$ .

## 6. Conclusions and Future Work

In this paper we have presented an approach for quantitative assessment of security attributes for an intrusion tolerant system. A state transition model that describes the dynamic behavior of such a system is used as a basis for developing a stochastic model. This is a generic model that enables the study of a variety of intrusion tolerance strategies as well as assess different impacts of a security attack. Since the memoryless property of exponential distribution implies the absence of aging and learning, it does not seem appropriate for modeling attacker behavior. In this paper, we have identified several general probability distribution functions that can be used to describe the attacker behavior and solved the semi-Markov process for several security related attributes. These include the steady-state availability and the mean time to security failure. Also, by differentiating between various absorbing states, we have computed the probability of security failure due to violations of different security attributes. The model analysis is illustrated in a numerical example.

One of the goals of our future work is to design and conduct experiments based on the recent experiences of the DARPA Information Assurance (IA) program [13]. These experiments should provide us with a better understanding of the behavior exhibited by attackers, help us to refine its stochastic description and lead to better estimates of the model parameters. As a part of the ongoing SITAR project, we plan to conduct semi-automated and automated experiments. Putting a human attacker team (Red Team) against a set of system's autonomic defenses is an example of semi-automated experiment. Fault injection [19], the well-known technique for testing and validating fault tolerant systems, is one of the techniques that provide a capability of automating the experimentation. In this paper, the absence of exact value of model parameters is sought to be addressed instead by studying the sensitivity of different security attributes to small changes in the parameter values.

Another goal of our future research is to consider quality attributes such as performance, performability, and survivability in addition to the security attributes studied in this paper. The analysis of multiple quality attributes and their tradeoffs will yield insights into system's strengths and weaknesses and provide basis for carrying out cost/benefit analysis.

## References

- [1] Bugtraq archive. <http://www.securityfocus.com>.
- [2] In *Validation Framework Workshop discussions, DARPA OASIS Pls Mtg., Hilton Head, SC, March 11-15, 2002*.
- [3] U. Bhat. *Elements of Stochastic Processes*. 2Ed., John Wiley, New York, 1984.
- [4] J. T. Blake, A. L. Reibman, and K. S. Trivedi. Sensitivity analysis of reliability and performability measures for multiprocessor systems. *Proc. of ACM SIGMETRICS*, pages 177–186, 1988.
- [5] D. Cox and H. Miller. *The Theory of Stochastic Processes*. Chapman and Hall, 1990.
- [6] J. Dobson, J. Laprie, and B. Randell. Predictably dependable computing systems. *Bulletin of the European Association for Theoretical Computer Science*, 1990.
- [7] R. J. Ellison et al. Survivability: Protecting your critical systems. *IEEE Internet Computing*, 1999.
- [8] K. Goševa-Popstojanova, F. Wang, R. Wang, F. Gong, K. Vaidyanathan, K. Trivedi, and B. Muthusamy. Characterizing intrusion tolerant systems using a state transition model. In *DARPA Information Survivability Conference and Exposition (DISCEX II)*, volume 2, pages 211–221, 2001.
- [9] E. Jonsson and T. Olovsson. A quantitative model of the security intrusion process based on attacker behavior. *IEEE Trans. Software Eng.*, 23(4):235, April 1997.
- [10] J. C. Laprie. Dependability of computer systems: Concepts, limits, improvements. In *Proc. of the ISSRE-95*, pages 2–11, 1995.
- [11] P. A. Lee and T. Anderson. *Fault Tolerance: Principles and Practice*. Springer Verlag, 1990.
- [12] B. Littlewood, S. Brocklehurst, N. Fenton, P. Mellor, S. Page, and D. Wright. Towards operational measures of computer security. *Journal of Computer Security*, 2:211–229, 1993.
- [13] J. Lowry. An Initial Foray into Understanding Adversary Planning and Courses of Action. In *DARPA Information Survivability Conference and Exposition (DISCEX II)*, volume 1, pages 123–133, 2001.
- [14] J. Lowry and K. Theriault. Experimentation in the IA Program. In *DARPA Information Survivability Conference and Exposition (DISCEX II)*, volume 1, pages 134–140, 2001.
- [15] J. Medhi. *Stochastic Processes*. Wiley Eastern, New Delhi, 1994.
- [16] R. Ortalo et al. Experiments with quantitative evaluation tools for monitoring operational security. *IEEE Trans. Software Eng.*, 25(5):633–650, Sept/Oct 1999.
- [17] P. Mellor, S. Page, and D. Wright. Code red worm. <http://www.sarc.com/avcenter/venc/data/codered.worm.html>, 2001.
- [18] K. S. Trivedi. *Probability and Statistics with Reliability, Queuing, and Computer Science Applications (2nd ed.)*. John Wiley & Sons, 2001.
- [19] J. M. Voas and A. K. Ghosh. Software fault injection for survivability. In *DARPA Information Survivability Conference and Exposition (DISCEX'00)*, volume 2, pages 338–346, 2000.
- [20] F. Wang, F. Gong, C. Sargor, K. Goseva-Popstojanova, K. Trivedi, and F. Jou. SITAR: A scalable intrusion-tolerant architecture for distributed services. *Proc. of 2nd Annual IEEE Systems, Man, and Cybernetics Informations Assurance Workshop, West Point, NY, June 2001*.