门限 RSA 中的子密钥优化分配算法

崔竞松[1)] 彭 蓉[2)]

[1)](武汉大学计算机学院 武汉 430072)

[2)](武汉大学软件工程国家重点实验室 武汉 430072)

摘 要 无 Dealer 的(t,n)门限 RSA 算法能够使容侵系统在部分系统遭受攻击的情况下,继续保持系统私钥的安全性和可用性.在传统的无 Dealer 的环境中,为了使系统在遭受攻击时,以 $d=\sum d_i$ 方式共享私钥的 n 个参与方中的任意 t 个参与方有能力重构原始私钥 d,要求每个参与方必须持有 C_n^t 个子密钥.这种共享方式是诸多门限 RSA 的密钥共享方式中最简单、也是最有效的一种.然后 C_n^t 随着容侵系统的规模增长太快.所以减少每个参与者所应拥有的子密钥对于提高容侵系统的容侵能力尤为重要.该文提出了一种弹性搜索算法并搜索得到优化的子密钥分配方案.该优化方案能够在不降低系统安全性的前提下,以更少的子密钥实现了相同的门限结构,从而降低了门限密码系统的密钥管理以及签名和解密的相关操作的复杂度.

关键词 密钥分配;门限 RSA;容侵系统

中图法分类号 TP309

The Optimized Subkey Distribution Algorithm in Threshold RSA

CUI Jing-Song[1)] PENG Rong[2)]

[1)](*School of Computer Science, Wuhan University, Wuhan* 430072)

[2)](*State Key Laboratory of Software Engineering, Wuhan* 430072)

Abstract The (t,n) threshold RSA algorithm enables the system to keep the security and availability of the private key without a trusted dealer when parts of system are already under attack. In the traditional threshold RSA sub key distribution scheme, each party that shares the private key in the form of $d=\sum d_i$ must keep C_n^t sub keys secretly in order to enable the whole system to reconstruct the original private key by any t out of n parties. It is the simplest one of all the forms of sharing RSA key and it has the best efficiency out of them. However C_n^t increases with the scale of intrusion tolerance (IT) system very quickly. So it is very important to decrease the number of sub keys which belong to each party to enhance the IT ability of the ITS. This paper presents a scalable searching algorithm, which can find out the optimized sub key distribution scheme that can decrease the number of sub keys that should be held by each party without weakening the system's security. And an optimized scheme is found and listed in this paper. It can greatly reduce the complexity of key management and related operations in signing and decryption.

Keywords key distribution; threshold RSA; intrusion tolerance system

1 引 言

门限 RSA 算法分为两大类:有 Dealer 的算法和无 Dealer 的算法.有 Dealer 的门限 RSA 算法[1,2]要求系统中存在着一个参与各方都绝对信任的 Dealer.它必须诚实可靠.而无 Dealer 的门限 RSA 算法[3]没有这一点要求,但其算法实现和密钥管理

收稿日期:2003-12-24;修改稿收到日期:2004-08-20.崔竞松,男,1975 年生,博士,讲师,主要研究方向为信息安全、网络安全、算法优化.E-mail: cui_js@sina.com.彭 蓉,女,1975 年生,博士,讲师,主要研究方向为并行处理、分布式计算.

相对复杂.

在无 Dealer 的门限 RSA 算法中,n 个参与方可以通过系列协议[4],在指定 e 的条件下,协商生成 RSA 算法所需要的公钥 N,并且参与各方 P_i 还秘密持有一个 d_i 满足 $e\sum d_i \equiv 1 \bmod \phi(N)$,且任何少于 $\lfloor n/2 \rfloor$ 方都不可能获得 N 的分解式. 因此系统的私钥 $d=\sum d_i$ 的秘密性得到了有效的保护,但其容错能力较弱. 假如 n 个参与者中任意 1 个损坏,就会导致整个系统无法正常行使 d 的功能(如签名). 因此需要进行子密钥的门限分配,以对系统私钥提供容错性方面的保护.

2 门限分配方案

门限 RSA 中子密钥门限分配任务可以用如下方式描述:

任务 1. 在 (t,n) 门限 RSA 中,设参与各方为 $P_i, i=1,2,\cdots,n$,系统私钥为 d,要求任意 t 个不同的参与者 $\{P_{a_1}, P_{a_2}, \cdots, P_{a_t}\}$ 能够重构出 d,而任意 $t-1$ 个则不能,其中 $a_l \in [1,n], l=1,2,\cdots,t$.

传统的解决方法[5]是:先将 d 分成若干随机数之和,再将这些随机数按照 C_n^t 全排列的方式分配给参与各方.

例如:对于 $t=2, n=3$,先产生随机数 $d_1, d_2, \cdots, d_6$ 且满足

$$d = d_1 + d_2 = d_3 + d_4 = d_5 + d_6,$$

而后,按照如下方式分配子密钥:

P_1	P_2	P_3
d_1	d_2	
	d_3	d_4
d_5		d_6

即 P_1 秘密拥有 d_1 和 d_5,P_2 秘密拥有 d_2 和 d_3,P_3 秘密拥有 d_4 和 d_6.

显然,从 $\{P_1,P_2,P_3\}$ 中任取 2 方都可以重构出 d:

$\{P_1,P_2\}$: $d = d_1 + d_2$

$\{P_1,P_3\}$: $d = d_5 + d_6$

$\{P_2,P_3\}$: $d = d_3 + d_4$.

由于分配方案中行与行之间的子密钥是随机独立的,因此,任何一行的子密钥对于猜测其它行的子密钥没有提供任何帮助. 而且每行中至少需要 t 个量方能计算得到 d. 因此,这种解决方法是安全的.

显然对于同一个门限分配任务,可以有多种分配方案[6]. 例如:对于一个 $(3,4)$ 门限分配任务,至少可以有如下两种方案:

表 1 (3,4)门限分配方案 a

P_1	P_2	P_3	P_4
$d_{1,1}$	$d_{1,2}$	$d_{1,3}$	
$d_{2,1}$	$d_{2,2}$		$d_{2,3}$
$d_{3,1}$		$d_{3,2}$	$d_{3,3}$
	$d_{4,1}$	$d_{4,2}$	$d_{4,3}$

其中,对于任意 $i=1,2,\cdots,4$ 有 $d=\sum_j d_{i,j}$.

表 2 (3,4)门限分配方案 b

P_1	P_2	P_3	P_4
$d_{1,1}$	$d_{1,2}$	$d_{1,3}$	$d_{1,3}$
$d_{2,1}$	$d_{2,1}$	$d_{2,2}$	$d_{2,3}$

其中,对于任意 $i=1,2$ 有 $d=\sum_j d_{i,j}$.

显然以上两种方案都可以完成 $(3,4)$ 门限结构. 其中方案 b 仅使用 2 组子密钥即可达到方案 a 使用 4 组子密钥所达到的功能,且安全性相当.

显然,对于 (t,n) 门限分配任务,参与各方至多需要持有 C_n^t 个子密钥. 在实际应用中,由于 C_n^t 呈指数增长(一般在应用中要求 $t \geq n/2$),如 $C_3^2=3, C_5^3=10, C_9^5=126, C_{11}^6=462$,使得密钥管理难度较大,从而约束着门限系统的规模,最终间接制约着系统的容侵、容错能力.

作者提出了一种针对门限子密钥分配问题的解决方案[7],力图使用较少的子密钥,达到等效的门限结构.

3 优化分配算法

3.1 构造优化方案

为了研究方便,我们将上述门限分配方案中的子密钥的表达方式转换成如下形式:

表 3 转换后的(3,4)门限分配方案 b

P_1	P_2	P_3	P_4
1	2	3	3
1	1	2	3

表 3 中第 i 行的 1,2,3 分别对应相应的 $d_{i,1}, d_{i,2}, d_{i,3}$. 显然这种表示方法更简明,且与上述方案 b 是一致的.

定义 1. 对于给定的 t 和 n,定义组合 $\tau:\tau=\{a_1,a_2,\cdots,a_t\}$,其中 $a_i \in [1,n], i=1,2,\cdots,t$,且 $a_i=a_j$ iff $i=j$.

定义 2. 对于给定的 t 和 n,定义 τ 的全集 R:$R=\{\tau\}$.

则任务 1 等价于任务 2：

任务 2. 对于给定的 t,n，寻找 s 最小的矩阵 $\boldsymbol{K}_{s,n}$，使得该矩阵满足

$$\forall_{\tau\in R}(\exists_{x\in[1,s]}(\{k_{x,a}\mid a\in\tau\}=\{1,2,\cdots,t\})) \quad (*)$$

其中 $k_{x,a}$ 是矩阵 $\boldsymbol{K}_{s,n}$ 第 x 行第 a 列的元素.

目前国际上尚未出现对于上述问题的有效解决方法. 因此，本文提出了一种构造算法以获得满足(*)的较优化的矩阵.

3.1.1 优化矩阵的构造

矩阵 $\boldsymbol{K}$ 由 s 行组成. 可以证明：对于每一行有 G_n^t 种不等价的可能. 其中，

$$G_n^t=\begin{cases}1, & t=1 \text{ 或 } n\\ tG_{n-1}^t+G_{n-1}^{t-1}, & 1<t<n\end{cases}.$$

显然，由于 G_n^t 呈指数增长($G_3^2=3, G_5^3=25, G_9^5=6951, G_{11}^6=179487$)，因此，当 $n>10$ 时，采用穷举法搜索最优解已经不可能. 所以，本文通过寻找局部最优解的方式来构造全局满意解. 即希望构造 s 较小的矩阵 $\boldsymbol{K}$.

定义 3. 定义 $satisfy(x,\tau)$ 表示 $\{k_{x,a}\mid a\in\tau\}=\{1,2,\cdots,t\}$，即对于问题 2 中的(*)，$\tau$ 可以被矩阵 $\boldsymbol{K}$ 中的第 x 行满足；

定义 4. 定义 $\Gamma_x=\{\tau\mid satisfy(x,\tau)\}$，即矩阵第 x 行可以满足的所有组合的集合；

定义 5. 定义 $R_x=R-\bigcup_{i=1}^{x}\Gamma_i$，即矩阵中前 x 行都未满足的组合的集合.

则任务 2 可以转化为任务 3：

任务 3. 对于给定的 t,n，逐行构造 K，使得 K 除满足(*)的要求外，在构造第 x 行时，要求 R_x 最小.

由于每个子密钥在使用的过程中的使用方法和地位都是相等的，矩阵中同一行中的数字大小是无序的，所以我们只关心：哪些列的元素是相同的，哪些是不同的. 显然，这是一个等价类划分问题.

如果将 n 列的初始状态视为 n 个等价类，则每一行的划分方案的确定可以看作一个合并过程，即任务 4.

任务 4. 对于给定的 t,n，在 n 个等价类中，重复挑选其中的 2 个进行合并，直至剩下 t 个等价类. 并使得合并后 R_x 最小.

3.1.2 构造算法

本文设计了如下算法完成任务 4：

算法思路：当给定 R_{x-1} 时，如果将第 x 行的第 u 列与第 v 列所在的等价类合并，则 R_{x-1} 中所有包含 u 和 v 的组合将都不可能被第 x 行所满足. 这些组合将被加入到 R_x 中，并必须由后续的行满足. 因此，可以选择适当的等价类进行合并以使每次合并后 R_x 增长得最少，希望第 x 行合并完毕后 R_x 尽可能小，以使后续行以尽可能少的行数满足剩余的组合 R_x.

算法变量含义：

x 表示正在构造的矩阵行的行编号；

R_x 的含义同定义 5；

G_i 表示第 i 个等价类；

当合并剩下 gc 个等价类时，$U_{i,j}$ 表示由于将第 i 个和第 j 个等价类合并而导致不能够被第 x 行满足的组合的集合，S_{gc} 表示仍然能够被第 x 行满足的组合的集合，T_{gc} 表示不能够被第 x 行满足的组合的集合；

s 和 $k_{x,j}$ 的含义与任务 2 中的描述相同.

矩阵 $\boldsymbol{K}$ 的构造算法如下：

1. $x=1, R_0=R$；
2. while $R_{x-1}\neq\varnothing$
   $gc=n, S_{gc}=R_{x-1}, T_{gc}=\varnothing, G_i=\{i\}, i=1,2,\cdots,n$；
   while $gc>t$
   $w=MaxInt$；
   for $i:=1$ to $gc-1$
   for $j:=i+1$ to gc
   $U_{i,j}\leftarrow\{\tau\mid\tau\in S_{gc}\wedge\exists_{u,v}(u,v\in\tau\wedge u\in G_i\wedge v\in G_j)\}$；
   if $|U_{i,j}|<w$ then $w=|U_{i,j}|, i'=i, j'=j$；
   (**)
   $S_{gc-1}=S_{gc}-U_{i',j'}, T_{gc-1}=T_{gc}+U_{i',j'}$,
   $G_{i'}=G_{i'}+G_{j'}$；
   交换 $G_{j'}$ 和 G_{gc}，$gc=gc-1$；
   对所有 j 满足 $j\in G_i, i=1,2,\cdots,t$，置 $k_{x,j}=i$；
   $R_x=T_t, x=x+1$；
3. $s=x-1$.

上述构造算法可以逐行生成矩阵 $\boldsymbol{K}$(共 s 行). 对于每一行，上述算法将初始的 n 个等价类 $G_1, G_2, \cdots, G_n$ 合并成为 t 个等价类 $G_1, G_2, \cdots, G_t$. 每个等价类 G_i 内的元素在矩阵的该行中对应的列上的元素都等于 i.

3.1.3 算法正确性证明

引理 1. 在上述构造算法中，对于任意 $x\in[1,s]$ 有 $R_{x-1}\supset R_x$.

证明. 对于给定的 x：

(1) 对任意 $\tau\in R_x$，有 $\tau\in T_t$. 又∵ $T_t\supseteq T_{t+1}\supseteq\cdots$

$\supseteq T_n=\varnothing$，∴ $!\exists_{r\in[t+1,n]}(\tau\in T_{r-1}\wedge\tau\notin T_r)$，即当 $gc=r$ 时，$\exists_{i',j'}(\tau\in U_{i',j'})$，∵$U_{i',j'}\subseteq S_r$ ∴$\tau\in S_r$，又∵ $S_r\subseteq S_{r+1}\subseteq\cdots S_n=R_{x-1}$. ∴$\tau\in R_{x-1}$，∴$R_{x-1}\supseteq R_x$.

(2) 证明 $R_{x-1}\neq R_x$. 使用反证法：假设 $R_{x-1}=R_x$，即 $S_n=R_{x-1}=R_x=T_t$. 易证对任意 $r\in[t,n]$ 有 $S_r\cup T_r=R_{x-1}$ 且 $S_r\cap T_r=\varnothing$. ∴$S_t=\varnothing$. ∵ $S_n=R_{x-1}\neq\varnothing$，且 $S_r\subseteq S_{r+1}$，∴ $!\exists_{r\in[t+1,n]}(S_{r-1}=\varnothing\wedge S_r\neq\varnothing)$. 当 $gc=r$ 时有 $U_{i',j'}=S_{gc}$，设 $\tau=\{a_1,a_2,\cdots,a_t\}\in S_r$，显然 $a_1,a_2,\cdots,a_n$ 分别属于不同的 G_i. 由于 G_i 无序，所以不妨设 $a_i\in G_i(i=1,2,\cdots,t)$. 易证对任意 $i\neq j$ 有 $G_i\cap G_j=\varnothing$. 又∵ $gc=r>t$，∴ $\forall_{u\in\tau}(u\notin G_r),\tau\notin U_{k,r}(k=1,2,\cdots,t)$. 显然有 $U_{k,r}\subset S_{gc}$，∴ $|U_{k,r}|<|S_{gc}|=|U_{i',j'}|$. 而上述算法中 (**) 行的 w 记录着 $|U_{i,j}|$ 的最小值，这表明 i',j' 应满足：

$$|U_{i',j'}|=\min_{i,j\in[1,gc]\wedge i<j}\{|U_{i,j}|\}$$

根据上述推导，当 $i=k,j=r$ 时产生矛盾. ∴假设不成立. ∴$R_{x-1}\neq R_x$.

根据(1)，(2)可得 $R_{x-1}\supset R_x$. 证毕.

定理 1. 上述构造算法可在有限时间内终止.

证明. ∵$R_0=R$ 且由引理 1 可知 $R_0\supset R_1\supset\cdots$，∴一定存在 $s\leq|R|=C_n^t$ 使得 $R_s=\varnothing$. 即构造算法能在 s 次循环后终止. 证毕.

定理 2. 上述构造算法得到的矩阵 K 可以满足任务 2 中的(*)，即 $\forall_{\tau\in R}(\exists_{x\in[1,s]}(\{k_{x,a}|a\in\tau\}=\{1,2,\cdots,t\}))$

证明. 对任意 $\tau\in R$，根据引理 1 和定理 1 有 $!\exists_{r\in[1,s]}(\tau\in R_{r-1}\wedge\tau\notin R_r)$，即当$x=r$ 时，$\neg\exists_{u,v,i}(u,v\in\tau\wedge u,v\in G_i\wedge u\neq v)$，即在矩阵 K 中，组合 τ 的所有成员所对应的列在第 r 行的值皆不相等，即 $k_{r,i}\neq k_{r,j}(i\neq j)$. ∵$k_{r,j}\in[1,t],(j=1,2,\cdots,n)$，∴$\{k_{r,j}\}$ 中共有 t 个不相同的值：$1,2,\cdots,t$. 而 $|\tau|=t$，∴根据格笼原理，从$\{k_{r,j}\}$中选取 t 个不相等的值组成集合只可能是$\{1,2,\cdots,t\}$，∴$\{k_{r,a}|a\in\tau\}=\{1,2,\cdots,t\}$.

证毕.

综上所述，定理 1 保证了构造算法是可停机的；定理 2 保证了算法生成的矩阵是满足应用要求的.

显然该构造算法的时间复杂度为 $O(t^3)$. 所以该算法可以高效地得到优化的门限密码子密钥分配方案.

3.2 弹性搜索算法

在上述算法构造 K 的每一行时，该算法力求 T (即 R_x)以最慢的速度增长，使 $R_x,R_{x+1},\cdots$以较快的速度递减至$\varnothing$. 当 $R_x=\varnothing$时，矩阵 K 即构造完毕.

但由于在上述任务 3 和任务 4 中存在有局限性，所以上述算法所得到的分配方案并不一定是最优的. 通过算法改进，仍有可能得到更优的方案.

(1) 在解决任务 4 时，由于在每一步合并时 $|U_{i,j}|$ 最小并不一定保证任务 4 完成后 R_x 最小，因此，可将上述算法的(**)改进为将 $|U_{i,j}|$ 较小的一批$\langle i',j'\rangle$记录于一个队列 I 中，并逐对尝试. 由此即可产生多种对于第 x 行的分组方案候选.

(2)由于在解决任务 3 时，R_x 最小只能保证是这一行的最优解，但并不一定是全局的最优解. 因此，当尝试完了第 x 行的候选方案后，在一定的条件 C 下，可以回溯搜索第 $x-1$ 行，以尝试其它方案.

通过上述两种扩展方法可以得到一种弹性搜索算法. 其中，队列 I 的长度 $|I|$ 控制着搜索的宽度，回溯条件 C 控制着搜索的深度. 通过调整这两个参数可以在搜索代价和矩阵 K 的优化程度之间进行折衷.

3.3 搜索结果

通过对 $|I|$ 和 C 的选择，作者在 P4 1.3G 的 PC 机上，经过近 120h，搜索得到了一组门限 RSA 子密钥优化分配方案如表 4 所示. 表 5 列出了传统方案中需要子密钥的个数 C_n^t.

通过比较可以看出，优化分配方案所使用的子密钥的个数约为传统的 C_n^t 分配方案的 $1/n$. 而且随着 n 的增大，优化方案的优势将更加明显. 表中突出部分是在实际应用环境中 t,n 经常选择的范围.

由于通过上述弹性搜索算法搜索得到的分配矩阵能够满足问题 2 中的(*)，所以通过该矩阵构造出的(t,n)门限结构的正确性和安全性都与传统方案相同.

由于密钥的门限分配方案不含任何秘密信息，且在系统的运行过程中一般不会变更，所以子密钥的门限分配方案可以以离线方式进行长时间搜索，而后一次性的固化在系统之中. 搜索时间的长短不会影响实际系统的运行效率.

表 4 优化分配方案中的子密钥个数

n	t											
	2	3	4	5	6	7	8	9	10	11	12	13
3	**2**											
4	**2**	**2**										
5	3	**3**	3									
6	3	**3**	**5**	3								
7	3	4	**6**	6	4							
8	3	4	**6**	**8**	8	4						
9	4	4	8	**12**	**13**	9	5					
10	4	6	11	**13**	**20**	17	13	5				
11	4	6	13	20	**28**	**31**	25	14	6			
12	4	7	16	27	**29**	**46**	**45**	33	18	6		
13	4	7	17	32	47	**64**	**75**	66	44	20	7	
14	4	8	19	39	62	**66**	**108**	**116**	94	56	25	7

表 5　传统分配方案中的子密钥个数

n	t											
	2	3	4	5	6	7	8	9	10	11	12	13
3	**3**											
4	**6**	**4**										
5	10	**10**	5									
6	15	**20**	**15**	6								
7	21	35	**35**	21	7							
8	28	56	**70**	**56**	28	8						
9	36	84	126	**126**	**84**	36	9					
10	45	120	210	**252**	**210**	120	45	10				
11	55	165	330	462	**462**	**330**	165	55	11			
12	66	220	495	792	**924**	**792**	**495**	220	66	12		
13	78	286	715	1287	1716	**1716**	**1287**	715	286	78	13	
14	91	364	1001	2002	3003	**3432**	**3003**	**2002**	1001	364	91	14

4　结论与展望

本文提出了一种针对门限 RSA 中子密钥分配方案的弹性搜索算法，并通过该算法获得了一组优化的子密钥分配方案. 该方案可以使用约 C_n^t/n 组子密钥构造出与传统方案等效的子密钥分配方案. 优化方案一方面大幅降低了门限系统的密钥管理难度，另一方面也放宽了对门限系统规模的约束，有利于构造容错、容侵能力更强的门限系统.

可以证明，对(t,n)门限 RSA 子密钥分配问题，密钥分配矩阵 K 的行数的下限为

$$\left\lceil C_n^t \Big/ \left(\left(1+\frac{1}{j}\right)^r j^t\right)\right\rceil, j=\left\lfloor \frac{n}{t} \right\rfloor, r=n-j\times t.$$

本文工作所得到的分配方案距离这个下限值还有一定的距离. 在本文的基础上继续进行研究，仍有可能获取更优化的分配方案，或者构造并证明最优方案.

参　考　文　献

1　Rabin T.. A simplified approach to threshold and proactive RSA. In: Proceedings of CRYPTO'98, Springer LNCS 1462, 1998, 89～104

2　Jing Ji-Wu, Feng Deng-Guo. An intrusion tolerant CA scheme. Journal of Software, 2002, 13(8): 1417～1422(in Chinese)
(荆继武，冯登国. 一种入侵容忍的 CA 方案. 软件学报, 2002, 13(8): 1417～1422)

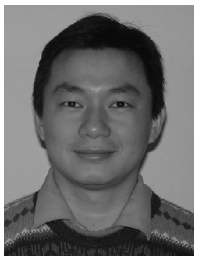
3　Damgard I. B., Koprowski M.. Practical threshold RSA signatures without a trusted dealer. In: Proceedings of Eurocrypt'01, LNCS 2045, Springer-Verlag, 2001, 152～165

4　Boneh D., Franklin M.. Efficient generation of shared RSA keys. In: Proceedings of Crypto'97, Springer-Verlag LNCS 1233, 1997

5　Malkin M., Wu T., Boneh D.. Experimenting with shared generation of RSA keys. In: Proceedings of the Internet Society's 1999 Symposium on Network and Distributed System Security(SNDSS), 1999

6　Wu T., Malkin M., Boneh D.. Building intrusion tolerant applications. In: Proceedings of the 8th USENIX Security Symposium, 79～91

7　Cui Jing-Song. The research of intrusion tolerance CA and threshold cryptography technology [Ph. D. dissertation]. Wuhan: Wuhan University, 2003(in Chinese)
(崔竞松. 容侵认证中心与门限密码技术研究[博士学位论文]. 武汉：武汉大学, 2003)



CUI Jing-Song, born in 1975, Ph.D.. His research interests include information security, network security and algorithm optimization.

PENG Rong, born in 1975, Ph.D.. Her research interests include parallel processing, distributed computing.

Background

The paper is a part of ITS research project of Information Security Lab. in Wuhan University. The global goal of this research project is to build secure and efficient ITS, including IT application and IT architecture. The Lab. has studied in this area for more than 3 years, proposed two IT research models and developed an IT CA prototype system. This paper aims to improve the IT ability of ITS, which exploits the threshold RSA algorithm, by increasing the number of share servers while the number of sub keys used by each server remains small or increases slower to enhance to overall performance of ITS.