Sensitivity Analysis of an Attack Containment Model

Ram Dantu[1], João W. Cangussu[2], and Janos Turi[3]

[1] Department of Computer Science, University of North Texas
rdantu@unt.edu
[2] Department of Computer Science, University of Texas at Dallas
[3] Department of Mathematical Sciences, University of Texas at Dallas
{cangussu, turi}@utdallas.edu

Abstract. A feedback control model has been previously proposed to regulate the number of connections at different levels of a network. This regulation is applied in the presence of a worm attack resulting in a slow down of the spreading worm allowing time to human reaction to properly eliminate the worm in the infected hosts. The feedback model constitutes of two queues, one for safe connections and another for suspected connections. The behavior of the proposed model is based on three input parameters to the model. These parameters are: (i) the portion of new connection requests to be sent to the suspect queue, (ii) the number of requests to be transferred from the suspect to the safe queue, and (iii) the time out value of the requests waiting in the suspect queue. The more we understand the effects of these parameters on the model, the better we can calibrate the model. Based on this necessity, a sensitivity analysis of the model is presented here. The analysis allows for the computation of the effects of changing parameters in the output of the model. In addition, the use of a sensitivity matrix permits the computations of not only changes in one parameter but also combined changes of these parameters. From the sensitivity analysis we have verified our assumption that the changes in the input parameters have no effect on the overall system stability. However, there will be a short period of instability before reaching a stable state.

1 Introduction

In the past active worms have taken hours if not days to spread effectively. This gives sufficient time for humans to recognize the threat and limit the potential damage. This is not the case anymore. Modern viruses spread very quickly. Damage caused by modern computer viruses (example - Code red, sapphire and Nimda) [1, 2] is greatly enhanced by the rate at which they spread. Most of these viruses have an exponential spreading pattern [2]. Future worms will exploit vulnerabilities in software systems that are not known prior to the attack. Neither the worm nor the vulnerabilities they exploit will be known before the attack and thus we cannot prevent the spread of these viruses by software patches

P. Kantor et al. (Eds.): ISI 2005, LNCS 3495, pp. 127–138, 2005.

or antiviral signatures [3]. A feedback control model has been proposed to slow down the spreading rate of a worm in order to allow proper human reaction. Experiments have shown that the proposed model is an effective alternative to achieve such goals.

The feedback model constitutes of two queues, one for safe requests and one for suspected requests. A set of parameters determines the behavior of the model based on: (i) the portion of new requests to be sent to the suspect queue (also called delay queue), (ii) the number of requests to be transferred from the suspect to the safe queue, and (iii) the time out value of the requests waiting in the suspect queue. The accuracy and behavior of the model depends on the selection of these parameters. In order to use the model effectively, it is important to determine how a small change in one parameter impacts the overall results. For a given network topology, a set of hosts, firewall rules, and IDS signatures, we need to configure a set of input parameters. However, the selection of these parameters is critical for optimum operation of the state model. For example, the selection decides on how fast we can converge to a given set point. Sensitivity analysis aims to ascertain how a given model depends on its input parameters [4]. This analysis will help us in determining how confident are we in our results and how much will the results change if our selected values are slightly wrong. We will give a completely different outcome or change the outcome only slightly. Also, parameters can be changed at any point in time and the study conducted here allows the determination of the side effects of these changes at different stages of a system. That is, what are the consequences if changes are performed before the infection, at early stages of the infection, or later once almost all the nodes have already been affected? In summary, the goal of this paper is to improve the understanding of the behavior of the model to allow a better selection of parameter values to consequently improve its performance under distinct circumstances.

This paper is organized as follow. The state model used for the feedback control of an attack is described in Section 2. The technique used to compute the sensitivity matrix to analyze the model as well as a detailed description of the results on the sensitivity analysis are presented in Section 3. Conclusions and remarks are drawn in Section 4.

2 Feedback Model

The architecture of the proposed approach [5] is composed of (centralized or distributed) controllers that are responsible for collection and monitoring of all the events in the network. These controller are knowledgeable about the network topology, firewall configurations, security policies, intrusion detections and individual events in the network elements. They are logical functions and can be deployed anywhere in the network. The controllers are represented by state models and are described next.

2.1 State Model

It is assumed that, as the number of requests increases, a portion of them will be sent to a delay queue to be served later. The remaining ones are sent to a safe queue to be served immediately. The overall structure of this queuing system is depicted in Figure 1. Parameters related to the size of the delay queue and the number of dropped (time-out) connections are used to control the total number of connections resulting in a slow down of a spreading worm. Sapphire worm spreading is taken as an example to show the applicability of our approach. The number of requests generated by the worm increase according to an s-shape format [2]. The goal of the example is to slow down the spreading velocity of a worm by controlling the total number of connections ($C(t)$) detected by a firewall. A model capturing the behavior of the system, i.e., how the number of total connections is changing is needed to achieve this goal. We assume here that as the number of request increases, a portion of them will be sent to a delay queue to be served later. Parameters related to the size of the delayed queue and the number of dropped connections are used to control the total number of connections resulting in a slow down of a spreading worm [5]. The rate of change of the number of requests ($\frac{\partial C}{\partial t}$) is proportional to the number of served requests ($-C(t)$), plus the number of connections transferred from the delayed queue ($\beta \times D(t)$, where β is the transfer rate and $D(t)$ is the number of delayed requests on the queue) and a portion of the new requests sent directly to the safe queue ($\alpha \times u(t)$, where α is the percentage of not delayed requests). This results in Eq. 1 below.

$$\dot{C}(t) = -C(t) + \beta D(t) + \alpha u(t) \tag{1}$$

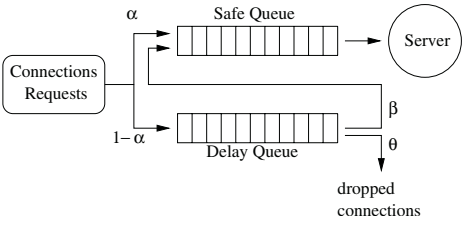

Fig. 1. Queue system which the state model is constructed upon

The rate of change in the size of the delayed queue ($\frac{\partial D}{\partial t}$) is proportional to a fraction of the new incoming requests sent to the queue ($(1-\alpha) \times u(t)$) minus the requests transferred to the safe queue ($-\beta \times D(t)$, where β is the transfer rate), minus the time-out requests($-\theta \times D(t)$, where θ is the time-out rate). This leads to Eq. 2.

$$\dot{D}(t) = -\beta D(t) - \theta D(t) + (1-\alpha)u(t) \tag{2}$$

It is assumed here that all requests in the safe queue, at one point in time, are served. Therefore, at each time a server is allocated to handle the requests at the safe queue, it serves all the new incoming requests plus all the ones transferred from the delayed queue at the previous time period. Combining Eqs. 1 and 2 in a state variable format leads to system in Eq. 3.

$$\begin{bmatrix} \dot{C}(t) \\ \dot{D}(t) \end{bmatrix} = \begin{bmatrix} -1 & \beta \\ 0 & -(\beta+\theta) \end{bmatrix} \begin{bmatrix} C(t) \\ D(t) \end{bmatrix} + \begin{bmatrix} \alpha \\ 1-\alpha \end{bmatrix} u(t) \tag{3}$$

The output part ($Y(t)$) of the system from Eq. 3 can be designed as needed, as long it is a function of the state vector $X(t) = [C(t)\ D(t)]^T$. Let us assume we are interested in the size of both queues $C(t)$ and $D(t)$ in addition to the rate of change in the safe queue ($\dot{C}(t)$). The size of the queues are the values of the state vector. The first derivative of $C(t)$ is represented by Eq. 1. The three desired outputs specified above and the respective equations lead to the output part ($Y(t)$) of a state model as presented by Eq. 4.

$$\begin{bmatrix} C(t) \\ \dot{C}(t) \\ D(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & \beta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} C(t) \\ D(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \alpha \\ 0 \end{bmatrix} u(t) \tag{4}$$

Since Eqs. 3 and 4 represent our model, we refer to them, hereafter as the `FBAC Model` (Feedback Attack Containment). Now let us assume an input $u(t)$ as in Eq. 5 presenting an S-shape behavior.

$$\dot{u}(t) = Ku(t)(1 - \frac{u(t)}{s}) \tag{5}$$

where s represents the saturation point of the S-curve and K is the growth parameter. Eq. 6 represents the solution of Eq. 5 where $u(0)$ represents the initial condition. It has an S-shape format mimicking the behavior of a spreading worm.

$$u(t) = \frac{s}{1 + \left(\frac{s}{u(0)} - 1\right)\ e^{-Kt}} \tag{6}$$

Now consider a system with the following parameter values: $\alpha = 0.3$ (only 30% of the requests are served immediately while the remaining 70% go to the delay queue), $\beta = 0.15$ (15% of the requests in the delay queue are transferred to the safe queue) and $\theta = 0.2$ (20% of the requests on the delay queue are timed-out at each time stamp). Also, consider the input $u(t)$ as specified in Eq. 6 with parameters $K = 0.08$ and $s = 30$. The output of the model for the three specified values are shown in Figure 2. As can be seen in the figure, the size of both queues follow the S-shaped behavior of the input function. However, the safe queue saturates with eighteen requests while the delayed queue saturates with sixty requests per time unit. The velocity $\dot{C}(t)$ (rate of change of requests) increases initially until a inflection point is reached and it starts to decrease until the saturation level in the size of the save queue is achieved and the velocity goes to zero. The effects of the values of the parameters and consequences of changes in these values are analyzed next in Section 3.

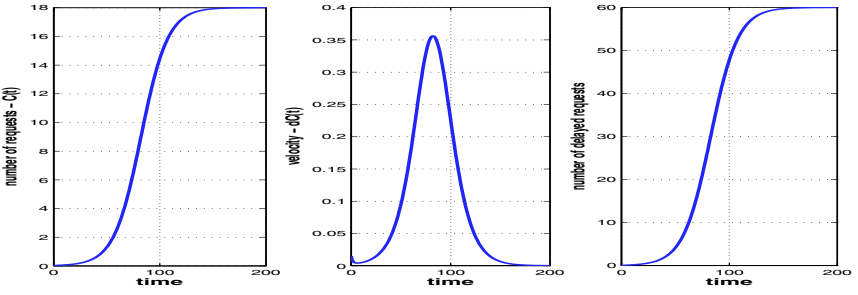

Fig. 2. Output behavior of the `FBAC Model` for $\alpha = 0.3$, $\beta = 0.15$, and $\theta = 0.2$

3 Sensitivity Analysis

The understanding of how the state variables change according to changes in the input parameters of the model can provide information to help on the determination of the values of these parameters. In other words, we are interested in analyzing how sensitive is the model to changes in its parameters. Three parameters are of interest in the state model from Eqn. 3: β, θ, and α. Consider now a parameter vector $G = [\beta\ \theta\ \alpha]^T$. The goal is to compute $\dfrac{\partial X(t)}{\partial G}$, where $X = [C\ D]^T$ is the state vector of interest. A sensitivity matrix can be computed to achieve this goal [4].

3.1 Computation of the Sensitivity matrix

One alternative for the computation of the sensitivity matrix would be the use of the Kronecker product [6] that allows the differentiation of vectors with respect to a matrix. However, the fact that the model in Eq. 3 presents a nice upper triangular format combined to the fact that we have a vector and not a parameter matrix allows an easier way to achieve the same goals. The differential equations for the elements of the state vector X are defined by Eqs. 7 and 8 below.

$$\dot{C}(t) = -C(t) \ + \ \beta D(t) \ + \ \alpha u(t) \tag{7}$$

$$\dot{D}(t) = -(\beta + \theta)D(t) \ + \ (1 - \alpha)u(t) \tag{8}$$

Solving Eqs. 7 and 8 result in Eqs. 9 and 10 below.

$$C(t) = e^{-t}c0 + \int_0^{\tau} e^{-t+\tau}\left(\beta\, D(\tau) + \alpha\, u(\tau)\right)\ d\tau \tag{9}$$

$$D(t) = e^{t(-\beta-\theta)}d0 + \int_0^{\tau} e^{(-\beta-\theta)(t-\tau)}\left(1-\alpha\right) u(\tau)\ d\tau \tag{10}$$

A Jacobian matrix [7] for the functions $C(t)$ and $D(t)$ with respect to the parameters β, θ, and α can be compute as in Eqn. 11. This matrix represents the sensitivity matrix $SM(t)$ for the model in Eqn. 3.

$$SM(t) = \begin{bmatrix} \frac{\partial C(t)}{\partial \beta} & \frac{\partial C(t)}{\partial \theta} & \frac{\partial C(t)}{\partial \alpha} \\ \frac{\partial D(t)}{\partial \beta} & \frac{\partial D(t)}{\partial \theta} & \frac{\partial D(t)}{\partial \alpha} \end{bmatrix} \tag{11}$$

where the elements of the matrix above are computed according to Eqs. 12,..., 17.

$$\frac{\partial D(t)}{\partial \beta} = -te^{t(-\beta-\theta)}\, d0 - \int_0^{\tau} (t-\tau)\, e^{(-\beta-\theta)(t-\tau)}\, (1-\alpha)\, u\; d\tau \tag{12}$$

$$\frac{\partial D(t)}{\partial \theta} = -te^{t(-\beta-\theta)}\, d0 - \int_0^{\tau} (t-\tau)\, e^{(-\beta-\theta)(t-\tau)}\, (1-\alpha)\, u\; d\tau \tag{13}$$

$$\frac{\partial D(t)}{\partial \alpha} = -\int_0^{\tau} e^{(-\beta-\theta)(t-\tau)} u\; d\tau \tag{14}$$

$$\frac{\partial C(t)}{\partial \theta} = \int_0^{\tau} e^{-t+\tau}\beta \left(-\tau\, e^{\tau\,(-\beta-\theta)}\, d0 - \int_0^{z} (\tau - z)\, e^{(-\beta-\theta)(\tau-z)}\, (1-\alpha)\, u\; dz \right) d\tau \tag{15}$$

$$\frac{\partial C(t)}{\partial \beta} = \int_0^{\tau} e^{-t+\tau} \left(e^{\tau\,(-\beta-\theta)}\, d0 + \int_0^{z} e^{(-\beta-\theta)(\tau-z)}\, (1-\alpha)\, u\; dz + \beta \times \left(-\tau e^{\tau(-\beta-\theta)}\, d0 - \int_0^{z} (\tau - z)\, e^{(-\beta-\theta)(\tau-z)}\, (1-\alpha)\, u\; dz \right) \right) d\tau \tag{16}$$

$$\frac{\partial C(t)}{\partial \alpha} = \int_0^{\tau} e^{-t+\tau}\beta \int_0^{z} e^{-(\beta+\theta)*(\tau-z)} * u\; dz\; d\tau \tag{17}$$

Matrix $SM(t)$ can now be used to compute variations in the state variables in response to perturbations in specific parameters or combinations thereof. Eq. 18 is used to compute the variations.

$$\begin{bmatrix} \Delta C(t) \\ \Delta D(t) \end{bmatrix} \;=\; SM(t) \;\times\; \Delta_m \qquad where\ \Delta_m = \begin{bmatrix} \Delta\beta \\ \Delta\theta \\ \Delta\alpha \end{bmatrix} \tag{18}$$

To obtain the variations for individual or combined changes in the values of parameters, we set the Δ_m matrix. For example, to analyze the effects of a 5% change in β and a -10% change in θ, we set Δ_m to $[0.05\beta \;\; -0.1\theta \;\; 0]^T$, which when substituted in Eq. 18 gives us $[\Delta C(t)\;\; \Delta D(t)]^T$. $\Delta C(t') < 0$ at time $t = t'$ implies a decrease in the number of connections in the safe queue. Logically, $\Delta C(t') > 0$ implies an increase in the corresponding queue. A positive slope represents an increase in the queue size while a negative one represents a decrease. That is, though the overall size of the queue maybe smaller than the queue with no changes, a positive or negative slope represents how the modified system is changing according to time. $\Delta D(t')$ presents a similar behavior with respect to the size of the delayed queue.

3.2 Results of the Sensitivity Analysis

We now use the sensitivity matrix to analyze the sensitivity of the state variables of `FBAC Model` to variations in its parameters. Unless stated otherwise, we assume that `FBAC Model` represents a system with parameter values: $\beta = 0.2$, $\theta = 0.21$, and $\alpha = 0.3$. These values are arbitrary and do not affect the results of the analysis reported here. We also assume here that the input $u(t)$, representing the number of new requests follows the behavior of Eq. 6 fort $K = 0.08$ and $s = 20$.

Assuming the input above and the parameters values specified before, Figure 3 shows the behavior of the state variables C and D with respect to time for a 10% increase in the value of β ($\Delta_m = [0.1 \times \beta\ 0\ 0]^T$). As can be seen in the figure, the increase in the β parameter has opposite effects with respect to the safe and the delayed queue. β determines how many connections are removed from the delayed queue and send to the safe queue. Therefore, increasing β will naturally decrease the size of the last while increasing the size of the former. Also, $D(t)$ is more sensitive to this change than $C(t)$ as the overall absolute value of Δ_D is larger than Δ_C. The fact that α is less than 0.5 means that more requests are being sent to the suspect queue than to the safe queue. Consequently, the first one grows faster and larger then the second and more elements are transferred when β is increased. When no changes for the parameters are observed, the safe queue saturates around 13 connections per time unit and the delayed queue saturates at 34 connections per time unit. The size of the queues at saturation point shows that a 2.7% change in the safe queue is observed when α is increased by 10%. The delayed queue is more sensitive to the same change presenting a 5% increase.

Changes in α present a similar behavior as described for β as it increases the size of the safe queue and decreases the size of the delayed queue. However, the fact that α is larger than β magnifies the sensitivity. Consequently, a combined change of α and β also has a similar behavior. However, the sensitivity in this case is a result of the sum of the isolated changes for α and β. That is, $\Delta_{C_{\Delta_\alpha=10\%\Delta_\beta=10\%}} = \Delta_{C_{\Delta_\alpha=10\%}} + \Delta_{C_{\Delta_\beta=10\%}}$. The behavior of Δ_D being similar.

When the value of θ is increased both queues present a decay in their size. As θ represents the number of time out requests at the delayed queue, an increase in its value will naturally decrease the size of the queue as more requests are

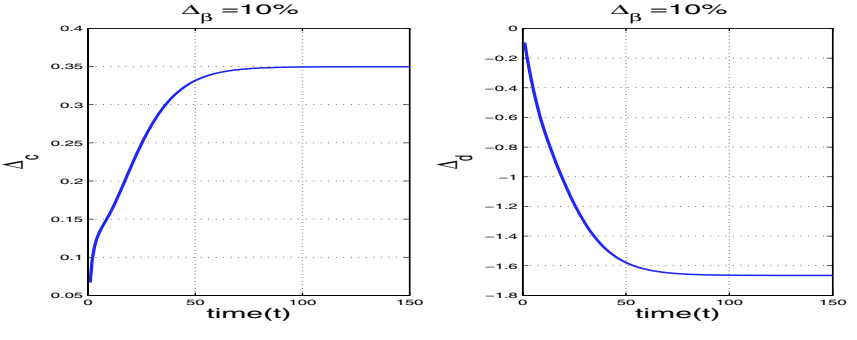

Fig. 3. Sensitivity of $C(t)$ and $D(t)$ for a 10% increase in the value of β

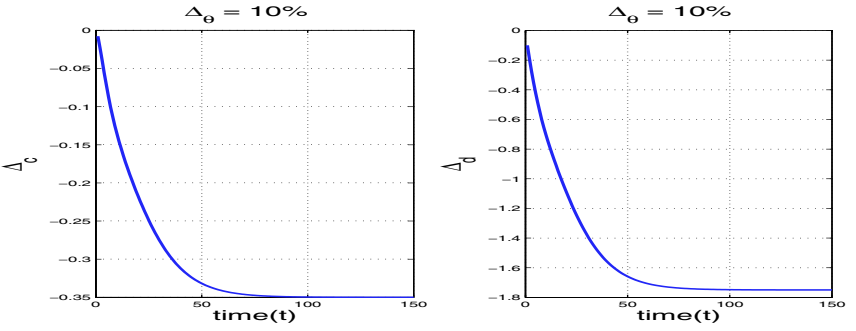

Fig. 4. Sensitivity of $C(t)$ and $D(t)$ for a 10% increase in the value of θ

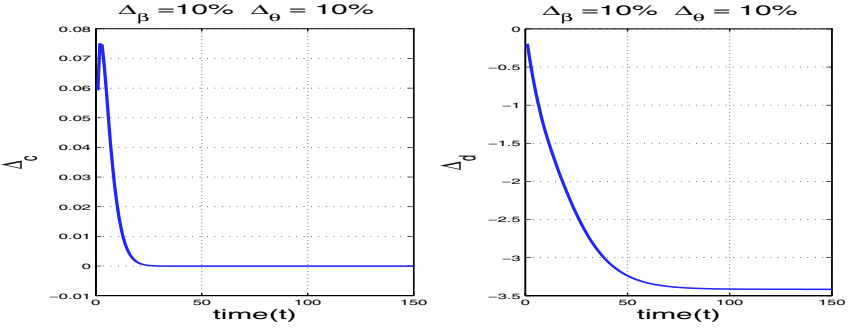

Fig. 5. Sensitivity of $C(t)$ and $D(t)$ for a 10% increase in the value of β and θ

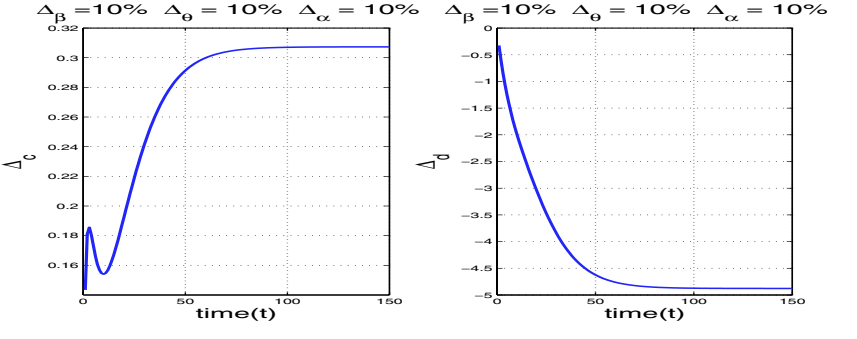

Fig. 6. Sensitivity of $C(t)$ and $D(t)$ for a 10% increase in the value of α, β, and θ

timing-out. Consequently, a smaller number of requests are transferred from the delayed to the safe queue which also experiences a decrease. Such behavior is shown in Figure 4. Combined changes for θ and α present a behavior similar to Figure 4. Though more requests are sent to the safe queue, more are dropped from the delayed queue and less are transferred from the last to the former.

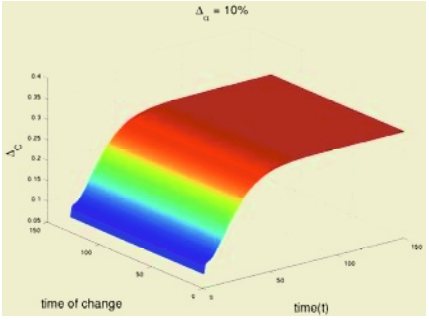

Fig. 7. Sensitivity of the safe queue with respect to a 10% change in the α parameter at different time stamps

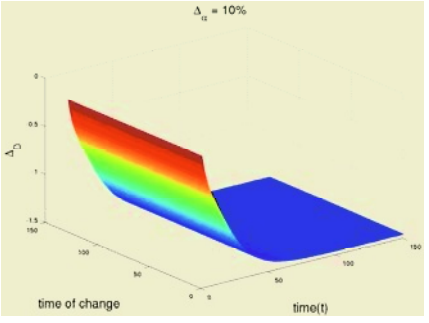

Fig. 8. Sensitivity of the delayed queue with respect to a 10% change in the α parameter at different time stamps

A combined change of θ and β produces initially small increase in the size off the safe queue. Though more requests are being transferred, initially, the dropped (due to time out represented by θ) requests are more than the transferred ones since θ is slightly larger than β. After some time, the changes average out and the sensitivity goes to zero as noticed from Figure 5.

A different behavior is observed when all the three parameters are subjected to 10% changes. In this case, the safe queue suffers an overall increase with some oscillation before saturating as can be seen in Figure 6. The oscillation is due to the fact that the overall number of requests (represented by $u(t)$) grows faster at the beginning and since α has also being increased, it compensates the changes in θ and β. After sometime, the changes in θ and β overcomes the growth of $u(t)$ and the change in α and the systems stabilizes.

All the results above are for changes of the parameters at time $t = 0$, i.e., at the beginning of the spreading of a worm. Next, we analyze not only the changes in the parameters but also the effect of delaying these changes ($t > 0$). The following plots represent the time, the time of the change of the parameter(s), and the sensitivity. As depicted in Figures 7 and 8, the time of change of α has no effect on the behavior of both queues. Though more requests are sent directly to the safe queue, less are being transferred from the delayed to the safe at any point in time. This explains the behavior of Figures 7 and 8.

Changes in β represent the behavior shown in Figures 9 and 10. The size of the delayed queue builds up with time. The later the change in β, the larger the size of the delayed queue at that time. Therefore, more requests are transferred from the delayed to the safe queue and an immediate increase is observed in the last. However, the size of the delayed queue starts to decrease and so does the size of the safe queue. This is observed until the increase in the number of requests (input $u(t)$) compensates this behavior and the system starts to present the same behavior as of changing β at time $t = 0$. This oscillation can be seen as one step toward the equilibrium point, i.e., the saturation level of the queues and the input. The behavior of the delayed queue for a 10% change in θ is similar

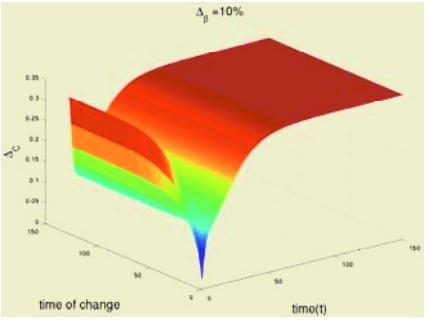

Fig. 9. Sensitivity of the safe queue with respect to a 10% change in the β parameter at different time stamps

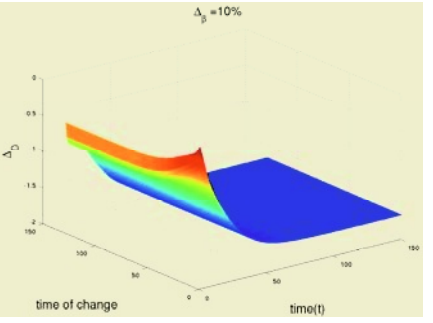

Fig. 10. Sensitivity of the delayed queue with respect to a 10% change in the β parameter at different time stamps

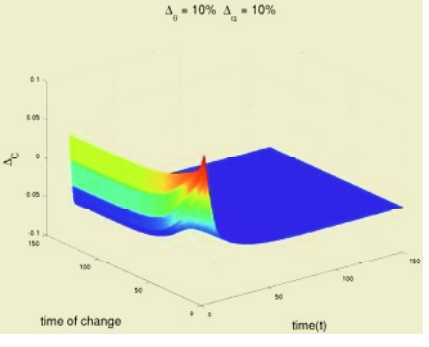

Fig. 11. Sensitivity of the safe queue with respect to a 10% change in α and θ at different time stamps

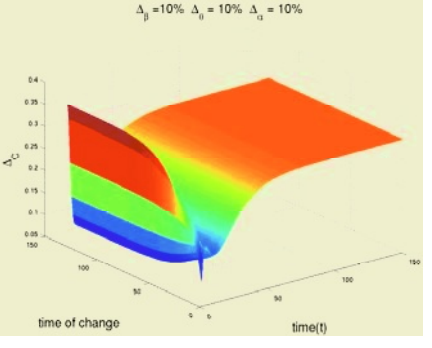

Fig. 12. Sensitivity of $C(t)$ and $D(t)$ for a 10% increase in the value of α, β and θ

to the one in Figure 10 (as it is any change that involves either β, θ, or both). However, the safe queue presents a continuous decrease up to the saturation point as less requests are now transferred. As with the changes in the parameters at time $t = 0$, the combined change of α and β presents a similar behavior as the change of β (Figures 9 and 10). However, as expected, the magnitude of the changes are amplified when both parameters suffer a simultaneous perturbation. The sensitivity of `FBAC Model` with respect to a combined change of 10% in parameters α and θ at increasing time stamps is depicted in Figure 11. In this case, at the beginning of the change when the input values are smaller, the change in α overcomes the change in θ. That is, initially the additional requests sent directly to the safe queue (a consequence of increasing α) are larger than the additional time-out requests (a consequence of increasing θ). However, as the delayed queue saturates, the time-out requests become larger than the extra

requests send to the safe queue which then presents an overall decrease as seen in Figure 11. The behavior for combined changes of β and θ has a similar behavior as described above. However, the magnitude of the sensitivity and the reasons for the behavior are different. In the first case the maximum increase of the safe queue is around 0.3% which is amplified by more than six times when the changes are for β and θ. Until the size of the delayed queue starts to decrease due to the change in θ, more requests are transferred to the safe queue which presents an initial increase in its size. After some time period, the changes compensate each other and the sensitivity goes to zero.

The plot in Figure 12 shows the sensitivity of the safe queue of `FBAC Model` when all the three parameters are increased by the same factor. A prompt increase in the size of the queue is observed at the beginning. The later the changes of the parameters, the larger the increase. After that, the queue size suffers a decrease, denoted by the initial negative slope in Figure 12, followed by an increase until a stable point is achieved. This oscillation is due to the same arguments presented earlier when discussing Figure 6.

4 Concluding Remarks

Automatic containment of the propagation of computer worms is a difficult task since we do not know the behavior of future worms. But we do know from past experience that worms propagate by attempting large number of connections to other hosts. If successful, the infected remote machine will try to infect the other machines and the process continues. We are researching a containment model based on the feedback control theory. In order to validate our model, we have carried out sensitivity analysis on the input parameters (α, β, and θ). We observed the change in size of these two queues with res pect to time. In addition we observed another dimension (time of change) with respect to the state of the worm propagation.

We have increased the values of α, β, and θ in increments of 10%. In all the three cases, the system became stable before 50 time units and this is well before the beginning of the attack period. Hence from our analysis we observe that there is no impact of changing the individual input parameters on the system accuracy or the stability in containing the attacks. But when we apply the change at a later time, say during 100 time units, a sudden increase in β will result in sudden increase in C and we consider this as a merging or clumping effect. This is due to sudden transfer of connections from the delay queue to safe queue. Next we have increased two out three parameters simultaneously in increments of 10%. In all the cases we observed a small oscillation where the size of safe queue increases/decreases and suddenly reverses the trend. For example, simultaneous increase of θ and α will result in a sudden increase of safe queue but trend reverses as θ (drop of connections in the delay queue) increases. This is due to the fact that initially large number of connections are enqueued to the safe queue in addition to transfer of connections from delay to the safe queue. However, as the connections are timed out and dropped at the delay queue, less

and less number of connections are transferred to the safe queue. Finally we have increased all three parameters simultaneously and we found similar oscillation effects in the beginning. This is expected behavior when compared with the two-parameter-change.

In all the experiments described above, the size of the oscillation depends on the instant of change with respect to the S-shaped input. The later the changes, the larger the oscillation but the system comes back to a stable state within short period of time and hence there is no impact of this change on the outcome of the model. Hence from the sensitivity analysis we have verified our assumption that the changes in the input parameters have no effect on the overall system stability. But there will be short period of instability in the beginning and after that the system reaches a stable state and stays there for rest of the time. We have analyzed the impact due to small changes in the input parameters. The overall system behavior may be different for large variations of the input parameters. The analysis of these changes is subject of future work.

References

1. S. Staniford, V. Paxson, and N. Weaver, "How to own internet in your spare time," in *Proceedings of the USENIX Security Symposium*, pp. 149–167, August 2002.
2. D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "The spread of the sapphire worm." http://cs.berkely.edu/ nweaver/sapphire.
3. M. M. Williamson, "Throttling viruses: Restricting propagation to defeat malicious mobile code," in *18th Annual Computer Security Applications Conference*, pp. 61–68, December 2002.
4. *Sensitivity Analysis.* John Wiley & Sons, 2000.
5. R. Dantu, J. W. Cangussu, and A. Yelimeli, "Dynamic control of worm propagation," in *Proceedings. ITCC 2004. International Conference on Information Technology*, vol. 1, pp. 419–423, April 5-7 2004.
6. J. W. Brewer, "Matrix calculus and the sensitivity analysis of linear dynamic systems," *IEEE Transactions on Automatic Control*, vol. 23, pp. 748–751, August 1978.
7. R. A. DeCarlo, *Linear systems : A state variable approach with numerical implementation.* Upper Saddle River, New Jersey: Prentice-Hall, 1989.