Tradeoffs of Service, Security and Attack for Cyber System Survivability

Nong Ye and Billibaldo M. Aranda
Arizona State University
Tempe, Arizona, U.S.A.
nongye, bimartin@asu.edu

***Abstract*—Tradeoffs within the limitation of given system resources often have to be made for a cyber system to sustain its more critical services under the damage of cyber attacks. Understanding and modeling cause-effect dynamics of activities (service, security and attack), system state and Quality of Service (QoS) are essential to establish the dynamic adaptation capability of making tradeoffs among services, security mechanisms and cyber attacks. This paper presents our methodology of collecting system dynamics data under various service, security and attack conditions and analyzing system dynamics data to uncover cause-effect dynamics of activities, system state and QoS performance and especially tradeoff effects of services, security mechanisms and attacks. The methodology is illustrated through our study involving an experiment to collect system dynamics data under various conditions of two services (voice communication service and motion detection service), two security protection mechanisms (encryption and intrusion detection), and five attacks. Experimental data of system dynamics is analyzed to produce findings about activitiy-state-QoS dynamics and tradeoff effects of services, security mechanisms and attacks. We also illustrate how such findings can be used to assess the damage of attacks on the state of system resources and QoS performance and determine system adaptation strategies for system survivability under the damage of attacks.**

Keywords-Quality of service; voice communication service, motion detection service, security protection, cyber attacks, survivability

I. INTRODUCTION

When cyber attacks cause unforeseen damage to a cyber system, the system must have the capability of dynamically adapting itself to survive and recover from the damage [1-6]. The dynamic adaptation usually involves pulling out reserves of additional system resources or making tradeoffs within limitations of existing system resources. The use of system reserves for infrequent occurrence of system damage by cyber attacks is costly. There is always a limitation in how much system reserves can be held. There is always a possibility that the damage caused by cyber attacks leads to a severe shortage of system resources. Hence, tradeoffs within the limitation of given system resources must be made for the system to sustain its more critical services under the damage of cyber attacks.

Three types of activities may occur on a cyber system at the same time: service activities, security activities, and attack activities. Various computer and network services may be running on the system to fulfill a mission. Security mechanisms may also be running at the same time to protect the system from cyber attacks. Cyber attacks may add other activities to the system. An activity causes changes in the state (e.g., availability) of system resources which in turn lead to changes in system performance and Quality of Services (QoS) [7-9]. QoS is a subset of system performance measures that are specifically associated with services for fulfilling a mission. Understanding and modeling such cause-effect relationships of activities, system state and QoS are essential to establish the dynamic adaptation capability of making tradeoffs among services, security mechanisms and cyber attacks. However, such models of activity-state-QoS dynamics are not readily available from the design of computer and network systems which provides mostly algorithm-based operational models. As pointed out in [10], activity-state-QoS dynamic models from existing studies focus mostly on specific resources (e.g., router, CPU, memory, and hard disk) individually and limited aspects of dynamic models. The dynamic adaptation for survivability should not simply consider effects of services, security mechanisms and cyber attacks on an individual resource or the reconfiguration of an individual resource because all system resources interact with and place constraints on one another. The QoS performance depends on activity-state-QoS dynamics at the system scale, i.e., effects of service, security and attack activities on all system resources and QoS of all competing service processes/threads.

This paper describes our methodology of uncovering activity-state-QoS dynamics driven by service, security and attack activities and using findings of activity-state-QoS dynamics to consider tradeoffs among services, security mechanism, and cyber attacks for system survivability under the damage of cyber attacks. The methodology includes an experiment to collect system dynamics data under various conditions of services, security mechanisms and cyber attacks and the statistical analysis of the experimental data to uncover activity-state-QoS dynamics and tradeoffs among services, security mechanism and cyber attacks. The methodology is illustrated through a specific experiment involving two specific services of voice data communication and motion detection, two security mechanism of encryption and intrusion detection, and five cyber attacks of ARP Poison, ping flood, vulnerability scan, fork bomb, and remote dictionary.

In Section II of the paper, the experiment along with the collection of experimental data is described. Section III presents the statistical analysis of experimental data. Section IV reports major findings of activity-state-QoS dynamics and service-security-attack tradeoffs. Section V concludes the paper.

II. Experiment for Data Collection

An experiment was conducted to collect system dynamics data under various conditions of services, security mechanisms and cyber attacks. Section II.A describes the experimental conditions and setup. Section II.B specifies the data collection facility used.

A. Experimenrtal Conditions and Setup

The experiment includes two services of voice data communication and motion detection, two security mechanisms of encryption and intrusion detection, and five cyber attacks of ARP Poison, ping flood, vulnerability scan, fork bomb, and remote dictionary.

The voice communication service (VCS) is developed by converting an open-source video conferencing software package [11] into a web service using C# in .NET. There is one server and multiple clients, each running on a separate computer. Multiple clients can simultaneously request and receive the voice communication service from the server. For VCS, encryption is used as the security protection mechanism. The encryption is implemented within VCS software using the Advanced Encryption Standard (AES) developed by Daemen and Rijmen [12]. Three service parameters, the number of clients (C), the sampling rate of recording the voice data stream (S), and the size of the buffer holding the voice data before its transmission (B), are incorporated to introduce various levels of service demands and system configuration. The AES encryption mechanism has two parameters: the percentage of encryption (E) and the key length (K). VCS is a communication-intensive service. QoS of VCS is concerned mainly with the throughput performance of voice data communication. Table I shows various levels of the service and security parameters for VCS that are used in the experiment to set up various experimental conditions.

TABLE I. Experimental Conditions for VCS

Parameters	Level 1	Level 2	Level 3
Sampling rate (S)	44100Hz	132300Hz	220500Hz
Number of Client (C)	1	3	5
Buffer size (B)	16KBytes	32KBytes	48Kbytes
Encryption Percentage (E)	0%	50%	100%
Key Length (K)	128 bits	192 bits	256 bits
Cyber Attack	no attack	attack	

The motion detection service (MDS) is developed by converting an open-source motion detection software package [13] into a web service using C# in .NET. MDS processes video files of different sizes and analyze them to estimate the motion level. The service and clients of MDS run on the same computer. Video files are stored on the server. All clients run concurrently through their threads processing video files. There are two service parameters: the video size (VS) of the video file and the number of threads (T) which represents the number of clients using MDS. Table II shows various levels of VS and T that are used to set up various experimental conditions for MDS. For MDS, we run intrusion detection software SNORT as the security protection mechanism which is implemented to run independently from MDS. SNORT [14] is a network intrusion prevention and detection system that performs intrusion detection based on signature recognition and protocol-based anomaly detection. MDS is a computation-intensive service. QoS of MDS is concerned mainly with 1) the motion level which is measured by the average number of pixels that is detected to change in a frame divided by the total number of pixels in a frame, and 2) the delay which is measured by the average time taken to process 1 frame.

TANBLE II. Experimental Conditions for MDS

Parameters	Level 1	Level 2	Level 3
Video size (VS)	22x18	44x36	88x72
Threads (T)	1	3	5
Security (Se)	no security	security	
Cyber Attack	no attack	attack	

Cyber attacks with different attacking mechanisms and objectives can compromise different system resources and change the state of system resources and QoS performance of services in different ways. The Common Attack Pattern Enumeration and Classification (CAPEC) developed by MITRE Corporation [15] gives a classification and description of attack mechanisms. In this study, five cyber attacks of ARP Poison, ping flood, vulnerability scan, fork bomb, and remote dictionary, which have different attack mechanisms, are included in the experiment. The ARP poison attack is a man-in-the-middle attack in which the attacker poisons the ARP tables of victim computers in order to access messages between those computers. Cain and Abel® v4.9.30 [16] is used to perform the ARP Poison attack. The ping flood attack is a denial of service (DoS) attack by sending large numbers of ICMP echo requests to a victim computer and thus flooding the network bandwidth of the computer. Ping ® v2.0 [17] is used to perform the attack. The vulnerability scan attack searches for vulnerabilities in a computer or network system by sending SYN packets to all the ports (0 – 65535) to identify open ports and analyzing responses to discover network services running at those ports. Nmap ® v4.76 [18] is used to perform the attack. The fork bomb attack is an DoS attack through resource depletion in which a program continuously creates copies of itself consuming system resources and slowing down the processing of legitimate processes. The remote dictionary attack uses a dictionary of passwords to gain access to the administrator account of a victim computer through the Windows remote desktop connection utility.

Tscrack® v2.1 [7] is used to perform the remote dictionary attack.

Table I shows totally 486 (=3*3*3*3*3*3*2) combinations of experimental conditions for VCS and 36 combinations of experimental conditions for MDS. The 486 conditions for VCS are run in a random order with one run for each condition to collect 30 observations of system dynamics data from the server. Then the reserved order of the random order is used to run the 486 conditions again. System dynamics data from both orders of experimental runs are used in data analysis in order to eliminate the possible effect of a run order on data observations. The 36 experimental conditions are MDS are run in a similar manner to collect system dynamic data.

B. Data Collection

Windows performance objects [19] are used to collect system dynamics data that reflects activities, state of various system resources and their interaction and QoS performance on the server for VCS and the server for MDS. Totally 15 performance objects, including Process, Memory, Paging File, Physical Disk, IP, UDP, TCP, Server, System, Web Services, Network, Objects, Processor, Redirector and Terminal Service session (TSS), are monitored during the data collection. Each object has a number of counters that give activity, state and performance variables of the object. There are 384 variables from these 15 performance objects. The interval of data sampling is set to one second.

III. Statistical Data Analysis

The experimental data is first screened to remove variables that are not related to effects of the experimental parameters. The remaining variables are then analyzed to uncover activity-state-QoS dynamics and tradeoff effects of services, security mechanism and cyber attacks.

A. Data Screening

Data variables are removed from further data analysis if they are not related to effects of the experimental parameters. A data variable is removed for one of the following reasons.

- The variable records the highest or peak value since the computer is last restarted. For example, the variable *Virtual bytes peak* of the Process object records the highest virtual address space in bytes used by a service (VCS or MDS) since the server is last restarted. Hence, in one order of the experimental runs, values of the variable keep increasing over time.
- The variable collects the cumulative value over time since the computer is last restarted. For example, the variable *Datagrams outbound discarded* of the IP object counts the number of output IP datagrams with no errors that are discarded due to reasons such as lack of buffer space since the computer is last restarted. Values of the variable keeps increasing with time.
- The variable collects data that is not affected by the experimental conditions. For example, the variable *Priority base* of the Process object measures the base priority of the service (VCS or MDS) running. Neither of the services is designed to adjust the service priority.

Totally 165 variables are removed. The remaining 219 variables are used in further data analysis.

B. Data Analysis

An Analysis of Variance (ANOVA) [20] is performed for each variable of Windows performance object to determine if the variable shows effects of the VCS experimental parameters in Table I. These experimental parameters are the independent variables in the ANOVA. The variable from the Windows performance object is the dependent variable in the ANOVA. The Tukey HSD (Honest Significant Difference) test [20] is also performed on each of the variables with a significant main or interaction effect of the experimental parameters to determine experimental conditions that are significantly different from one another. Similar data analysis is performed for the MDS experimental parameters and data. Statistica 7® [21] is used to perform ANOVA and Tukey HSD tests. Figures 1 and 2 show examples of the variables which show significant effects of an experimental parameter, as determined by ANOVA and Tukey HSD tests. Since the experimental parameters are the activity variables, effects of these parameters on each variable of Windows performance objects as determined by ANOVA and Tukey HSD tests reveal cause-effect dynamics of activities, system state and QoS performance as well as tradeoff effects of various parameters on system state and QoS performance variables.

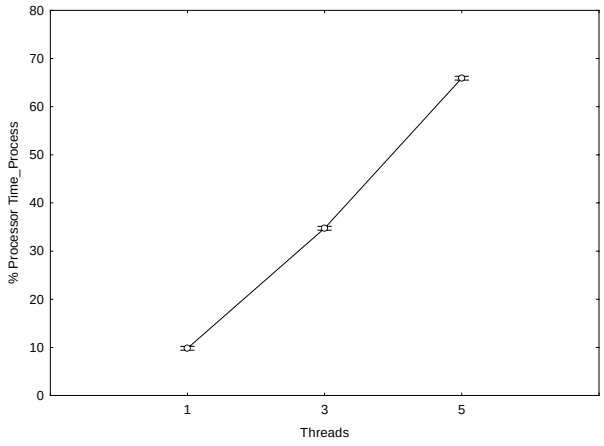

Figure 1. The increasing effect of the number of threads (T) on *% Processor time* of the Process object under the condition of the ARP Poison attack

If multiple experimental parameters have a significant effect on a variable, the Generalized partial Eta squared index [22] is computed and used to determine which effect is more dominant. A parameter with a larger effect size has more dominant effect on a state or QoS performance variable, and should be considered while planning an adaptation strategy for system survivability through tradeoffs between services and security mechanisms during attacks.

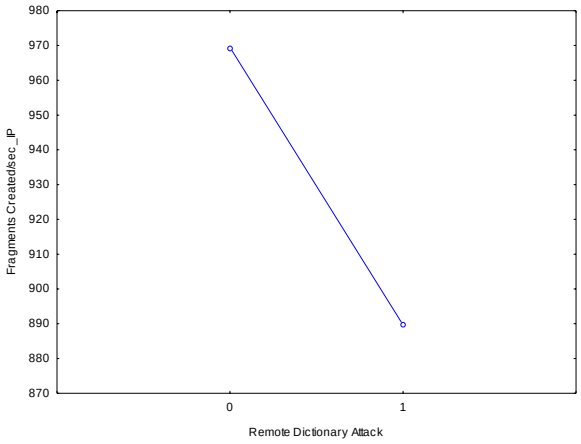

Figure 2. The different effect of the remote dictionary attack v.s. no attack on *Fragments Created/sec* of the IP object

IV. RESULTS AND DISCUSSIONS

Some findings about tradeoff effects of services, security mechanisms and cyber attacks for VCS and MDS are presented in the following sections.

A. Tradeoff Effects of Service, Security and Attack for VCS

Table III lists some findings for VCS that reveal tradeoff effects of service parameters, security parameters and attacks. In Table III, a state or QoS variable is shown in the form of *Counter Name_Object Name*. A number in parentheses in Table III indicates the effect size with a smaller number indicates a larger effect size. For example, Table III shows that *Fragments Created/sec_IP* increases its values as the S level increases, decreases its values as the E level increases, and decreases its value when we go from the no-attack condition to an attack condition. Such tradeoff effects of S, E and A on *Fragments Created/sec_IP* can be used in determining a system adaptation strategy for survivability. *Fragments Created/sec_IP* gives the rate at which datagram fragments are generated before they are sent by IP, and is a QoS measure of the voice communication throughput for VCS. When an attack occurs and degrades QoS of VCS as measured by *Fragments Created/sec_IP*, the sampling rate (S) of VCS can first be increased to feed more voice data for the network transmission, or the percentage of encryption can be decreased as a sacrifice to maintain the throughput level of voice data communication. Note that the effect of S is larger than the effect of E.

For VCS, the encryption percentage (E) has an increasing effect on the memory usage, CPU time and IO activity level, and a decreasing effect on the QoS performance and outgoing network traffic. The sampling rate (S) has an increasing effect on the QoS performance and outgoing network traffic. The key length (K) has an increasing effect on CPU time. The buffer size (B) has a decreasing effect on CPU time and outgoing UDP traffic. Attacks have a decreasing effect on the QoS performance, memory usage and IO activity level. These cause-effect dynamics of services, security mechanisms and attacks on the state of system resources and QoS can be used to determine how attacks cause the damage on the state of system resources and QoS performance and how service, security and system configurations can be adapted for system survivability under the damage of attacks.

TANBLE III. TRADEOFF EFFECTS OF SERVICE PARAMETERS, SECURITY PARAMETERS, AND ATTACKS FOR VCS

	Effects of		
State or QoS variable	**Service parameters**	**Security parameters**	**Attacks (A)**
Fragments Created/sec_IP	Increase with S (1)	Decrease with E (2)	Decrease with A
% Processor Time_Process	Decrease with B (3)	Increase with E (1) Increase with K (2)	
IO Data Bytes/sec_Process		Increase with E (1)	Decrease with A (2)
Page Faults/sec_Process		Increase with E (1)	Decrease with A (2)
Packets sent/sec_Network	Increase with S (1)	Decrease with E (2)	Increase with the vulnerability scan attack
Datagrams sent/sec_UDP	Increase with S (1) Decrease with B (2)	Decrease with E (3)	

B. Tradeoff Effects of Service, Security and Attack for MDS

Table IV lists some findings for MDS that reveal tradeoff effects of service parameters, security parameters and attacks. For example, *Available Bytes_Memory* decreases and the number of threads (T) and also decreases with attacks. When attacks occur and decrease *Available Bytes_Memory*, an adaptation can be taken to decrease the number of threads (MDS clients) to make sure that MDS can still run with sufficient available bytes in memory for continuing to provide the service.

TANBLE IV. TRADEOFF EFFECTS OF SERVICE PARAMETERS, SECURITY PARAMETERS, AND ATTACKS FOR MDS

	Effects of		
State or QoS variable	**Service parameters**	**Security parameters (Se)**	**Attacks (A)**
Available Bytes_Memory	Decrease with T (1)		Decrease with A
Pool Paged Bytes_Memory	Increase with T (1)	Decrease with Se under no-attack (2) Increase with Se under attacks (2)	

For MDS, the number of threads (T) has an increasing effect on the delay, CPU time, IO activity level, committed virtual bytes in memory, disk activity level, and web service traffic. The video size (VS) has an increasing effect on the delay, CPU time, page faults/sec, and motion level at VS of 88x72. The security protection through intrusion detection (Se) has an increasing effect on committed virtual bytes and pool nonpaged bytes in memory. Attacks have an increasing effect on committed virtual bytes in memory and incoming network

traffic. The five attacks in our experiment have little effect on the two QoS performance metrics, motion level and delay. These cause-effect dynamics of services, security mechanisms and attacks on the state of system resources and QoS can be used to determine how attacks cause the damage on the state of system resources and QoS performance and how service, security and system configurations can be adapted for system survivability under the damage of attacks.

V. SUMMARY

This paper presents our methodology of collecting system dynamics data under various service, security and attack conditions and analyzing system dynamics data to uncover cause-effect dynamics of activities, system state and QoS performance and especially tradeoff effects of services and security mechanisms during attacks. Such tradeoff effects can be used to determine the damage of attacks on the state of system resources and QoS performance and system adaptation strategies for system survivability under the damage of attacks. We would like to thank Patrick Hurley at AFRL for his constructive inputs and comments which help improving our work and this paper.

ACKNOWLEDGMENT

This work is sponsored by the Air Force Research Laboratory (AFRL) under award number FA8750-08-2-0155. The U.S. government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of, AFRL, or the U.S. Government.

REFERENCES

[1] L.-J. Zhang, W. Wang, L. Guo, W. Yang and Y.-T. Yang, "A survivability quantitative analysis model for network system based on attack graph," in the Proceedings of the 6th International Conference on Machine Learning and Cybernetics, pp. 19-22, August 2007.

[2] K. Xiao, N. Chen, S. Ren, Limin Shen, Xianhe Sun, Kevin Kwiat and Michael Macalik, "A workflow-based non-intrusive approach for enhancing the survivability of critical infrastructures in cyber environment," in the Proceedings of the 3rd International Workshop on Software Engineering for Secure Systems, pp. 20-26, May 2007.

[3] M. Atighetchi, P. Pal, F. Webber, R. Schantz, C. Jones and J. Loyal, "Adaptive cyberdefense for survival and intrusion tolerance," IEEE Internet Computing, vol. 8, no. 6, pp. 25-33, Nov. 2004.

[4] H. F. Lipson and D. A. Fisher, "Survivability – a new technical and business perspective," in the Proceedings of the 1999 Workshop on New Security Paradigms, pp. 33-39, 1999.

[5] X. Yi and Y. Zhang, "Survivability of information system," in the Proceedings of the 5th International Conference on Information, Communications and Signal Processing, pp. 1551-1555, 2005.

[6] Y. Zuo and B. Panda, "Unifying strategies and tactics: a survivability framework for countering cyber attacks," in the Proceedings of the IEEE International Conference on Intelligence and Security Informatics, pp. 119-124, June 2009.

[7] N. Ye, Secure computer and network systems: Modeling, analysis and design, London, UK: John Wiley & Sons Ltd, 2008.

[8] N. Ye, "QoS-centric stateful resource management in information systems," Information Systems Frontiers, vol. 4, no. 2, pp. 149-160, 2002.

[9] N. Ye, C. Newman and T. Farley, "A system-fault-risk framework for cyber attack classification," Information, Knowledge, Systems Management, vol. 5, no. 2, pp. 135-151, 2005.

[10] S. S. Yau, N. Ye, H. Sarjoughian, D. Huang, A. Roontiva, M. G. Baydogan, and M. Quqsith, "Towards development of adaptive service-based software systems," *IEEE Transactions on Services Computing*, Vo. 2, No. 3, pp. 247-260, 2009.

[11] F. M. Abdel-qader, "Examples to create your conferencing system in .NET, C# VOIP & video conferencing systems using H.323 and TAPI 3," 2007, retrieved on 02/15/2010 from http://www.codeproject.com/KB/IP/Video_Voice_Conferencing.aspx?fid=373359&df=90&mpp=25&noise=3&sort=Position&view=Quick&select=2645686.

[12] J. Daemen and V. Rijmen, "The Design of Rijndael: AES- The Advanced Encryption Standard," Secaucus, NJ: Springer-Verlag New York, Inc., 2001.

[13] A. Kirillov, "Motion detection algorithms," 2007, retrieved on 02/15/2010 from http://www.codeproject.com/KB/audio-video/Motion_Detection.aspx.

[14] Sourcefire. Snort v. 2.8.6, 2010, http://www.snort.org/.

[15] Mitre Corporation, "Common Attack Pattern Enumeration and Classification (CAPEC)", retrieved on 02/15/2010 from: http://capec.mitre.org/data/graphs/1000.html.

[16] M. Massimiliano. Cain & Abel Password Recovery software v4.9.30, http://www.oxid.it/.

[17] Tools4ever. Free Ping v2.0, http://www.tools4ever.com/.

[18] Nmap.org. Nmap ® v4.76, http://nmap.org/.

[19] Microsoft, Window Server 2003 performance counters reference, 2003. Retrieved on 02/15/2010 from: http://technet.microsoft.com/en-us/library/cc776490%28WS.10%29.aspx.

[20] D. C Montgomery, Design and Analysis of Experiments. Sixth edition, John Wiley & Sons, Inc, 2005.

[21] StatSoft, Inc. Statistica ® v7., http://www.statsoft.com/.

[22] S. Olejnik and J. Algina, "Generalized Eta and Omega Squared Statistics: Measures of effect size for some common research designs," Psychological Methods, vol. 8, no. 4, pp. 434-447, 2003.