An Adaptive Model for Tradeoff Between Service Performance and Security in Service-based Environments

Stephen S. Yau, Yin Yin and Ho G. An
School of Computing and Informatics, Arizona State University
{*yau, yin.yin, ho.an*}*@asu.edu*

Abstract

The messages-based communication among services in Service Oriented Architecture (SOA) is vulnerable to various security attacks, and has to be well protected by security mechanisms, which may sacrifice the service performance due to limited system resources. In this paper, an adaptive model for the tradeoff between service performance and security in service-based environments is presented. This model can be used to adjust security configurations to provide sufficient protection and satisfy service performance requirements for SOA-based software systems simultaneously.

The construction of the tradeoff model consists of the development of a set of metrics to quantitatively measure performance and security, the development of a tradeoff objective function incorporating service performance and security together, and the parameter estimation through experiments. A service-based secure voice communication service is developed as an example to show the construction of the tradeoff model.

1. Introduction

Service-oriented architecture (SOA) facilitates dynamic organization of a set of services to work seamlessly as a system, where each service communicates with other services through messages. Although message-based communication makes it easier to use services under different corporation firewalls, from a privacy perspective, this also poses increasing concern on the protection of such communication.

Current research on SOA security mainly focuses on the protection of services from malicious consumers through authentication and authorization [1-4], and the protection of messages through XML encryption and signature [5]. However, the impact of these security mechanisms on the systems' performance has not been addressed thoroughly. Security requirements are often in contrast with other performance requirements, like timeliness and throughput due to limited system resources. For example, a VoIP service has strict timeliness requirements [6]. However, any encryption algorithm for securing the voice data has to consume certain amount of CPU time and memory, and hence increases the package delay. Moreover, activity parameters of a service, such as the coding method of voice signal for the VoIP service, the key length for encrypting data transmissions and the percentage of packages encrypted, have significant impacts on resource consumptions.

In this paper, we will develop an adaptive model for the tradeoff between service performance and security for adaptable service-based systems (ASBS) [7]. Our tradeoff model can adjust the configurations and service operations of ASBS to balance the QoS and the security. The major distinction between our tradeoff model and most existing approaches [8-14] is that our model can compute the best tradeoff by minimizing a tradeoff objective function which incorporates service performance and security metrics together instead of intuitively switching from one security configuration to another until the desirable tradeoff is reached. The construction of our tradeoff model involves the development of a set of metrics to quantitatively measure performance and security and a tradeoff objective function incorporating both service performance and security metrics together so that we can consider performance and security simultaneously, and the estimation of our tradeoff model's parameters for performance and security metrics, which will be shown using a service-based secure voice communication service.

2. Related Work

With the increasing and diverse applications of service-based systems, it is important to know how to manage the resource for having better performance. Application level differentiated services [15, 16] control performance for different classes of service consumers. When the system resources are limited, fewer resources are allocated for normal consumers, and most resources are reserved for providing premium consumers' expected performance. Furthermore, when resources are extremely limited and even premium consumers' required performance cannot be fully satisfied, feedback controlled web services [10, 17] have been developed to adjust resource allocation to meet the most important performance, such as the allowable delay in real-time systems. These systems, however, do not consider security, and mainly focus on how to satisfy various performance requirements of consumers, rather than the control of the tradeoff between performance and security.

To control the tradeoff between performance and security, we first need to develop security metrics to facilitate the comparison of different security mechanisms, especially the security mechanisms for the same security functionality. Existing security metrics include qualitative metrics [12, 13] and quantitative metrics [8, 11]. Qualitative metrics classify various security mechanisms to several discrete levels, such as low, medium, and high. Security mechanisms in the higher level can provide better protection, but it is impossible to compare security mechanisms with the same security level. Furthermore, qualitative metrics are too coarse for fine control of the tradeoff between performance and security.

Compared with qualitative metrics, quantitative metrics generate a value for each security mechanism from its configurations, and hence are more accurate and can compare any two security mechanisms. However, the difference between the values generated by existing quantitative metrics cannot correctly reflect the difference between the security strengths of corresponding security mechanisms. For example, symmetric key encryption algorithms have different requirements on the key length from asymmetric key encryption algorithms. To achieve the same strength of security, the symmetric encryption algorithm AES needs a 256-bit key, but the asymmetric encryption algorithm RSA needs a 15,360-bit key, and elliptic encryption needs a 571-bit key [18]. Furthermore, the security strength is not absolute, but relative to the capability of attackers, which is also ignored in the existing quantitative metrics. To achieve the same strength of security, systems facing more powerful attackers have to use longer keys than systems facing weak attackers. This is why the National Security Agency suggests that 128-bit AES encryption is sufficient for SECRECT level data, but TOP SECRECT level data requires at least 192-bit AES encryption [19].

Because it is difficult to improve the encryption algorithms' efficiency, selective encryption is a comparably easier solution for reducing encryption algorithms' negative influence on performance. Selective encryption is motivated by the observation that the encryption of the most important part of the information may be sufficient to prevent attackers from knowing the whole information, especially for multimedia information. Hence, selective encryption can encrypt information as little as possible in order to greatly reduce the encryption's negative influence. However, current selective encryption algorithms are all developed for specific applications like image selective encryptions [20, 21] and video selective encryptions [22, 23]. There is no general selective encryption algorithm that can be used by our tradeoff model for any application. Hence, we need to develop another way to reduce encryption algorithms' negative influence on performance.

3. Performance and Security Metrics

As discussed in the introduction, while performance usually has natural metrics which are easy to be quantified, security is difficult to be quantitatively measured. In this section, we will first use the security configuration vectors to specify the security parameters, and then develop metrics for both performance and security.

3.1. Security Configuration Vector

A security configuration vector $SCV = \{F, A, l, p\}$ consists of the following four components,

- *Security Functionality F*. Security functionality specifies what protection the security mechanism provides, like confidentiality, integrity, and non-repudiation. Confidentiality prevents the attacker from knowing the content of packages, integrity prevents the attacker from modifying the content of packages, and the non-repudiation prevents the attacker from refusing the acceptance of packages.
- *Algorithm Name A*. Because there are multiple algorithms providing the same security functionality with different protection strengths, the algorithm name specifies which algorithm is used by the security mechanism. For confidentiality, the security mechanism can use DES or AES, and a security mechanism can provide the integrity of the protected information with SHA-1 Hash or Digital Signature.
- *Key Length l*. Another important factor that influences the protection strength is the key length. A longer key provides stronger protection but consumes more system resources. Hence, specifying key length in the security configuration vector gives the security mechanism more choices in the tradeoff between performance and security.
- *Protection Percentage p*. While research on cryptography and security has mainly focus on the development of semantic security mechanisms, which protect every bit of information, sometimes such extensive protection is unnecessary and unaffordable. Encrypting a part of packages, instead of encrypting all packages, will dramatically reduce the overhead introduced by security mechanisms and still prevent attackers from learning much information. Some selective encryption algorithms have been proposed for specific application, like image selective encryptions [20, 21] and video selective encryptions [22, 23]. Because we are developing a tradeoff model for all applications, the selective encryption is implemented as encrypting packages with probability *p*.

3.2. Performance Metric

Systems may have different performance requirements for different applications. In [6], the performance requirements of various types of network applications are specified and analyzed by considering both human factors and technology factors. For example, a voice communication application has strict timeliness requirements measured by delay and jitter, but has some tolerance of data loss and errors. Furthermore, the performance of SBS is usually measured from various aspects, such as timeliness, throughput, used bandwidth, allocated memory, and occupied CPU percentage. A performance model has been developed to analyze the interrelations among these different performance requirements and different performance aspects using non-binary classification and regression trees on experimental data, which can predicate the value of a performance aspect from other performance aspects in [7].

We are focusing, in this paper, on the relation between performance and security, i.e., security configuration vectors' influences on the performance. For simplicity, we only take the average delay of packages *D* and package traffic *T* as an example to show how to develop the metric to relate *D* and *T* to the security configuration vector $SCV = \{F, A, l, p\}$. The *delay D* is defined as the average time required by the security mechanism to take in a package, protect the package with the parameters specified in *SCV*, and return the package back. The *traffic T* is defined as the number of packages handled by the security mechanism per second. There may be various ways to define the functions of *D*, *T*, *SCV*, but all functions should have the following properties:

- The delay *D* consists of the delay of receiving and sending packages, and the delay of protecting packages. When the first part of the delay involves all incoming packages, the second part of the delay only involves packages that are protected. When the current traffic is *t* and the protection percentage specified in *SCV* is *p*, the number of protected package is *tp*.
- When the current traffic *t* is 0, the delay D is also 0 for any *SCV*. Hence, we have $D(SCV, 0) = 0$.
- When the current traffic *t* is very small but not 0, the security mechanism needs to wait for the packages coming in, and hence has sufficient time to handle every package. In this case, the delay *D* is mainly determined by the parameters in *SCV*. Thus, if the traffic $0 < t \leq T_1$, we have $D(SCV, t) = D_1(SCV)$, where $D_1(SCV)$ is a constant related to *SCV*.
- When the current traffic *t* becomes larger and exceeds T_1, the security mechanism does not have sufficient system resources to protect a package on time and send it out before the next package comes in. In this case, when one thread is keeping in protecting and sending the package, another thread is required to accept the new incoming package. These two threads will compete for system resources, and increase *D*. Hence, if $T_1 < t_1 < t_2$, we have $D(SCV, t_1) < D(SCV, t_2)$.
- The lager the current traffic *t*, the faster *D* increases as *t* increases. Hence, if $T_1 < t_1 < t_2$, $\partial D(SCV, t_1)/\partial t_1 > \partial D(SCV, t_2)/\partial t_2 > 0$.
- When the current traffic *t* keeps increasing and the security mechanism's system resources have been exhausted, the security mechanism starts to drop packages. That is, there is an upper limit $T_2(SCV)$ of the traffic. Hence, the range of traffic is $0 \leq t \leq T_2(SCV)$.
- For a constant traffic *t*, more efficient algorithm *A*, shorter key length *l*, or smaller protecting percentage *p* leads to smaller *D*. However, the effect of the combination of different *A*, *l*, and *p* needs to be further investigated through experiments.

Based on the above observation, we have the following function as the metric of delay, which is simple and satisfies all above properties.

$$D(SCV, t) = \begin{cases} 0, & if\ t = 0 \\ a + a'p, & if\ 0 < t \leq T_1 \\ a + a'p + be^t + b'e^{tp}, & T_1 < t < T_2 \end{cases} \quad (1)$$

where *p* is the protection percentage, a, b, T_1, T_2 are four parameters related to *SCV* but independent from *t*. All parameters will be estimated and validated by experiments.

Figure 1 shows two metrics of delay for two security configuration vectors SCV_1 and SCV_2, which have the same encryption algorithm and key length, but different protection percentage p_1 and p_2. Because $p_1 > p_2$, the security mechanism using SCV_2 will encrypt more packages with the same traffic than SCV_1, and hence the metric of delay with SCV_2 has larger parameters T_1 and T_2 and increases faster with traffic.

3.3. Security Metric

For a security configuration vector $SCV = \{F, A, l, p\}$ the security is measured as the attacker's probability of cracking a protected package with one attack attempt. First, if the package is not protected, that package can be cracked trivially with probability 1. Second, even when the package has been protected, the package can still be cracked with certain probability because there is no perfect protection. Suppose the probability of cracking a package protected by a security configuration vector is $v(l)$ and the attacker's capability is *c* (i.e., the attacker can attack the package *c* times every second), where *l* is the key length. Hence, in one second, the probability that the attacker can

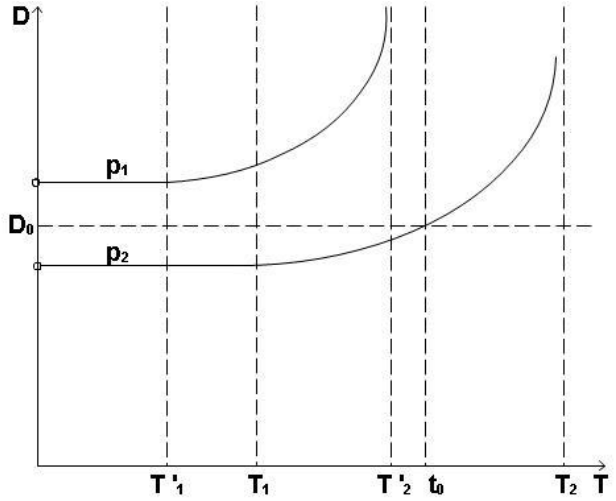

Figure 1. The relations between delay *D* and traffic *T* for two security configuration vectors with the same algorithm and key length, but different protection percentages p_1 and p_2, where $p_1 > p_2$. D_0 is the minimum delay requirement.

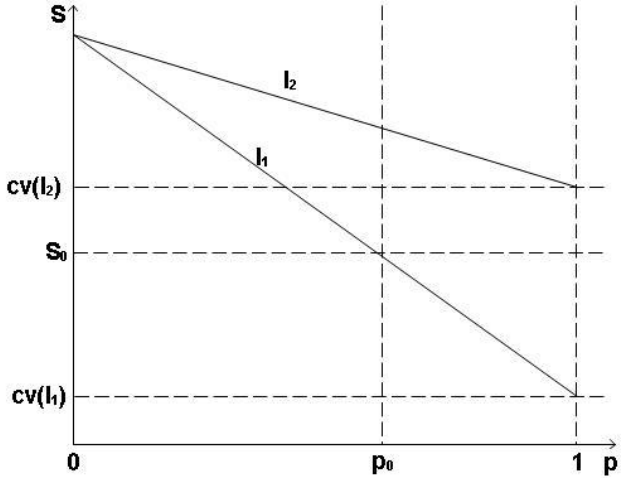

Figure 2. The relations between security and protection percentage for the same algorithm with two different key lengths l_1 and l_2, where $l_1 > l_2$. S_0 is the minimum security requirement.

successfully crack a protected package is *cv*(*l*). The overall security can be measured as

$$S(l, p, c) = (1 - p) + pcv(l) \qquad (2)$$

The parameter *v*(*l*) depends on the design of the security mechanism and evolution of attacking techniques. For AES encryption algorithm and brutal force attack, $v(128) = 2^{-127}$, $v(192) = 2^{-191}$, and $v(256) = 2^{-255}$. For RSA encryption algorithm and the general number field sieve attack (GNFS) [24], the most efficient integer factoring algorithm for integers with more than 100 bits, $v(768) = 0.93 \times 10^{-23}$, $v(1024) = 0.77 \times 10^{-26}$, and $v(2048) = 0.65 \times 10^{-33}$.

The parameter *c* is the computing power of the attacker encountered by the security mechanism, which can be estimated by the sensitivity of the information handled by the security mechanism. First, the capabilities of potential attackers for military systems and for commercial systems are different. Second, if the security mechanism is handling information that should be protected for a very long period, the security mechanism needs to estimate the capability of attackers in the future.

For the same algorithm with two different key lengths $l_1 > l_2$, Figure 2 shows two relations between the security metric and the protection percentage. With the longer key length l_1, the same algorithm can reach the same security level with a smaller protection percentage.

4. Adaptable Tradeoff between Performance and Security

The performance and security metrics allow us to quantitatively calculate how much protection a security configuration vector can provide and how much performance will be decreased by that security configuration vector. Hence, we can make and control the tradeoff between performance and security by adjusting the parameters in the security configuration vector. In this section, we will discuss three different tradeoff strategies to maximize performance, maximize security, and balance performance and security according to users' preferences respectively. For the first two strategies, although they are trying to maximize one aspect, either performance or security, it does not mean that the other aspect will be ignored. Any system should have minimum requirements on both performance and security, and any tradeoff strategies should first satisfy such minimum requirements and then do the tradeoff.

In this section, we will first discuss how to check whether the minimum requirements of performance and security can be satisfied, and then discuss the tradeoff strategies.

4.1. Minimum Requirement Validation

Basically, the tradeoff between performance and security is implemented through resource allocation. First, the SBS has to allocate certain amounts of resources for both performance and security to satisfy their minimum requirements. Then, if there are more resources, the SBS can allocate the available resources for better performance or better security. Hence, to check whether the tradeoff is possible, we first need to make sure that the minimum performance and security requirements can be satisfied.

As in the development of performance metrics, we also use delay as an example here. Assume that the SBS

requires that the delay should be less than D_0, the traffic will be up to T_0, and the success probability of an attacker with capability c should be less than S_0. First, from the minimum security requirement, we have

$$S(l,p,c) \le S_0 \Rightarrow p \ge (1-S_0)/(1-cv(l)) \quad (3)$$

Then, to satisfy the minimum performance requirement, from the metric of delay (1), we have

$$\begin{cases} p \le (D_0 - a)/a', & if\ T_0 \le T_1 \\ p \le ln((D_0 - a - be^{T_0})/b')/T_0, & if\ T_1 < T_0 \le T_2 \end{cases} \quad (4)$$

Hence, to check whether the minimum performance and security can be satisfied, we only need to check whether there exists algorithms and key lengths l satisfying

$$\frac{1-S_0}{1-cv(l)} \le \begin{cases} (D_0 - a)/a', & if\ T_0 \le T_1 \\ ln((D_0 - a - be^{T_0})/b')/T_0, & if\ T_1 < T_0 \le T_2 \end{cases}$$

All parameters, including $a, a', b, b', v(l), T_1$ and T_2, are determined by the security algorithm and key length l. Because any SBS only supports a limited number of security algorithms and key lengths, we can check whether both the minimum performance and security requirements can be satisfied by enumerating all supported security algorithms and key lengths to see if the above condition can be satisfied.

For example, Figure 1 shows that the *SCV* with the protection percentage p_1 cannot satisfy the minimum performance requirement D_0. Figure 2 shows that the *SCV* with the key length l_2 cannot satisfy the minimum security requirement S_0.

4.2. Tradeoff Objective Function

When both minimum performance and security requirements are satisfied, the SBS can use the available resources for better performance or security. To have better security, as shown in the security metric (2), the SBS can use either a stronger algorithm with a longer key, or a larger protection percentage. On the other hand, the performance metric (1) shows that a stronger algorithm with a longer key will generate larger parameters a, a', b, b' and smaller T1 and T2, and a larger protection percentage will make the delay increases faster with the traffic. Both of these will reduce the performance. Hence, to control the tradeoff between performance and security, we have to combine the performance metric and security metric together as a tradeoff objective function. The tradeoff objective function between the performance metric (1) and the security metric (2) can be defined as

$$G(SCV,t) = aD(SCV,t) + bS(l,p,c) \quad (5)$$

where $SCV = \{F, A, l, p\}$, a and b are two weighting factors representing the SBS consumer's preferences on performance and security, respectively. To normalize the function G, we assume that $a + b = 1$ and $0 \le a, b \le 1$.Then, to compute the optimized tradeoff between D and S, the SBS is to minimize the value of G with constraints $D(SCV, T_0) \le D_0$ and $S(l,p,c) \le S_0$.

4.2.1. Performance Biased Objective Function. The performance biased objective function is a tradeoff objective function that tries to maximize performance without violating the minimum security requirement. For the tradeoff objective function (5), the performance biased objective function sets the weighting factor b of the security to 0. In this case, to minimize the tradeoff objective function is equivalent to minimizing the delay D.

When the encryption algorithm and the key length are fixed, the performance biased tradeoff should always use the minimum protection percentage $(1-S_0)/(1-cv(l))$ computed in (3).

4.2.2. Security Biased Tradeoff Function. The security biased tradeoff function is a tradeoff objective function that tries to maximize security without violating the minimum performance requirements. For the tradeoff objective function (5), the security biased tradeoff function sets the weighting factor a of the performance to 0. In this case, to minimize the tradeoff objective function is equivalent to minimizing the attacker's success probability S.

When the encryption algorithm and the key length are fixed, we can compute the upper limit for the protection percentage from the minimum performance requirements as shown in (4), which represents the possible maximum security.

4.2.3. Non-linear tradeoff objective function. If the SBS consumer's preferences on performance and security, i.e., the weighting factors a and b, do not change with the real-time performance and security conditions, we call such a tradeoff objective function as a *linear tradeoff objective function*, like the performance biased tradeoff function and the security biased tradeoff function. On the contrary, if the weighting factors a and b are related to the current performance and security, we call such a tradeoff objective function as a non-linear tradeoff objective function. There may be various ways to define a non-linear tradeoff model, but all definitions should have the following properties:

- The weighting factor of performance (security) increases when the performance (security) approaches the minimum performance (security) requirement.
- The minimum performance or security requirements are critical for the SBS. Hence, when the minimum

performance (security) requirements are not satisfied, the weighting factor of performance (security) becomes infinite.

Thus, a possible *non-linear tradeoff objective function* for the performance metric (1) and the security metric (2) can be defined as follows:

$$G'(SCV,t) = e^{a'/(D_0-D(SCV,t))}D(SCV,t) + e^{b'/(S_0-S(SCV,t))}S(l,p,c) \quad (6)$$

where a', b' are two constants. The weighting factors of delay and security will increase faster with larger delay and security, and will become infinite when the minimum delay and security requirements are not satisfied.

5. Parameter Estimation and Experimental Results

In this section, we will develop an adaptable service-based secure voice communication system as an experimental scenario to show how to build our tradeoff model between delay and the confidentiality of packages. In this scenario, there is one voice communication service to provide real-time voice data service for multiple clients simultaneously, and a security service to provide package encryption. For secure voice communication, the voice communication service can call the security service to encrypt packages before sending the packages to clients.

We have implemented the voice communication service by converting an open-source video conferencing software package [25] in a web service using C# in .NET environment. The voice communication service supports three sampling rates (132330, 176400, 220500), and three client numbers (1, 3, 5), which together generate nine different settings, called *traffic levels* for the voice communication service. The security service is implemented with the Cryptography library provided by C#. To make the complexity of the experiment controllable, the security service only supports the AES encryption, but has three key lengths (128-bits, 192-bits, 256-bits) and four encrypting percentages (25%, 50%, 75%, 100%). Hence, there are twelve different security configuration vectors for the security service. For the whole adaptable service-based secure voice communication system, there are total 108 settings generated from nine traffic levels and twelve security configuration vectors.

Besides these two services in this experiment, we also have developed a QoS monitor with Windows Performance Objects, which collects the average delay of each package. In this experiment, we run all 108 settings of the voice communication system and collect all performance data from the QoS monitor. The collected performance data is analyzed to estimate the parameters for the tradeoff model between the delay and the confidentiality of packages. In particular, we have studied how the security configuration vector affects the delay of packages under different traffic levels.

Table 1. Experimental data for 128-bit key length and 25% protection percentage.

# Package In	# Packages Out	Dropping Rate (%)	Avg. Delay (seconds)
12123	12123	0.00%	0.008
15981	15981	0.00%	0.007
20181	20181	0.00%	0.007
36369	36369	0.00%	0.007
47943	47943	0.00%	0.009
60543	60540	0.00%	0.01
60615	60605	0.02%	0.013
79905	79755	0.19%	0.014
100905	100550	0.35%	0.014

Our experimental data on the relations between delay and traffic for all possible key lengths and protection percentages are shown in Figure 3, where all the data can be roughly grouped into four groups. For each group, there are three sets of data with the same protection percentage but different key lengths, which are coincide with each others. This observation implies that

- The key length has no significant effect on the relation between delay and traffic. That is, the parameters *a*, *a'*, *b*, *b'*, T_1 and T_2 of the delay metric (1) are independent of the key length.
- The protection percentage dominates the relation between delay and traffic. With the same traffic, a larger protection percentage generates longer delay than a smaller protection percentage *p*. That is, the parameters *a*, *a'* of the delay metric (1) are larger with a larger *p*.
- When the traffic increases, larger *p* makes the delay increases faster than smaller *p*. That is, the parameters *b*, and *b'* of the delay metric (1) are larger with a larger *p*.
- For different *p*, the delay starts to increase exponentially at the same traffic. That is, the parameters T_1 and T_2 of the delay metric (1) are independent of *p*.

Table 1 lists our experimental data on 128-bit key length and 25% protection percentage. The first column is the number of packages taken by the system. The second column is the number of packages sent out by the system. When the number in the second column is smaller than the number in the first column, it means that the system has dropped some packages due to limited resources. The dropping rate is shown in the third column. The last column is the average delay of each package. From Table l, we can see that 1) when the incoming traffic does not

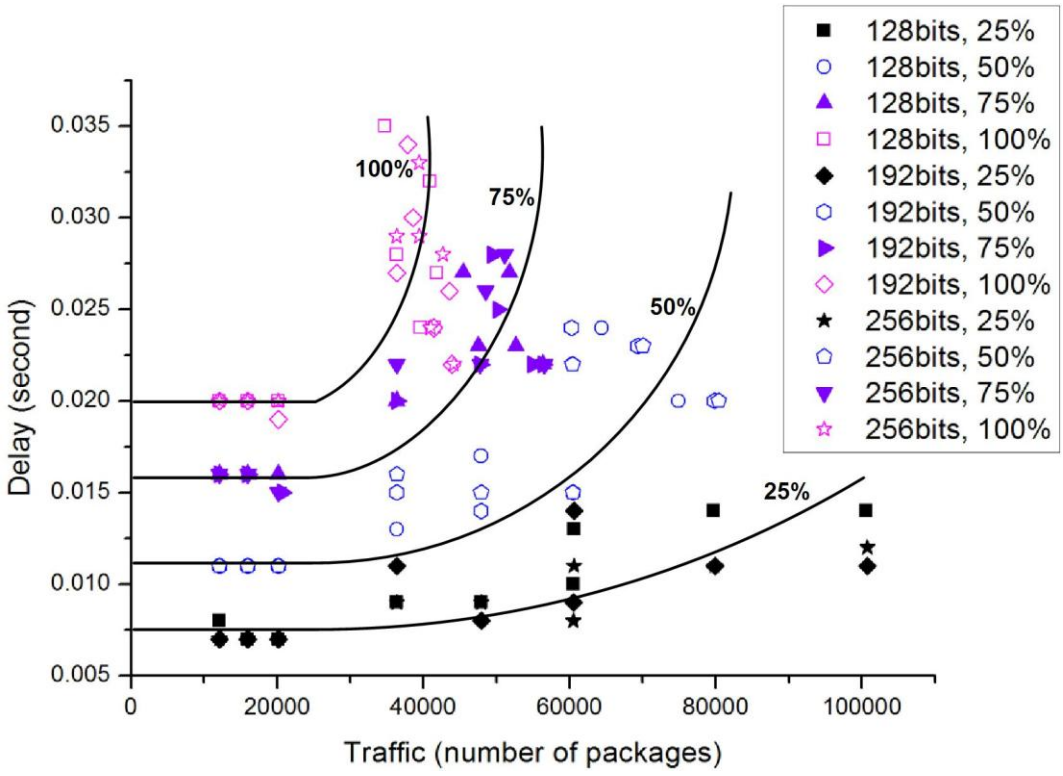

Figure 3. The relations between traffic and delay for different key lengths and protection percentages.

exceed 36369, the average delay is stable at 0.007 second, 2) when the incoming traffic exceeds 36369, the average delay starts to increase, and 3) when the incoming traffic exceeds 60543, the system starts to drop packages. Hence, for the security configuration vector SCV_1=(*Confidentiality, AES, 128 bits, 25%*), we can estimate the delay metric (1)'s parameters as $T_1 = 36369$, $T_2 = 60543$. Based on our experimental data, the delay metric for SCV_1 can be estimated as follows:

$$D(SCV,t) = \begin{cases} 0, & t = 0 \\ 0.007, & 0 < t \le 36369 \\ 0.007 + 0.0009 \times e^{2\times 10^{-5} t}, & 36369 < t < 60543 \end{cases}$$

To have a more accurate estimate and validate the above estimation, we need to run the experiment under more traffic levels, which are currently being done.

6. Conclusions and Future Research

In this paper, we have presented an adaptable tradeoff model between service performance and security in service based environments. We have developed quantitative performance and security metrics, and combined them together as a tradeoff objective function with two weighting factors. Because these two weighting factors represent the preferences on performance and security, we have discussed how to achieve the maximum performance, the maximum security, and the optimized balance between performance and security by adjusting the weighting factors. We also have implemented a secure voice communication scenario to show how to estimate the parameters for the tradeoff model.

Future research along this line includes extending the tradeoff model to make tradeoff among a set of performance aspects and security simultaneously, and conduct extra experiments to study the security metric when several different security algorithms are involved.

Acknowledgement

This work is supported by National Science Foundation under grant number CCF-0725340. The authors would like to thank Dazhi Huang for many valuable discussions.

References

[1]. S. Godik and T. Moses, "eXtensible Access Control Markup Language (XACML) Version 1.0", OASIS Standard, February, 2003.

[2]. P. Hallam-Baker and S. Mysore, "XML Key Management Specification (XKMS 2.0)", W3C Recommendation, http://www.w3.org/TR/xkms2, 2005.
[3]. S. Bajaj, D. Box, D. Chappell, F. Curbera, G. Daniels, P. Hallam-Baker, M. Hondo, C. Kaler, D. Langworthy and A. Malhotra, "Web Services Policy Framework (WS-Policy)", Technical report, BEA Systems Inc., Intnational Business Machines Corporation, Microsoft Corporation, Inc., SAP AG, Sonic Software, and VeriSign Inc., September, 2004
[4]. A. Pashalidis and C.J. Mitchell, "A Taxonomy of Single Sign-On Systems", *Proc. Info. Security and Privacy*, pp. 249-264, 2003.
[5]. M. Mactaggart, "Enabling XML Security: An Introduction to XML Encryption and XML Signature", *IBM Developer Works*, vol. 9, 2001.
[6]. Y. Chen, T. Farley, and N. Ye, "QoS Requirements of Network Applications on the Internet," *Info., Knowledge, Systems Management*, vol. 4, no. 1, 2004, pp. 55-76.
[7]. S. S. Yau, N. Ye, H. Sarjoughian, and D. Huang, "Developing Service-based Software Systems with QoS Monitoring and Adaptation," *Proc. 12th Int'l Workshop on Future Trends of Distributed Computing Systems (FTDCS 2008)*, 2008, pp.74-80.
[8]. D. Lie and M. Satyanarayanan, "Quantifying the Strength of Security Systems," *Proc. of 2nd USENIX Workshop on Hot Topics in Security*, 2007, pp. 1-6.
[9]. W. Yurcik, C. Woolam, G. Hellings, L. Khan, and B. Thuraisingham, "SCRUB-tcpdump: A Multi-level Packet Anonymizer Demonstrating Privacy/Analysis Tradeoffs," *Proc. of 3rd Security and Privacy in Communications Networks and the Workshops (SecureComm)*, 2007, pp. 49-56.
[10]. C. Lu, Y. Lu, T.F. Abdelzaher, J.A. Stankovic, and S.H. Son, "Feedback Control Architecture and Design Methodology for Service Delay Guarantees in Web Servers," *IEEE Trans. on Parallel and Distributed Systems*, vol. 17, no. 9, 2006, pp 1014-1027.
[11]. K. Kang and S. Son, "Systematic Security and Timeless Tradeoffs in Real-Time Embedded Systems," *Proc. 12th IEEE Int'l Conf. on Embedded and Real-Time Computing Systems and Application*, 2006, pp. 183-189.
[12]. S.H. Son, R. Zimmerman, and J. Hansson, "An Adaptable Security Manager for Real-Time Transactions," *Proc. 12th Euromicro Conf. on Real-Time Systems*, 2000, pp. 63-70.
[13]. E. Spyropoulou, T. Levin, and C. Irvine, "Calculating Costs for Quality of Security Service," *Proc. 16th Annual Conf. on Computer Security Applications*, 2000, pp. 334-343.
[14]. S. S. Yau, M. Yan, and D. Huang, "Design of Service-based Systems with Adaptive Tradeoff between Security and Service Delay," *Proc. 4th Int'l Conf. on Autonomic and Trusted Computing*, 2007, pp. 394-401.
[15]. L. Eggert and J. Heidemann, "Application-Level Differentiated Services for Web Services," *J. World-Wide Web*, vol. 2, no. 3, 1999, pp. 133-142.
[16]. G. Rao and B. Ramamurthy, "DiffServer: Appliction Level Differentiated Services for Webservers," *Proc. IEEE Int'l Conf. on Communication*, vol. 5, 2001, pp. 1633-1637.
[17]. T.F. Abdelzaher, J.A. Stankovic, C. Lu, R. Zhang, and Y. Lu, "Feedback Performance Control in Software Services," *IEEE Control Systems Magazine*, vol. 23, no. 3, 2003, pp. 74–90.
[18]. A. Lenstra, E.Verheul, "Selecting Cryptographic Key Sizes," *Jour. of Cryptology*, vol 14, 2001, pp. 255-293.
[19]. L. Hathaway, "National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information", National Security Agency, 2003.
[20]. H. Cheng and X. Li, "Partial Encryption of Compressed Images and Video," *IEEE Trans.. on Signal Processing*, vol. 48, no. 8, 2000, pp. 2439-2451
[21]. M.V. Droogenbroeck and R. Benedett, "Techniques for a Selective Encryption of Uncompressed and Compressed Images," *Proc. of Adv. Concepts for Intelligent Vision Systems (ACIVS)*, 2002, pp. 9-11.
[22]. C. Shi, S.Y. Wang, and B. Bhargava, "MPEG Video Encryption in Real-Time Using Secret key Cryptography," *Proc. Int'l Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, 1999.
[23]. W. Zeng and S. Lei, "Efficient Frequency Domain Selective Scrambling of Digital Video," *IEEE Trans. on Multimedia*, vol. 5, no. 1, 2003, pp. 118-129.
[24]. C. Pomerance, "A Tale of Two Sieves," *Notices of the AMS*, vol. 43, no. 12, 1996, pp. 1473–1485.
[25]. F. Abdelqader, "Examples to Create Your Conferencing System in .NET, C# VOIP & Video Conferencing Systems using H.323 and TAPI 3", available at http://www.codeproject.com/KB/IP/Video_Voice_Conferencing.aspx