Chapter 4

QoS of Private Cloud in Finite Population Environment

Cloud computing provides a new paradigm to meet the emerging business needs of industries for accessing distributed computing resources such as infrastructure, hardware and software applications on-demand over the internet as services [35, 53, 54]. As the technology and the needs are growing very fast, in future there may be multiple vendors offering different services with different Quality of Services (QoS) and at various prices [3, 55]. This would necessitate new tools and mechanisms for analyzing the performance of the system for matching the offerings with requirements. This chapter presents an analytical finite population model for performance evaluation of a private cloud computing system. Various performance measures of the cloud system for finite population environment indicate that the proposed provisioning technique help the cloud operators to tune the resources accordingly to improve the QoS targets.

4.1 Introduction

Cloud computing has been regarded as the new computing epitome that would redefine the way computing resources have been bought and utilized

[56]. The cloud computing, based on its business model, is now regarded as one of the important utilities [9, 57]. An organization has to invest capital expenditure for having its own dedicated computing resources. In the cloud computing environment, the organization has to pay for the operational cost only. That is, the organization pays for the consumed services only and it does not pay for the software or hardware resources. The cloud computing system consists of the following distributed components: servers, network devices, resources, storage systems, applications and services, etc. Users have the flexibility to use these distributed resources together to create a unique environment for themselves.

A private cloud configuration is attractive to organizations to leverage the scalability benefits of a shared virtual infrastructure that have a known application workload footprint in terms of CPU, memory, and storage utilization, and that have potentially stringent security considerations. It effectively eliminate the provisioning constriction present in the datacenter virtualization model while maintaining cost savings as the same many-to-one relationship exists between VMs and physical hosts [58]. In the private cloud environment, data and processes are managed within the organization without the restrictions of network bandwidth, which can be of great benefit for security compliance. In addition, private cloud services offer the provider and the user greater control of the cloud infrastructure, because user access and the networks used are limited and designated. To create private cloud initial cost is expensive, but gets minimal at later phases of using it as a service.

Cloud systems are receiving requests for different services and would in turn be provisioning virtual machines for servicing them [59, 60]. It would be easy to model the incoming requests and the provisioning of VMs using statistical models as all these functions are random processes. In this chapter, we present an analytical finite population model for performance evaluation of a private cloud computing system.

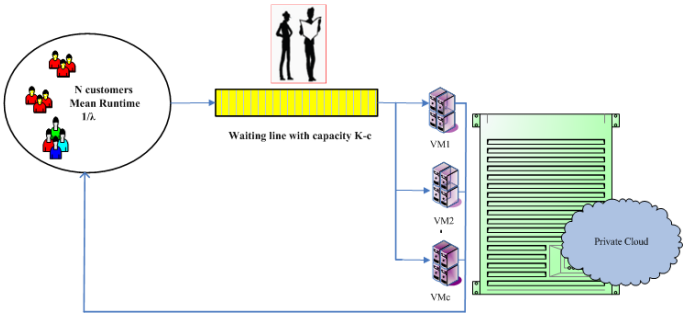

Figure 4.1: System model.

4.2 Service Policy 1

When a web application is deployed on a cloud, the cloud controller as the portal of a cloud establishes a queue to hold the client requests. A client requests for services through a cloud intermediary known as a cloud coordinator. When a client sends a request to the web application on the cloud, the cloud coordinator identifies the request and forwards the request to the appropriate web application. The client request will be put on the queue depending on the number of client requests arriving to the web application per unit time. If the number of client requests are small and the system is fast, the requests will be serviced without delay. On the other hand, if the number of client requests are large, the requests will have to wait in the queue for a while or a long time, depending on the speed of the system and the resource availability. The system model is depicted in Figure 4.1.

If the number of client requests is greater than the number of freely available VMs Policy 1 scheduling schedules the requests to the available VM and the other requests have to wait in the buffer till there turn to come [61]. The input source is limited, that is, the number of client request to the cloud system is of finite source N. For $c < K < N$ the system is considered as delay-loss system, that is, till there are $K-1$ client requests in the system the client request can arrive into the system but after that as the system is full the requests must return back to the source. The service times are inde-

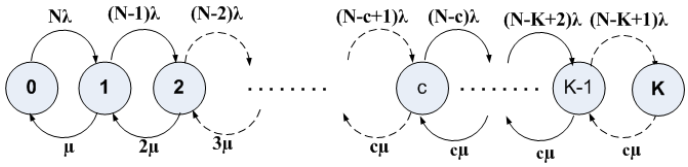

Figure 4.2: Rate transition diagram for policy-1.

pendent and exponentially distributed and the mean service time is $1/\mu$. The client requests are generated from N population according to quasi-random arrival process [62]. The state transition diagram is illustrated in Figure 4.2, where the state n denotes the number of client requests in the system. The one-dimensional process of the number of client requests at the service center is a birth-death process on a finite state space $\{0, 1, \ldots, K\}$ [63]. The quasi-random arrival process is a state-dependent birth process with the following rate

$$\lambda_i = (N - i)\lambda, 0 \leq i \leq K - 1,$$

and the death process with the rate

$$\mu_i = \begin{cases} i\mu & : \ 1 \leq i \leq c, \\ c\mu & : \ c < i \leq K. \end{cases}$$

We define P_i as the steady state probability that there are i client requests in the system. The steady-state distribution for finite buffer multi server queueing system with c homogeneous VMs are given by

$$P_i = \begin{cases} \binom{N}{i}\rho^i P_0, \ 0 \leq i < c, \\ \frac{i!}{c!c^{i-c}}\binom{N}{i}\rho^i P_0, \ c < i \leq K, \end{cases} \tag{4.1}$$

where $\rho = \lambda/\mu$. Using normalizing condition $\sum_{i=0}^{K} P_i = 1$, we have

$$P_0 = \left[\sum_{i=0}^{c-1}\binom{N}{i}\rho^i + \sum_{i=c}^{K}\binom{N}{i}\frac{i!\rho^i}{c^{i-c}c!}\right]^{-1}. \tag{4.2}$$

Computational aspects:

In most cloud computing system when the number of client request is large,

numerical difficulties arises in the direct use of the steady state probabilities. We show numerically stable methods of computation that avoids the computation of factorials and large power of loads. We establish steady state probabilities based on recursive relations [64, 65]. Taking $\phi_i = \frac{P_i}{P_0}$ and using the relation of the birth-death process, recursive procedure operates as follows.

$$\begin{aligned}
\phi_0 &= 1, \\
\phi_i &= \left(\frac{N-i+1}{i}\right)\rho\phi_{i-1}, \quad 0 \le i \le c-1, \\
\phi_i &= \left(\frac{N-i+1}{c}\right)\rho\phi_{i-1}, \quad c \le i \le K.
\end{aligned}$$

Since, we know $P_0 = 1 - \sum_{i=1}^{K} P_i$,dividing both sides by P_0, we have

$$1 = \frac{1}{P_0} - \sum_{i=1}^{K} \frac{P_i}{P_0} = \frac{1}{P_0} - \sum_{i=1}^{K} \phi_i.$$

Hence, $P_0 = \left[1 + \sum_{i=1}^{K} \phi_i\right]^{-1}$ and $P_i = \phi_i P_0$.
This computation is more stable than the direct use of (4.1).

4.2.1 Performance indices

Performance evaluation is an important aspect of cloud computing which is of crucial interest for both cloud providers and cloud customers. The effectiveness and utility of queueing model can be depicted and estimated by means of its performance indices. We obtain various performance measures in terms of steady state probabilities.

The expected number of client requests in the system (L_s) and Expected number of client requests in the queue (L_q) are respectively, given as

$$\begin{aligned}
L_s &= \sum_{i=0}^{K} iP_i \\
&= NP_0\left[\sum_{i=1}^{c-1}\binom{N-1}{i-1}\rho^i + \sum_{i=c}^{K-1}\binom{N-1}{i-1}\frac{i!\rho^i}{c^{i-c}c!}\right]. \\
L_q &= \sum_{i=c+1}^{K}(i-c)P_i = L_s - c + \sum_{i=0}^{c-1}(c-i)P_i
\end{aligned}$$

The mean number of busy servers ($\bar{c}$) and the mean number of requests in the source ($\bar{m}$) are respectively

$$\bar{c} = \sum_{i=1}^{c-1} iP_i + c\sum_{i=c}^{K} P_i, \quad \bar{m} = N - L_s.$$

The utilization of the system (U_c), utilization of the server (U_s) and the utilization of the sources (U_t) respectively, can be calculated as

$$U_c = 1 - P_0, \quad U_s = \frac{\bar{c}}{c}, \quad U_t = \frac{\bar{m}}{N}$$

The mean number of idle servers ($\bar{S}$) can be derived as

$$\bar{S} = c - \bar{c}$$

The effective arrival client request rate λ^e into the system is thus different from the overall arrival client request rate and is given by

$$\bar{\lambda} = \lambda \sum_{i=0}^{K-1} (N-i)P_i = \lambda \left[N - L_s(N-K)P_K\right]$$

The mean waiting time (W_q) and response time (W_s) using Little's formula can be derived as follows

$$W_q = \frac{L_q}{\bar{\lambda}}, \quad W_s = \frac{L_s}{\bar{\lambda}}.$$

Using Bayes' theorem we can see that for the probability of blocking, we have

$$P_B(N,c,K) = \frac{(N-K)P_K(N,c,K)}{\sum_{i=0}^{K}(N-i)P_i(N,c,K)} = P_K(N-1,c,K).$$

In particular if $K = N$, then $\bar{\lambda} = \lambda(N - L_s) = \mu\bar{c}$, $W_s = \frac{L_s}{\lambda(N-L_s)}$

4.2.2 Numerical Illustration

This section illustrates the numerical tractability that shed a light on the performance aspect of cloud computing which is of crucial interest for both cloud providers and cloud customers.The graphs for policy-1 are shown in Figure 4.3 to Figure 4.7. Figure 4.3 depicts the effect of λ on the expected

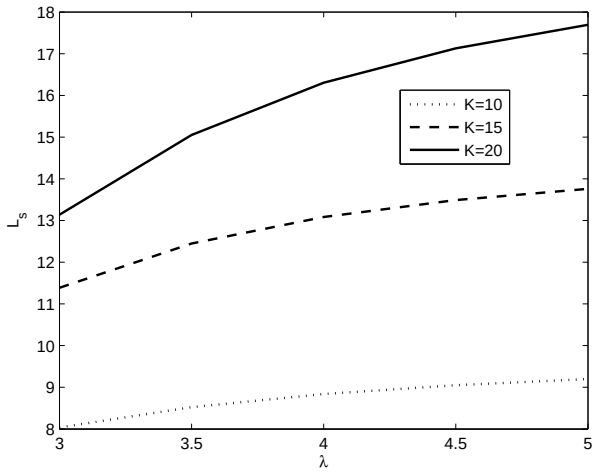

Figure 4.3: Impact of L_s on λ.

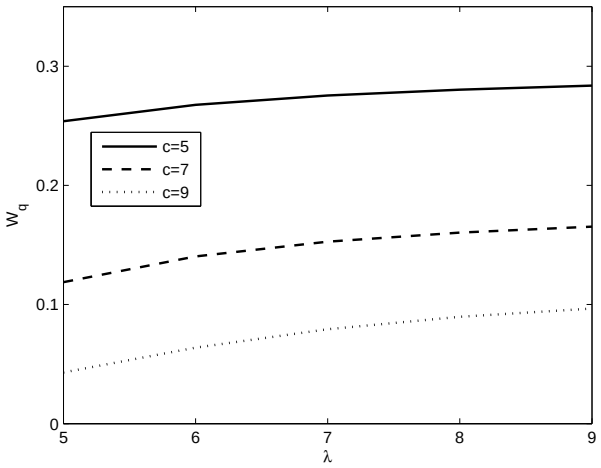

Figure 4.4: Impact of W_q on λ.

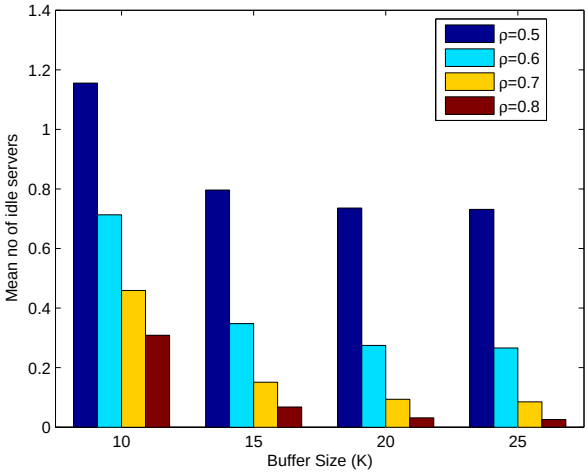

Figure 4.5: Impact of $\bar{S}$ on K.

number of client requests in the system (L_s) for various buffer sizes N. The parameters are taken as $N = 30$, $c = 5$ and $\mu = 10.0$. It is seen that as λ increases L_s increases monotonically. For fixed λ ,L_s decreases as the buffer size increases in the system. Figure 4.4 plots the impact of λ on the average waiting time in the buffer for different VMs. It can be observed that the waiting time in the buffer W_q increases with the increase in λ. As the number of VMs increases the average waiting time decreases which indicate that the server is available in the system more frequently, clearing the accumulated requests and as a result W_q decreases.

Figure 4.5 shows the impact of mean number of idle server's $\bar{S}$ on the buffer size in the cloud system for various ρ. The parameters are taken as $N = 30$, c=10. It can be observed that as ρ increases the mean number of idle servers' decreases. As expected for fixed ρ, the mean number of idle servers decreases as the buffer size in the cloud system increases. The effect of population size on the utilization of sources for various numbers of VMs c in the cloud system is presented in Figure 4.6. We see that for all the cases as population size increases, the utilization of sources also increases for a fixed number of VMs. One can also observe that as number of VMs c increases the

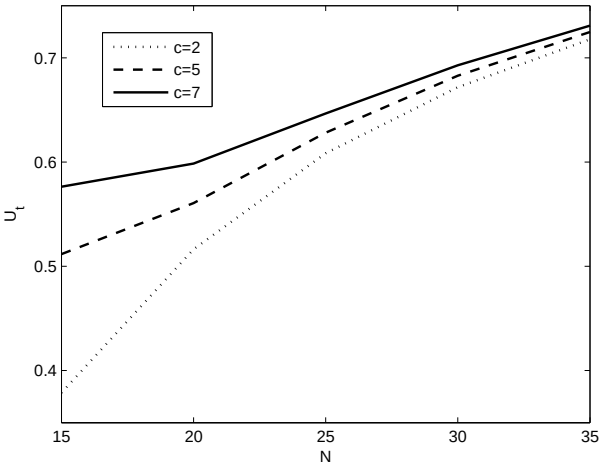

Figure 4.6: Impact of N on U_t.

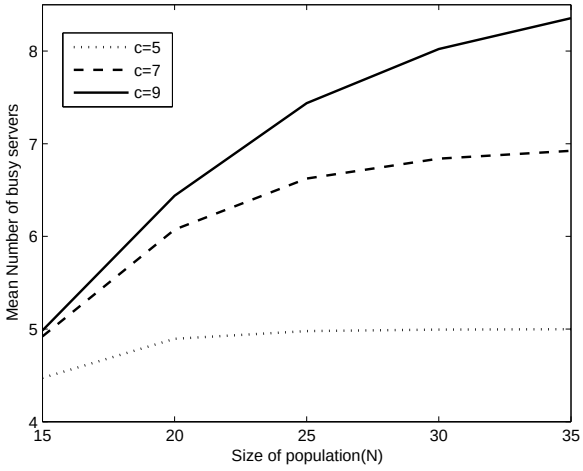

Figure 4.7: Effect of N on $\bar{c}$.

Table 4.1: Server Utilization by varying N and c.

λ	$K = N$	$c = 10$	$c = 15$	$c = 20$	$c = 25$	$c = 30$
3	30	0.6882	0.4615	0.3462	0.2769	0.2308
	35	0.7932	0.5384	0.4038	0.3231	0.2692
	40	0.8833	0.6150	0.4615	0.3692	0.3077
	45	0.9487	0.6907	0.5192	0.4154	0.3462
	50	0.9842	0.7642	0.5769	0.4615	0.3846
	55	0.9996	0.8332	0.6344	0.5077	0.4231
5	30	0.9269	0.6658	0.5000	0.4000	0.3333
	35	0.9831	0.7723	0.5833	0.4667	0.3889
	40	0.9980	0.8675	0.6663	0.5333	0.4444
	45	0.9999	0.9399	0.7481	0.6000	0.5000
	50	0.9999	0.9811	0.8259	0.6666	0.5556
	55	0.9999	0.9963	0.8947	0.7326	0.6111

utilization of sources also increases. Figure 4.7 illustrates the impact of the population size N on the mean number of busy servers $\bar{c}$ for various virtual machines c. It can be observed that when population size increases the mean number of busy servers in the system increases. For a fixed population size, the mean number of busy servers increases as the number of VMs c increases. Table 4.1 shows the server utilization for various values of $K{=}N$ and c by fixing μ as 10. The table shows for a fixed value of λ the server utilization decreases as the number of VMs in the system increases. One can also observe that for a fixed value of buffer size N and fixed number of VMs c the server utilization increases as the arrival request λ to the system increases. For a fixed value of λ and VMs c the server utilization increases as buffer size N increases. The system administrator can manage the parameters to improve the system.

4.3 Service Policy 2

Policy 2 scheduling is a partial cloud approach that provides an option for a partial allocation with blocking. This means, when the waiting buffer is

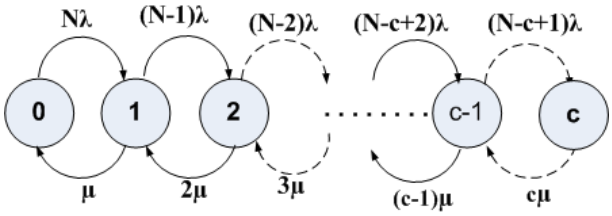

Figure 4.8: Rate transition diagram for policy-2.

full the client requests are denied (blocked). We adopt $M/M/c/c/N$ queuing system for the performance analysis of the above mentioned model. The client requests are generated from N population according to quasi-random arrival process. The state transition diagram is illustrated in Figure 4.8, where the state n denotes the number of client requests in the system. The one-dimensional process of the number of client requests at the service center is a birth-death process on a finite state space $\{0, 1, \ldots, c\}$. As before we can easily see that the number of client requests in the system is a birth-death process with rates

$$\begin{aligned} \lambda_i &= (N-i)\lambda, \quad 0 \leq i < c-1 \\ \mu_i &= i\mu, \qquad\qquad 1 \leq i \leq c. \end{aligned}$$

We define P_i as the steady state probability that there are i client requests in the system. The distribution can be obtained as

$$P_i = \binom{N}{i} \rho^i P_0, \ 1 \leq i \leq c, \tag{4.3}$$

where $\rho{=}\lambda/\mu$ is called the offered load per idle source. Using the normalization condition, we get

$$P_0 = \left[\sum_{i=0}^{c} \binom{N}{i} \rho^i \right]^{-1} \tag{4.4}$$

Hence

$$P_i = \frac{\binom{N}{i}\rho^i}{\sum_{j=0}^{c} \binom{N}{j}\rho^j}, i = 0, 1, \ldots, c. \tag{4.5}$$

Table 4.2: Performance indices for Engset loss system with c=6, μ=1, Nλ fixed.

		$N=6$	$N=60$	$N=300$	$N=600$	$N=900$
	P_c	0.000008499	0.000398040	0.000486721	0.000498745	0.000502801
N$\lambda=1$	P_B	0.000000000	0.000364204	0.000478576	0.000494580	0.000500004
	L_s	0.857143000	0.983254000	0.996202000	0.997843000	0.998391000
	P_c	0.004096000	0.099381100	0.113531000	0.115342000	0.115948000
N$\lambda=4$	P_B	0.000000000	0.094840300	0.112577000	0.114863000	0.115628000
	L_s	2.400000000	3.414590000	3.508180000	3.519780000	3.523640000
	P_c	0.015625000	0.000398040	0.000486721	0.000498745	0.000502801
N$\lambda=6$	P_B	0.000000000	0.000364204	0.000478576	0.000494580	0.000500004
	L_s	0.900000000	0.983254000	0.996202000	0.997843000	0.998391000

Remark If we take $N\lambda = \xi$ constant and let $N \to \infty$,then

$$\binom{N}{i}\rho^i \frac{N(N-1)\dots(N-i+1)}{N^i}\frac{(N\rho)^i}{i!} \to \frac{(\frac{\lambda}{\mu})^i}{i!}$$

Thus, the Engset distribution of (4.5) converges to the Erlang distribution. This is shown in Table 4.2.

4.3.1 Performance indices

The various performance measures for policy-2 are as follows:
There are on average $N-L_s$ sources that are eligible to generate new requests with intensity $\frac{\lambda}{\mu}$. Thus, the offered load (a) is expressed as

$$a = (N-L_s)\frac{\lambda}{\mu} = \rho\sum_{i=0}^{c}(N-i)P_i = \rho N\frac{\sum_{i=0}^{c}\binom{N-1}{i-1}\rho^i}{\sum_{j=0}^{c}\binom{N}{j}\rho^j} \tag{4.6}$$

The carried load a_c, the expected number of servers that are occupied at given time, can be obtained as

$$a_c = \rho\sum_{i=0}^{c}NP_i = \rho N\frac{\sum_{i=0}^{c}\binom{N-1}{i-1}\rho^i}{\sum_{j=0}^{c}\binom{N}{j}\rho^j} \tag{4.7}$$

From the equations (4.6) and (4.7) we can yield request congestion as

$$P_B(N,c) = 1 - \frac{a_c}{a} = \frac{(N-c)P_c(N,c)}{\sum_{i=0}^{c}(N-i)P_i(N,c)} = P_c(N-1,c).$$

The expected number of client requests in the system (L_s) is given as

$$\begin{aligned} L_s &= \sum_{i=0}^{c} iP_i \\ &= \frac{N\rho}{1+\rho}\sum_{i=1}^{c}\binom{N-1}{i-1}\left(\frac{\rho}{1+\rho}\right)^{i-1}\left(1-\frac{\rho}{1+\rho}\right)^{N-i} \end{aligned}$$

The utilization of the system (U_c) and the mean number of busy servers ($\bar{c}$)can be calculated as

$$U_c = 1 - P_0, \quad \bar{c} = L_s$$

The utilization of the server (U_s) and the utilization of the sources (U_t) are respectively can be calculated as

$$U_s = \frac{\sum_{i=1}^{c} iP_i}{c} = \frac{\bar{c}}{c}, \quad U_t = \frac{N - L_s}{N}$$

The blocking probability or call congestion that is the probability that a client request finds the system is full at his arrival, by the help of Bayes' rule can be computed as

$$P_B(N,c) = \frac{(N-c)P_c(N,c)}{\sum_{i=0}^{c}(N-i)P_i(N,c)} = P_c(N-1,c).$$

Let E(N,c,ρ) refer the blocking probability, that is E(N,c,ρ) =P_c(N-1,c) which is known as Engset's loss formula. This can be solved numerically by the following recursion:

$$\begin{aligned} E(N,c,\rho) &= \frac{\binom{N-1}{c}\rho^c}{\sum_{i=0}^{c}\binom{N-1}{i}\rho^i} = \frac{\binom{N-1}{c-1}\frac{N-c}{c}\rho^c}{\sum_{i=0}^{c-1}\binom{N-1}{i}\rho^i + \binom{N-1}{c-1}\frac{N-c}{c}\rho^c} \\ &= \frac{\frac{N-c}{c}\rho E(N,c-1,\rho)}{1+\frac{N-c}{c}\rho E(N,c-1,\rho)} \\ &= \frac{(N-c)\rho E(N,c-1,\rho)}{c+(N-c)\rho E(N,c-1,\rho)}. \end{aligned}$$

We can initialize

$$E(N,1,\rho) = P_1(N-1,1) = \frac{(N-1)\rho}{1+(N-1)\rho}.$$

The effective arrival client request rate $\bar{\lambda}$ into the system is thus different from the overall arrival client request rate and is given by

$$\bar{\lambda} \;=\; \lambda \sum_{i=0}^{c-1} (N-i)P_i.$$

The mean response time (W_s) using Little's formula can be derived as follows

$$W_s = \frac{L_s}{\bar{\lambda}}.$$

In particular , when $c = N$,one can easily see that $L_s = \frac{N\rho}{1+\rho}$ and thus $U_s = \frac{\rho}{1+\rho}$, $\bar{m} = \frac{N}{1+\rho}$, $U_t = \frac{1}{1+\rho}$ and $P_B = 0$ which was expected.

4.3.2 Numerical Illustration

The graphs are presented in Figure 4.9 to 4.11 for policy-2. Figure 4.9 depicts the effect of λ on the expected number of client requests in the system L_s for different c. The parameters are taken as $N = 30$ and $\mu = 10.0$. It is seen that as λ increases L_s increases monotonically. But unlike policy-1 for fixed λ, L_s decreases as c increases in the system.

Figure 4.10 illustrates dependence of server utilization on ρ varying from 0.5 to 0.9 and c varying from 6 to 14. We observed that for fixed c as ρ increases server utilization U_s increases. Further with fixed ρ the server utilization U_s decreases as c increases. Figure 4.11 depicts the effect of probability of blocking PBL on ρ for different c. It is seen that as ρ increases PBL increases monotonically. For fixed ρ, blocking probability decreases as c increases in the system.

Figure 4.12 shows the impact of ρ on L_s for both the policies. It can be seen that L_s increases as ρ increases. But, L_s is less in policy 2 as compared to policy 1. Utilization of the sources U_t , Mean number of busy servers $\bar{c}$

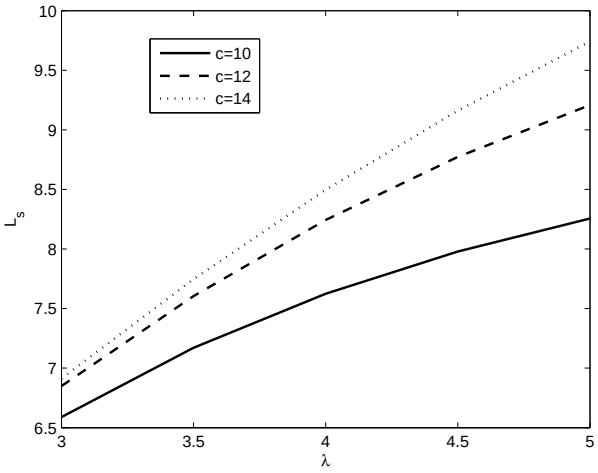

Figure 4.9: Impact of λ on L_s.

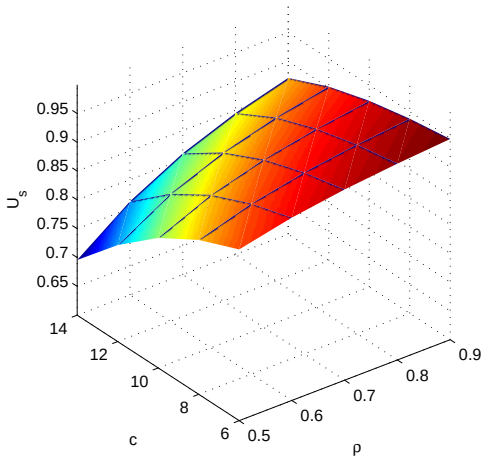

Figure 4.10: The Server utilization U_s for different values of ρ and c .

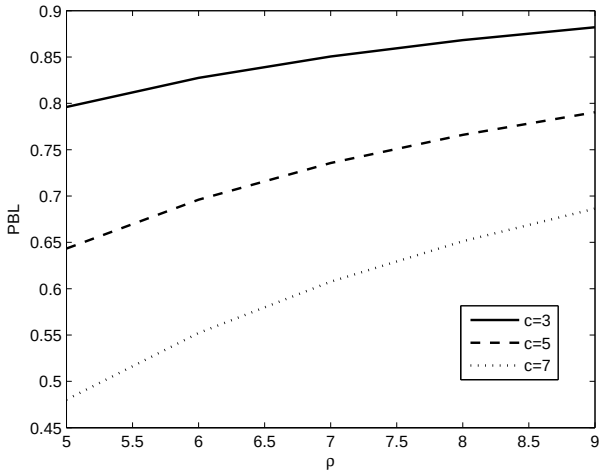

Figure 4.11: Impact of ρ on PBL.

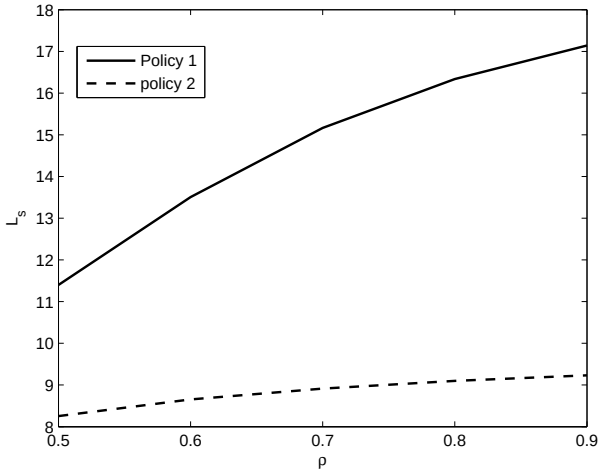

Figure 4.12: Impact of ρ on L_s for different policies.

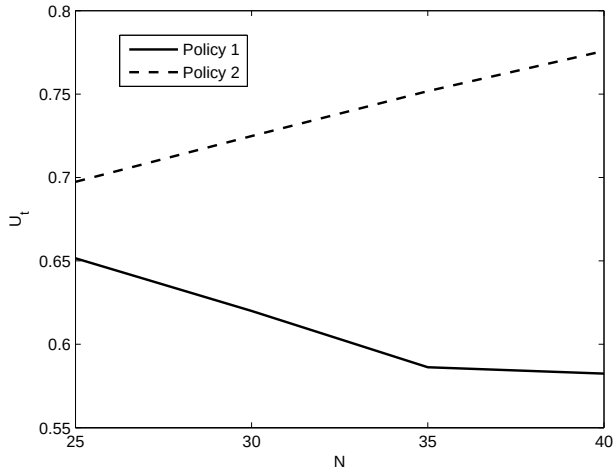

Figure 4.13: Impact of N on U_t for different policies.

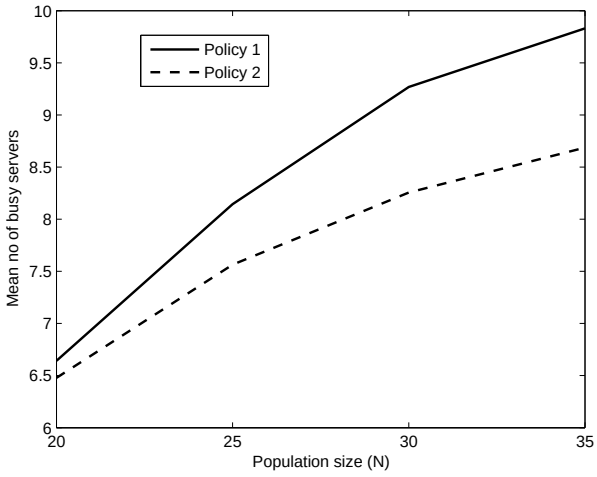

Figure 4.14: Impact of N on $\bar{c}$ for different policies.

Table 4.3: Comparison of utilization of the system(U_c), utilization of server(U_s) and mean number of busy servers($\bar{c}$) between the two policies.

			Policy 1			Policy 2		
c	$N=K$	ρ	U_c	U_s	$\bar{c}$	U_c	U_s	$\bar{c}$
5	10	0.5	0.983148	0.656634	3.28317	0.981221	0.617371	3.08685
		0.6	0.991373	0.729694	3.64847	0.989576	0.673400	3.36700
		0.7	0.995443	0.788646	3.94323	0.993902	0.717180	3.58590
		0.8	0.997525	0.835496	4.17748	0.996266	0.751862	3.75931
		0.9	0.998621	0.872260	4.36130	0.997622	0.779750	3.89875
	15	0.5	0.998372	0.898151	4.49076	0.996307	0.768956	3.84478
		0.6	0.998203	0.898151	4.49076	0.998203	0.810291	4.05145
		0.7	0.999847	0.974189	4.87094	0.999049	0.839992	4.19996
		0.8	0.999950	0.987277	4.93639	0.999461	0.862131	4.31065
		0.9	0.999983	0.993668	4.96834	0.999677	0.879156	4.39578
	20	0.5	0.999927	0.988746	4.94373	0.998988	0.843104	4.21552
		0.6	0.999990	0.997265	4.98633	0.999537	0.872420	4.36210
		0.7	0.999999	0.999318	4.99659	0.999766	0.892899	4.46449
		0.8	1.000000	0.999819	4.99910	0.999872	0.907908	4.53954
		0.9	1.000000	0.999949	4.99974	0.999925	0.919332	4.59666
10	20	0.5	0.999702	0.664137	6.64137	0.999688	0.647873	6.47873
		0.6	0.999919	0.742724	7.42724	0.999891	0.712140	7.12140
		0.7	0.999977	0.807376	8.07376	0.999971	0.761188	7.61188
		0.8	0.999993	0.858963	8.58963	0.999899	0.798663	7.98663
		0.9	0.999998	0.898782	8.98782	0.999996	0.827576	8.27576
	25	0.5	0.999963	0.814393	8.14393	0.999952	0.756439	7.56439
		0.6	0.999993	0.890853	8.90853	0.999988	0.809165	8.09165
		0.7	0.999999	0.940574	9.40574	0.999997	0.845723	8.45723
		0.8	1.000000	0.969576	9.69576	0.999999	0.871826	8.71826
		0.9	1.000000	0.98505	9.85050	1.000000	0.89106	8.91060
	30	0.5	0.999996	0.926862	9.26862	0.999991	0.825552	8.25552
		0.6	1.000000	0.973409	9.73409	0.999998	0.865242	8.65242
		0.7	1.000000	0.991537	9.91537	1.000000	0.891524	8.91524
		0.8	1.000000	0.997479	9.97479	1.000000	0.909832	9.09832
		0.9	1.000000	0.999261	9.99261	1.000000	0.923155	9.23155

versus N for both the policies are depicted in Figure 4.13 and Figure 4.14 respectively. Table 4.3 compares various parameters of both the policies. It can be observed that policy 1 utilizes the resources optimally.

4.4 Conclusion

In this chapter, two service policies along with an analytical resource prediction model for private cloud system have been proposed. Finite source finite buffer multiple server queueing model is applied to provide performance metrics of a private cloud computing system. We have developed a recursive method using the birth-death process, to obtain the steady-state system length distributions. Various performance measures such as utilization of the system, utilization of server, the expected number of client requests in the system, the expected number of client requests in the queue, mean number of idle servers, mean waiting time, response time and blocking probability are also carried out. The numerical computations under a range of parameters show that the proposed model improve the system performance and can optimize the organizational resources in cloud computing system.