ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.6, June 2007, pp.1522−1530
DOI: 10.1360/jos181522
© 2007 by *Journal of Software*. All rights reserved.

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于参数优化批处理的 TLS 协议*

齐 芳[1,2], 贾维嘉[1,3+], 鲍 丰[2], 吴永东[2], 王国军[1]

[1](中南大学 信息科学与工程学院,湖南 长沙 410083)

[2](Agency for Science, Technology and Research, Institute for Infocomm Research, Singapore 119613, Singapore)

[3](Department of Computer Science, City University of Hong Kong, Kowloon Tong, Kowloon, Hong Kong)

Parameter Optimization-Based Batching TLS Protocol

QI Fang[1,2], JIA Wei-Jia[1,3+], BAO Feng[2], WU Yong-Dong[2], WANG Guo-Jun[1]

[1](School of Information Science and Engineering, Central South University, Changsha 410083, China)

[2](Agency for Science, Technology and Research, Institute for Infocomm Research, Singapore 119613, Singapore)

[3](Department of Computer Science, City University of Hong Kong, Kowloon Tong, Kowloon, Hong Kong)

+ Corresponding author: Phn: +86-731-8836152, Fax: +86-731-8877711, E-mail: itjia@cityu.edu.hk

Qi F, Jia WJ, Bao F, Wu YD, Wang GJ. Parameter optimization-based batching TLS protocol. *Journal of Software*, 2007,18(6):1522−1530. http://www.jos.org.cn/1000-9825/18/1522.htm

Abstract: The primary goal of the Transport Layer Security (TLS) protocol is to provide confidentiality and data integrity between two communicating entities. Since the most computationally expensive step in the TLS handshake protocol is the server's RSA decryption, it is introduced that optimal batch RSA can be used to speedup TLS session initialization. This paper first indicates that the previous batch method is impractical since it requires a multiple of certificates, then it proposes the unique certificate scheme to overcome the problem. It is also introduced that the batching parameter is optimized when integrating users' requirements for Internet Quality of Service (QoS). To select the optimal batching parameters, not only the server's performance but also the client's tolerable waiting time is considered. Based on the analysis of the mean queue time, batching service time and the stability of the system, a novel batch optimal scheduling algorithm which is deployed in a batching Web server is proposed. Finally, the proposed algorithm is evaluated to be practical and efficient through both analysis and simulation studies.

Key words: transport layer security; batch RSA; TLS handshake; mean response time; tolerable waiting time

摘 要: TLS(transport layer security)协议的基本设计目标是为两个通信实体之间提供数据的保密性和完整性.由于在传输层安全握手协议中最耗费计算资源的步骤是服务器 RSA 解密运算,优化的批处理的 RSA 方法提出可以用于加速 TLS 会话的初始化.首先指出了以前的批处理方法由于要求多证书实现而实用性不强.然后提出了单一证书策略的方法,从而克服了这一问题.还提出结合用户对于因特网服务质量的要求优化了批处理参数.为了选择优化的

* Supported by the National Natural Science Foundation of China under Grant No.60503007 (国家自然科学基金); the CityU Research Project (APR) 9610027; the National Basic Research Program of China under Grant No.2003CB317003 (国家重点基础研究发展计划(973)); the Program for New Century Excellent Talents in University of the Chinese Ministry of Education under Grant No.NCET-06- 0686 (2006 年度教育部"新世纪优秀人才支持计划")

Received 2005-11-01; Accepted 2007-01-26

批处理的参数,不仅考虑了服务器的性能,而且还考虑了客户可容忍的等待时间.通过分析并在阐述平均排队时间、批处理服务时间和系统稳定性的基础上提出了一种新颖的优化批处理调度算法,已部署在服务器上.最后通过分析和模拟两种方法验证了所提出方案的实用性和有效性.

关键词: 传输层安全协议;批处理 RSA;TLS 握手协议;平均响应时间;可容忍的等待时间
中图法分类号: TP309 文献标识码: A

1 Introduction

Secure communications such as Internet banking and e-commerce are an intrinsic demand of today's world of online transactions. TLS protects communications by encrypting messages with a secret key negotiated in the TLS handshake protocol. Such a protocol allows the server and the client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before transmitting and receiving the first byte of data[1]. However, such a protocol needs intensive computational resource due to the cost of public-key operations.

The proposed scheme in this paper focuses on the batching technology which is a software-only approach for speeding up TLS's performance on a web server. Starting-point of the proposed scheme is a technique due to Fiat[2] for batch RSA decryption. Shacham and Boneh proved that it is impossible to use a single certificate in the present TLS system[3]. Whereas, this paper adapts the certificate mechanism so as to provide TLS setup with unique certificate issued by Certificate Authority (CA). Batch size is equal to four in the scheme of Shacham and Boneh[3]. However, it ignores the satisfaction of the users' requirements for Quality of Service such as tolerable waiting time. Tolerable waiting time is defined as the delay time a client can tolerate between a request for a secure web page and receiving the page. In addition, the proposed scheme in this paper models the client request as an M/D/1 queue[4] and uses approximate analytical solution of mean response time to optimize the batch size of the server. Consequently, the proposed optimal batch scheduling algorithm is employed in a batching web server.

The rest of the paper is organized as follows. Section 2 describes the existing batch method with Batch RSA and describes its problems. The proposed unique certificate scheme in batching TLS is presented in Section 3. The analytical model of parameter optimization in batching TLS handshake is presented in Section 4. Section 5 validates the solutions through both analysis and simulation studies.

2 Preliminaries

Definition 1. Given b distinct and pairwise relatively prime public keys $e_1,...,e_b$ all sharing a common modulus $N=pq$, relatively prime to $\phi(N)=(p-1)(q-1)$. n is the bit length of the public modulus N and k the bit length of the bigger of e_i. Given plaintext messages $m_1,...,m_b$. Furthermore, we have b encrypted messages $v_1,...,v_b$ (i.e. $v_i = m_i^{e_i} \bmod N$) one encrypted with each key. The multiplication computation phase is to generate the product $v=\prod_{i=1}^{b} v_i^{e/e_i} \bmod N$, where $e=\prod_{i=1}^{b} e_i$. The exponentiation phase yields $v^{1/e} \bmod N = \prod_{i=1}^{b} v_i^{1/e_i} \bmod N$, which is stored as m. The division computation phase is to break up the product m to obtain the plaintexts $m_i = v_i^{1/e_i}$ which we wish to decrypt simultaneously. This procedure is called Batch RSA[2].

Example. Fiat observed that when using small public exponent e_1 and e_2, it is possible to decrypt two cipher texts for approximately the price of one[3]. Suppose v_1 and v_2 is a cipher text obtained by encryption using the public key $(N,3)$ and $(N,5)$. Then the product $m=(v_1^5 \cdot v_2^3)^{1/15} \bmod N$ can be computed according to Definition 1. To decrypt m_1 and m_2, we must compute $v_1^{1/3}$ and $v_2^{1/5}$ mod N as follows:

$$v_1^{1/3} \bmod N = \frac{m^{10}}{v_1^3 \cdot v_2^2},\ v_2^{1/5} \bmod N = \frac{m^6}{v_1^2 \cdot v_2} \tag{1}$$

Shacham and Boneh showed that it is not possible to batch when the same public key is used for more than one message[3]. They provide a multiple certificate to implement the batch method by assigning the different public key in TLS handshake. The method has following disadvantages: Requirement for many different RSA certificates; additional payment for certificates; extra maintenance works of multiple certifications. The proposed unique certificate scheme solves the impractical of multiple certificate method in next section.

3 Unique Certificate Scheme in Batching TLS Handshake

In the standard TLS protocol, each client encrypts a 48-byte pre-master secret using e_i as the encryption exponents, and the server decrypts the cipher text independently so as to get the *Pre-master secret*. But batch RSA obtains the *Pre-master secrets* from multiple clients and hence improves the performance significantly.

Our unique certificate method is to reuse the message *ServerHello.random* in the protocol (see Fig.1). For simplicity, we only show the related processes and the modified information in the standard TLS handshake protocol. The following procedure is the unique certificate scheme for SSL handshake protocol: (1) Clients send "*Client hello*" massage that includes the cipher suits to the server and create random nonce r_c respectively. (2) Server responds with a "server hello" massage that includes server's public-key certificate and a random nonce $r_{s.}$ In this improvement, e_i is actually a part of *ServerHello.random*. Server only needs to send unique certificate to all the clients. (3) Clients choose a secret random 48-byte *Pre-master secret* m_1 and m_2 by inputting values m_1, m_2, r_c, r_s into hash function $f()$. It then encrypts m_i with e_i which is different from server's public-key and attaches the ciphertext to a "*Client key exchange*" message that is sent to server. (4) Server decrypts the *Pre-master secret* m_1 and m_2 simultaneously using batch RSA, and uses it to compute the *Shared master secret* s_1 and s_2 respectively.

The client will verify the certificate as usual, but encrypt the *pre-master secret* with received e_i instead of the public exponent in the certificate. Therefore, no extra charge is required, and it is easy to manage the certificate. On the other hand, since the certificate is used to prove the owner who knows the factors of the RSA moduli N only, this adaptation does not undermine the security strength of TLS protocol.

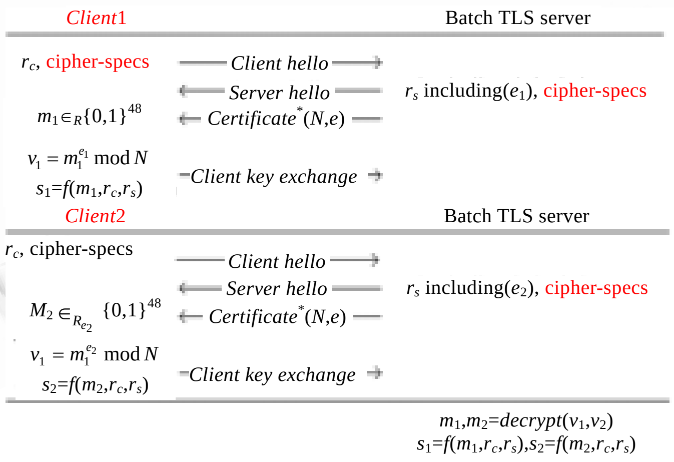


Fig.1 Unique certificate scheme

4 Analytical Model of Parameter Optimization in Batching TLS Handshake

In this section, analytical model of parameter optimization is constructed. The proposed scheme models the client request as an M/D/1 queue and uses approximate solution of mean response time to optimize the batch size of the server. Consequently, based on the analysis of the stability of the system, the proposed optimal batch scheduling algorithm is employed in a batching Web server.

4.1 Analytical mean queue time in batching queue model M/D/1

Suppose the client arrival process is Poisson distributed with an arrival rate, and the server response is regarded as an M/D/1 queue. Let the batching service time be τ which is determined by the batching size b.

The M/D/1 model can be approximated with a semi-Markov process (see Fig.2). The state of semi-Markov process is described by (i) where i indicates the number of clients waiting in the queue with i=idle and idle indicates the server is idle. Let the mean residual service time be $T_r=0.5\tau$[4].

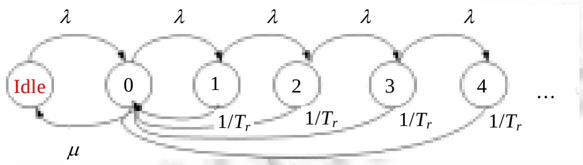


Fig.2 Semi-Markov process model of batching server

Let A_0 be the mean holding time in state idle, A_1 be the mean holding time in state (0), and A_2 be the mean holding time in state (i), for i>0. Thus

$$A_0=\int_0^\infty t\lambda e^{-\lambda t}\mathrm{d}t=1/\lambda\ ,\ A_1=\int_0^\tau t\lambda e^{-\lambda t}\mathrm{d}t+\int_\tau^\infty \lambda e^{-\lambda t}\mathrm{d}t=\eta_1/\lambda\quad A_2=\int_0^{T_r} t\lambda e^{-\lambda t}\mathrm{d}t+\int_{T_r}^\infty \lambda e^{-\lambda t}\mathrm{d}t=\eta_2/\lambda \tag{2}$$

where, $\eta_1=1-e^{-\lambda\tau}$ denotes the Markov transition Probability from idle to state (0), and $\eta_2=1-e^{-0.5\lambda\tau}$ denotes the Markov transition state (i) to state (i+1), for i>0.

Let π_0^* corresponds to the state (0). Then, the steady distribution π_0^* for semi-Markov can be solved as $\pi_0^*=(1-\eta_1)(1-\eta_2)/(1+\eta_1\eta_2-\eta_2)$ due to Ref.[5] (see Fig.2). *Prob*(*idle*) is approximated by π_0^*, which is obtained by solving the semi-Markov processed. Then the mean queue time with batching T_q is solved using the residual service time T_r and the steady state distribution π_0^* for semi-Markov when the server is idle.

$$T_q=(1-Prob(idle))T_r=(1-\pi^*)T_r=(1-(1-\eta_1)(1-\eta_2)/(1+\eta_1\eta_2-\eta_2))T_r \tag{3}$$

4.2 Analytical batching service time

Let the batching RSA decryption time in TLS handshake time be T_b. Let n be the bit length of the public modulus N and k be the bit length of the bigger of exponent. Because the public exponent e_i is chosen as small as possible to make auxiliary exponentiations cheap, the low-cost operations in the computation cost estimation can be ignored.

Asymptotic behavior of analysis of batch RSA can be estimated as $3n^3+(42b+k(3b^3+3b)-1)n^2+o(n^2)$[6].

The classic RSA's decryption mainly includes exponentiation computation which costs $3n^3+n^2+o(n^2)$.

It is assumed that the classic RSA decryption time is computed in advance and denoted as T_{rsa} with 1024 bits public modulus N and the exponent e 65537. The batching RSA decryption time in TLS handshake T_b can be estimated as the following.

As a result, the batching RSA decryption time is

$$T_b=\left(\frac{3n^3+n^2(42b+k(3b^3+3b)-1)}{b(3n^3+n^2)}\right)bT_{rsa}=\left(\frac{3n+42b+k(3b^3+3b)-1}{b(3n+1)}\right)bT_{rsa} \tag{4}$$

Since T_b is the majority of service time, the batching service time of the server τ is T_b roughly. As a TLS server waits for more than one RSA decryption request and performs one big computation for all decryptions, it can save a lot of running time capacity, being able to perform more TLS handshakes.

Theorem 1. To satisfy the client’s requirement for the stability of the system, the batching service time is less than the batch size multiplying mean Poisson distributed arrival time interval when the time in the Batch Queue Model M/D/1, thus

$$\tau \approx T_b < b/\lambda \tag{5}$$

Proof: Let $X_i(i=1,2,...)$ be the arrival time interval of two consecutive requests, and Y be the time interval of b consecutive requests. If the system achieves the stability when the time $t\rightarrow\infty$ for M/D/1 queue model, $T_b<E(Y)$, where $E(Y)$ is the expected value of Y. Because the X_i is a random variable with independent identical distribution, the average arrival time interval of b consecutive requests is

$$E(Y)=E\left(\sum_{i=1}^{b}X_i\right)=bE(X_i)=b/\lambda \tag{6}$$

Then Theorem 1 is proved.

Lemma 1. In the Batch Queue Model M/D/1, to satisfy the client’s requirement for the stability of the system, thus

$$T_q<b/2\lambda \tag{7}$$

Proof: In the Batch Queue Model M/D/1, the value of T_q is derived following Eq.(3)

$$T_q=\left(1-\frac{(1-\eta_1)(1-\eta_2)}{(1+\eta_1\eta_2-\eta_2)}\right)T_r=\left(\frac{1-e^{-\lambda\tau}}{1-e^{-\lambda\tau}+e^{-1.5\lambda\tau}}\right)T_r=\left(\frac{e^{-\lambda\tau}-1}{e^{-\lambda\tau}-1+e^{-0.5\lambda\tau}}\right)T_r=\left(\frac{1}{1+\frac{1}{(e^{-\lambda\tau}-1)e^{0.5\lambda\tau}}}\right)T_r \tag{8}$$

where, $T_r=0.5\tau$. Due to Theorem 1, it can be easily described as

$$T_q=\left(\frac{0.5\tau}{1+\frac{1}{(e^{-\lambda\tau}-1)e^{0.5\lambda\tau}}}\right)<\left(\frac{1}{1+\frac{1}{(e^{b}-1)e^{0.5b}}}\right)\left(\frac{b}{2\lambda}\right) \tag{9}$$

It can be easily described as when $b\geq 2$, $0.944\approx 1+\frac{1}{(e^2-1)e^{0.5\times 2}}\leq 1+\frac{1}{(e^b-1)e^{0.5b}}<1$. Then the value bound of the upper limit of T_q is estimated as $[0.944b/2\lambda,\ b/2\lambda]$. Then Lemma 1 is proved.

4.3 Optimal batch scheduling algorithm

If the constants and practical value of the client's tolerable waiting time T_t are known, the upper limit of batch size b can be estimated according to different arrival rates of the client. It is usually expected that the ideal response time of web sites is 1 or 2 seconds[7] and 50% people apparently will wait only 8 seconds for a website to download before they get fed up and move on[7]. It is assumed that the value T_t is equal to 1 second, 2 seconds and 8 seconds as examples both in the analytical model and simulation. The total customer response time T is denoted as the sum of T_q, T_c and T_b, where T_c is the time for waiting other client in the same batching. It is easily derived that the max value of T_c is $(b-1)\lambda$. Otherwise, the upper limit of T_b has been estimated as b/λ in Theorem 1. We can also estimate the upper limit of T_q as $b/2\lambda$ derived from Lemma 1. It is assumed that T will not exceed T_t. The upper limit of b is derived as the following

$$T=T_q+T_c+T_b=\frac{b}{2\lambda}+\frac{(b-1)}{\lambda}+\frac{b}{\lambda}<T_t\Rightarrow b<0.4(\lambda T_t+1) \tag{10}$$

The pseudo code (see Fig.3.) shows the optimal batch scheduling algorithm which is employed in a batching web server. Step one sorts b to satisfy max solution of $T_b<b/\lambda$ in ascending order from two to the upper limit using Eq.(10). The computation of T_b is performed using Eq.(5). In step two, the batch server performs a set of related tasks to optimal schedule. The server firstly constructs b queues $(Q(e_1),Q(e_2),\ldots,Q(e_b))$ for every exponent e_i. A

heavily loaded web server using a round strategy when dispatching the exponent to clients would incur minimal latency before receiving b TLS handshake requests. When new requests arrive, the server adds the client to the corresponding queue and initializes a timer for the client. If every queue has client, it means that the condition of optimal batch is satisfied. Then, the server excuses *batch_decryption*(). The meaning of function *batch_decyption* () is to do batch decryption with the client in the head of every not empty queue in the $(Q(e_1),Q(e_2),\ldots,Q(e_b))$. If the condition of optimal batch can not be satisfied, the server waits for a period of time that is equal to T_t surplus the max value of timer of the client in the head of every queue. Then the server does the *batch_decryption*(). The batch server has the ability to fall back on *conventional_RSA_decryption*() when only one client is in queues. If all queues do not have clients, the algorithm terminates.

```
Step1: Find out the solution of $b$
Input: $T_t,\lambda$;
Output: $T_b$, optimal batch size
Begin
1.   Compute the max batch size. maxbatchsize=
     int(0.4($\lambda T_t$+1)) (refer to Eq.(10)).
2.   If (maxbatchsize<=1) then do
3.     conventional_RSA_decryption(); return;
4.   For ($b$=2;$b$<=maxbatchsize;$b$++)
5.   {Success=false;
6.   Compute $T_b$ (refer to Eq.(4));
7.   If ($T_b<b/\lambda$) then {Optimalbatchsize=b;
     Success=true;}}
8.   If (!success) then return;
9.   If (success) then goto step2();
End
```

```
Step2: Optimal batch scheduling
Input: Optimal batch size, $T_t$;
Output: optimal batch scheduling.
Begin:
1.   Construct $b$ queues ($Q(e_1),Q(e_2),\ldots,Q(e_b)$)
       for every exponent $e_i$. maxtimer=0;
2.     Assign the exponents $\{e_1,e_2,\ldots,e_b\}$ to different clients using
       round robin strategy in serverhello message (refer to Fig.1.)
3.   while ($Q(e_1)$!=Null or ($e_2$)!=Null ... or ($e_b$)!=Null){
4.   If Client arrived then {match client. exponent=$e_i$,
     enqueue ($Q(e_i)$,client); initialize Client.timer}
5.   If ($Q(e_1)$!=Null and $Q(e_2)$!=Null ... and $Q(e_b)$!=Null)
6.   then {Do batch_decryption(); reset server_waiting_time;
     update queues ($Q(e_1),Q(e_2),\ldots,Q(e_b)$);)
7.   Else {          for ($j$=1; $j$<=$b$; $j$++)
8.   {If (($Q(e_j)$!=Null) and ($Q(e_j)$.head.timer>=maxtimer))
9.        {maxtimer=($Q(e_j)$.head.timer); } }
10.  If (server waiting time>=$T_t$–maxtimer) then
11.  {do batch_decryption(); reset server_waiting_time;
     update queues ($Q(e_1),Q(e_2),\ldots,Q(e_b)$);)
12.  Else continuing waiting for request of client;}
End
```

Fig.3 Optimal batch scheduling algorithm

5 Validation of Analytical Models and Performance Evaluation Study

5.1 Experiment configuration

The analytical models are executed on a machine with a Dell Intel Pentium IV processor clocked at 3.20GHz and 1GMB RAM. Specifically, this paper performs the simulation of batching RSA with very small public exponents, namely e=3,5,7,11,13,17 etc. The simulation result of the conventional RSA decryption time T_{rsa} with larger public exponent, namely e=65537 is about 32 ms which is tested using reiterative results. It is assumed public modulus N is 1024 bits length.

5.2 Validation of analytical models

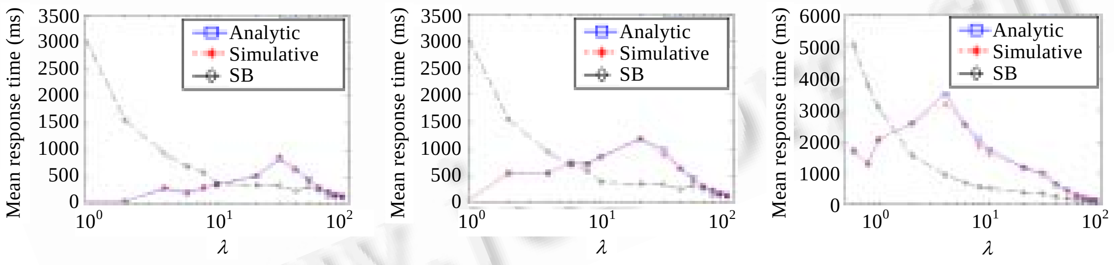
Table 1 validates optimal batch size described by the analytical model. As small arrival rates, b is almost uniform calculated by our analytical model (Table 1). Since the arrival rates are small (i.e., T_t=8, λ<0.6), there is very little opportunity to batch, and therefore, the solution of b is relative small (Table 1). Even at higher arrival rate, the analytical result and simulation result are very close. The solution of the optimal batch size is increased with λ both in analytic and simulation when λ<30 (i.e., T_t=1) approximately. Otherwise, T in this case is not increased obviously. The solution of b is decreased with the λ when λ>30 approximately.

Table 1 Optimal batch size results validation

λ	Optimal batch size					
	Analytical model			Simulation results		
	T_t=1	T_t=2	T_t=8	T_t=1	T_t=2	T_t=8
0.6	–	–	2	–	–	2
1	–	–	3	–	–	3
2	–	2	6	–	2	6
3	–	2	10	–	2	10
4	2	3	13	2	3	12
5	2	4	13	2	4	12
10	4	8	13	4	8	13
20	8	13	13	8	12	13
30	12	12	12	12	12	12
40	11	11	11	10	11	11
50	10	10	10	10	9	10
60	8	8	8	8	8	8
80	6	7	6	6	6	6
90	5	5	5	6	5	6
100	5	5	5	6	4	6

5.3 Performance evaluation

Fig.4(a)~Fig.4(c) illustrates the analytical mean response time T, our simulation results and that of SB (Shacham and Boneh) scheme. When λ is equal to 30 approximately, T reaches Maximum whereas the value is less than 1 second and decreased with the λ when using optimal batch size b (see Fig.4(a)). In the SB scheme, T of a batching system behaves poorly especially when λ does not exceed 10 requests/sec (see Fig.4(a)). When λ is larger than 10 and less than 100 approximately, the performance of the novel scheme and SB scheme are all satisfied with the clients' requirement of tolerable waiting time. However, it is shown that the solutions of b (Table 1) are larger than four. That means the optimal scheme can submit more decryption requests once to decryption device than SB scheme. The other two cases with T_t=2 (see Fig.4(b)) and T_t=8 (see Fig.4(c)) can be analyzed similarly as that with T_t=1 (see Fig.4(a)). The analytical and simulation results of T shows the novel batch scheme behaves nicely.



(a) T_t=1 (b) T_t=2 (c) T_t=8

Fig.4 Mean response time validation over client's tolerable waiting time

It is assumed that T_t is equal to 8 seconds in Fig.5(a)~Fig.5(c). These figures show that T is almost linear when λ is relatively small. This is due to the fact that T=T_q+T_c+T_b. When λ is relatively small, the main contribution to T is made by T_c. It is evident that the time T_c is increased linearly with b. T_b is also increased with b. Therefore T is also increased with b when λ is relatively large (i. e., λ=80).

A non-batching system becomes unstable when $\lambda>1/T_{rsa}=1/0.032=31.25$ due to the fact that a non-batching system becomes unstable when $\lambda>\tau$. But with batching, a batch system behaves nicely even at high loads. When the non-batching system is stable, the mean response time T' can be estimated as Eq.(11) (see Fig.6).The mean service time τ' is deterministic in the non-batching in M/D/1 queue model. Since T_{rsa} is the majority of service time, the mean service time τ' of the server is roughly T_{rsa}[5].

$$T' = \tau' + \tau'\left(\frac{\tau'\lambda}{2(1-\tau'\lambda)}\right) \approx T_{rsa} + T_{rsa}\left(\frac{T_{rsa}\lambda}{2(1-T_{rsa}\lambda)}\right) \tag{11}$$

Figure 7 (i.e. T_t=8) illustrates the comparison of the mean response time of the batching schemes with the non-batching scheme. The vertical axis in each graph is the mean response time over batch size divided by the mean response time with non-batching scheme. Refering to Table.1, it is shown that the optimal batch size is equal to 12 when λ=30. The speedup of mean response time is an optimal one which equals to 5.23 approximately. It is clear that with the optimal batch size the batching system has considerable advantages whereas costs little.

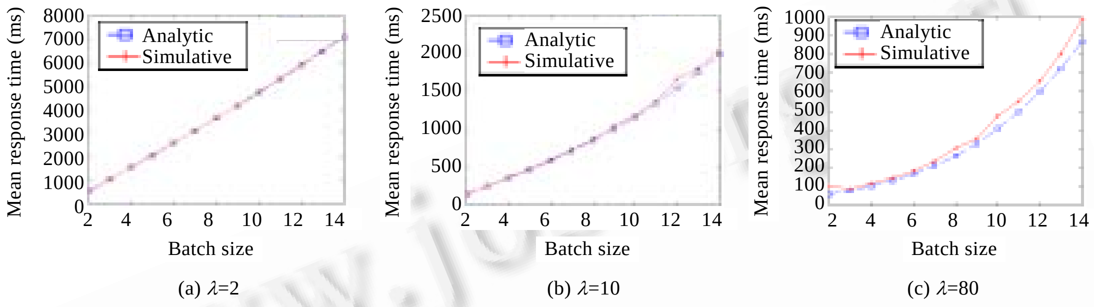


Fig.5 Mean response time validation over batch size

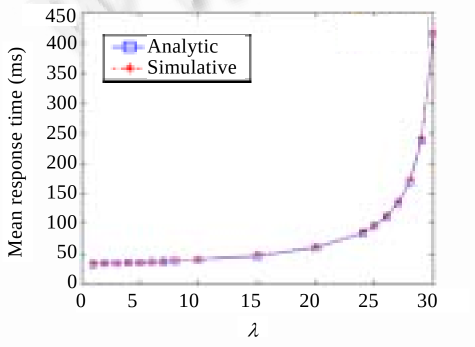


Fig.6 Mean response time for non-batching scheme

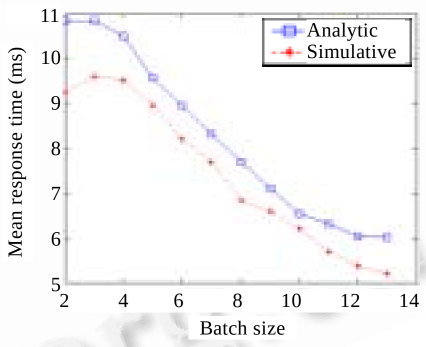


Fig.7 Mean response time speedup against non-batching

6 Conclusions

In conclusion, this paper proposes a novel method of assigning the set of public exponent e_i only using unique certificate in batching TLS handshake protocol. This paper also optimizes the batch size by developing an analytical model which satisfies the stability conditions of the system for the batching TLS handshake. The novel optimal batch scheduling algorithm is employed in a batching web server which satisfies the clients' requirement of stability of system and tolerable waiting time. The parameter optimization-based batching TLS handshake is a viable option for secure communications.

References:

[1] Sun LH, Ye DF, Lü SW, Feng DG. Security analysis and improvement of TLS. Journal of Software, 2003,14(3):518–523 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/518.htm

[2] Fiat A. Batch RSA. Journal of Cryptology, 1997,10(2):75–88.

[3] Shacham H, Boneh D. Improving SSL handshake performance via batching. In: Proc. of the RSA 2001. LNCS2020, San Francisco: Spring-Verlag, 2001. 28−43.

[4] Lin C. Performance Evaluation of Computer Network and Computer System. Beijing: Tsinghua University Press, 2001. 26−65 (in Chinese).

[5] Cheng WC, Chou CF, Golubchik L. Performance of batch-based digital signatures. In: Proc. of the 10th IEEE Int'l Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems. Fort Worth: IEEE Computer Society, 2002. 291−299.

[6] Vuillaume C. Side channel attacks on elliptic curve cryptosystems [MS Thesis]. Berlin: Technological University of Berlin, 2004.

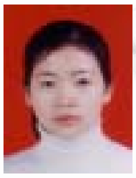
[7] Shan ZG, Lin C, Xiao RY, Yang Y. Web quality of service: Survey. Chinese Journal of Computers, 2004,27(2):145−156 (in Chinese with English abstract).

附中文参考文献:

[1] 孙林红,叶顶锋,吕述望,冯登国.传输层安全协议的安全性分析及改进.软件学报,2003,14(3):518−523. http://www.jos.org.cn/1000-9825/14/518.htm

[4] 林闯.计算机网络和计算机系统的性能评价.北京:清华大学出版社,2001.26−65.

[7] 单志光,林闯,肖人毅,杨扬.Web QoS 控制研究综述.计算机学报,2004,27(2):145−156.



QI Fang was born in 1978. She is a Ph.D. candidate at Central South University. Her current research areas are network security, mobile computing and wireless communications.

JIA Wei-Jia was born in 1957. He is a professor and doctoral supervisor at Central South University and a CCF Senior member. He is also associate professor at City University of Hong Kong. His research areas are designing routing protocols, wireless communications, mobile computing, parallel and distributed computing.

BAO Feng was born in 1962. He is a principal scientist in Singapore. He is also the manager of security department of the Institute for Infocomm Research. His current research areas are cryptography and information security.

WU Yong-Dong was born in 1970. He is a research scientist and the head of information security laboratory of the Institute for Infocomm Research. His current research areas are multimedia security, network security and digital right management.

WANG Guo-Jun was born in 1970. He is a professor and doctoral supervisor at Central South University and a CCF senior member. His research areas are computer networks, fault tolerant and dependable computing and software engineering.