Design and Analysis of a Replicated Server Architecture for Supporting IP-Host Mobility*

Jason P. Jue† and Dipak Ghosal‡
†Department of Electrical and Computer Engineering
University of California
Davis, CA 95616
‡Department of Computer Science
University of California
Davis, CA 95616
E-mail: jue@ece.ucdavis.edu, ghosal@cs.ucdavis.edu

Abstract

Mobility support in IP networks requires servers to forward packets to mobile hosts and to maintain information pertaining to a mobile host's location in the network. In the mobile Internet Protocol (mobile-IP), location and packet forwarding functions are provided by servers referred to as home agents. These home agents may become the bottleneck when there are a large number of mobile hosts in the network. In this paper, we consider the design and analysis of a replicated server architecture in which multiple home agents are used to provide mobility support. In order to minimize the delay across the home agents, one of the key aspects is the design of load balancing schemes in which a home agent may transfer the control of a mobile host to another home agent in the same network. The methods for triggering the transfer and the policy for selecting the next home agent define various load balancing schemes which have different performance characteristics. In this paper, we design a protocol that forms the building block for implementing such load balancing schemes, and we then study the performance characteristics of three selection schemes, namely, random, round-robin, and join the shortest queue (JSQ), and three transfers policies namely, timer-, counter- and threshold-based. The key results of this study are as follows: 1) The results show that both random and round-robin selection policies can yield modest load balancing gains, and that these gains increase when the traffic is more bursty (burstiness is defined as the ratio of the peak arrival rate to the mean arrival rate) as well as when there are more home agents. 2) The threshold-based transfer policy performs better than timer-based and counter-based policies, since in threshold-based policies transfers are made only when the queue is overloaded, unlike counter- and timer-based policies in which transfers can be made from an unloaded home agent to an overloaded home agent.

1 Introduction

A key objective in providing mobility support is to ensure that mobile hosts are able to send and receive messages when they move within a network or from one network to another. In IP networks, routing is based on the IP address of the host [12], [15]. When a host moves from one IP subnet to another, all IP packets addressed to that host will be routed to the host's old network. In order for the mobile host to receive messages at its new location, it must obtain an IP address in the

*This work has been supported in part by NSF Grant No. NCR-9703275.

new network, and messages sent to the original IP address need to be forwarded to the new IP address. In the mobile-IP protocol [1], this support is provided by two servers: the foreign agent and the home agent. A foreign agents supports the mobile host in the foreign network, while the home agent maintains the mobility binding and forwards packets to the mobile host when the mobile host has roamed out of its home network. While these agents can be implemented in the router, the protocol processing required by these agents are much more than the packet forwarding functions performed by a typical router. For example, the home agent has to perform proxy ARP and gratuitous ARP functions and must also encapsulate incoming IP packets into new IP packets before forwarding it to the appropriate router. Also, not limiting home agent functionality to the routers allows for a more scalable architecture in which other hosts in the network may be utilized to provide additional processing capacity as needed.

When there are large number of mobile hosts from the same home network, and if all of these mobile hosts are visiting foreign networks, then the servers that perform the agent functions can become bottlenecks. Furthermore, a single home agent or a single foreign agent is not a robust architecture, as it constitutes a single point of failure. In such a situation, it may be practical to employ multiple foreign agents and multiple home agents in an IP subnet. We view that these agents can be replicated in multiple routers and hosts. In order to effectively utilize the processing capacity of these mobility agents, it is necessary to evenly distribute the workload among them.

There has been extensive work in the area of load balancing [5][6][7][8]. Most of these works have considered balancing the load by transferring packets or jobs from one server to another. However, in the system that is being considered, this approach may be inefficient, since the cost of transferring a packet from one home agent to another may be on the same order as the cost of forwarding a packet to the foreign network. A more efficient approach would be to balance the load by initially directing packets to specific home agents rather that transferring packets among home agents after the packets have arrived to a home agent. In this work, we consider load balancing from a destination-oriented perspective, i.e., packets destined for a given mobile host are directed to a single home agent, and load balancing is accomplished by appropriately assigning mobile hosts to home agents. One approach is to statically assign mobile hosts to home agents and to foreign agents. In this approach, a single home agent will serve a mobile host for as long as that host is away from its home network, and a single foreign agent will serve a mobile host for the entire period during which the host remains in the foreign network. While this scheme may be able to evenly divide the number of mobile hosts among the mobility agents, it does not necessarily balance the instantaneous loads, since the load at a particular mobility agent will depend on the number of active connections to the mobile hosts which it is serving. For example, two home agents each serving the same number of mobile hosts may have significantly different loads if the mobile hosts being served by one home agent are receiving a high volume of traffic, while the mobile hosts being served by the other home agent are not receiving any traffic. Another disadvantage of a static assignment is that, since traffic tends to be bursty, the queue at a given home agent may tend to build up quickly during a burst, leading to high delays. If the burst is instead spread over a number of home agents, then the overall delay may be reduced.

In this paper, we attempt to solve the load balancing problem by presenting a scheme in which mobility agents take turns serving a mobile host. As a result, a mobile host which is receiving large amounts of traffic may be served by a number of mobility agents rather than being statically assigned to a single mobility agent. We describe the load balancing scheme only for home agents, but the results may also apply to foreign agents. The proposed scheme uses protocol functions already specified in the mobile-IP protocol; thus, the scheme requires only minimal changes to a home agent's software.

There are two key system design issues. The first issue is the discipline for selecting the next

home agent. We study three policies: random, round robin policy (RR), and join the shortest queue (JSQ). The random and the round robin selection policies are simple from an implementation standpoint, whereas the JSQ policy is more complex, and is used to provide a benchmark for the performance of the other two schemes. The other key issue is the mechanism that is used to trigger the transfer of a mobile host from one home agent to another home agent. We consider three schemes - timer-based, counter-based, and queue threshold-based. Each of these schemes is characterized by a different stream transfer parameter, namely, the stream transfer time, the stream transfer counter, and the stream transfer threshold respectively. These parameters define the time duration, the packet count, or the threshold value that results in the transfer of a mobile host from one home agent to another. The choice of these parameters determines both the load balancing gains as well as the associated overhead. For example, in the case of the timer based scheme, a small value of the stream transfer time can potentially result in high load balancing gains, since the input traffic can be evenly and uniformly distributed among the home agents on a packet-by-packet basis. However, decreasing the stream transfer time also increases the overhead, since more transfers are taking place. In this paper, we study this trade-off and compare the performance of the of the various selection and transfer policies, primarily through a detailed simulation model.

We compare the above load balancing scheme to two relatively static schemes: a burst-level load balancing scheme in which mobile hosts are transferred after each burst, and an equal-partition scheme in which the mobile hosts are uniformly assigned to home agents and no transfers are allowed. Our results show that both random and round-robin selection policies can yield modest load balancing gains, and that these gains increase when the traffic is more bursty (burstiness is defined as the ratio of the peak arrival rate to the mean arrival rate) as well as when there are more home agents. Furthermore, the threshold-based transfer policy performs better than timer-based and counter-based policies, since in threshold-based policies, transfers are made only when the queue is overloaded, unlike counter- and timer-based policies in which transfers can be made from an unloaded home agent to an overloaded home agent.

This paper is organized as follows. In Section 2, we discuss two extension of the Address Resolution Protocol (ARP), namely, the gratuitous ARP and the proxy ARP that are used in the mobile-IP protocol. We then discuss mobile-IP, focusing primarily on the aspect of the protocol that deals with the mobile host registration and routing. In Section 3, we discuss the mobile-IP protocol with multiple home agents and identify the two key aspects of the load balancing schemes, namely, 1) the transfer policy and 2) the selection policy. In Section 4, we describe the queueing model for the timer based transfer policy and the random selection policy, and outline the analysis for determining the mean response time of a packet at a home agent. The analytical and simulation results are discussed in Section 5. The comparison of the various load balancing policies through simulation results are discussed in Section 6. Finally, in Section 7, we conclude this paper and discuss some key future research directions.

2 Preliminaries

2.1 Address Resolution Protocol and Its Extensions

The address resolution protocol (ARP) is used in the TCP/IP protocol suite [12] to resolve network layer IP addresses to data link layer MAC addresses. In an Ethernet, a host which requires an address translation broadcasts an ARP query which specifies an IP address. The destination host responds with an ARP reply containing the requested MAC address. Each host also maintains an ARP cache which contains the translations of IP addresses to MAC addresses. There are two extensions to ARP that are used in basic mobile-IP: 1) the proxy ARP and 2) the gratuitous ARP.

Proxy ARP [12] is typically used when two physical networks with the same network ID are interconnected by a router. In such network, the router answers ARP requests on one of its networks for a host on another of its networks. This results in the destination IP address being mapped to the data link layer address of the router. All subsequent packets for the destination host are then sent to the router which then forwards the packets to the destination host in the other network.

In gratuitous ARP [12], the source host sends out an ARP request with a translation of its own IP address. This form of ARP 1) allows a host to determine if another host is already configured with the same IP address, and 2) allows other hosts in the network to update their ARP cache entry if the source host has changed its data link layer address. In an Ethernet, since the gratuitous ARP is sent out as a broadcast, each host in the network will update its ARP cache table entry and hence, this function cannot be used to selectively update the ARP cache in only a subset of hosts in the network.

2.2 Mobile-IP Protocol

The basic mobile-IP protocol has evolved out of the efforts of the mobile IP working group and specifies mobility support under IPv4. The basic mobile-IP protocol defines two agents for supporting mobility: the home agent and the foreign agent. The home agent is typically a router or host in the mobile host's home network which maintains a mobility binding (a permanent IP address to a temporary IP address translation) for the mobile host. A foreign agent may be a router or host in the network that the mobile host is visiting, and it provides the mobile host with a temporary IP address. If the number of available IP addresses in the foreign network is limited, then the foreign agent may act as a proxy for the mobile host. In this case, the temporary IP address will be the IP address of the foreign agent, and the foreign agent will forward packets to the mobile host. In this paper, we will assume that the foreign agent is acting as a proxy for the mobile host.

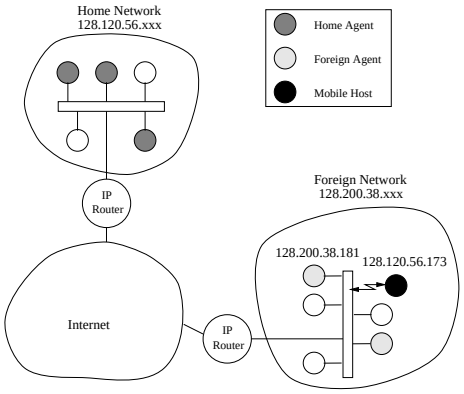

Figure 1: Mobile-IP with multiple home agents and foreign agents.

When a mobile host roams into a new network, it registers by sending a registration request to the foreign agent. In the request, the mobile host provides its permanent IP address and the IP address of the home agent. The foreign agent in turn sends a registration request to the home agent. This message, shown in Fig. 2, contains the permanent IP address of the mobile host, the IP address of the home agent, the IP address of the foreign agent (i.e., the temporary IP address), a lifetime, and an identifier which uniquely identifies the registration request. When the home agent receives the registration request, it updates the mobility binding of the mobile host and sends an acknowledgment back to the foreign agent. When the foreign agent receives the acknowledgment, it updates its own table and relays the reply to the mobile host. In the home network, the home agent

uses a gratuitous ARP [12] to update the ARP cache of all the hosts and routers that currently have an ARP cache entry for the mobile host. When the mobile host moves back into the home network, it de-registers with the home agent and sends a gratuitous ARP to update the ARP cache entries of the hosts and routers in the home network.

Type	Code	Lifetime
Permanent IP address		
Home Agent's IP address		
Care-of address/Temporary IP address		
Identification		
Identification		

Figure 2: Mobile-IP registration packet.

IP packets destined for a mobile host which is outside of its home network are routed through the mobile host's home agent. The home agent acts as a proxy for the mobile host, and responds to ARP queries for the mobile host with a proxy ARP reply [12]. The result is that the mobile host's IP address is now bound to the link layer address of the home agent, and all packets for the mobile host are now directed to the home agent. When the home agent receives an IP packet destined to the mobile host, it uses the mobility binding to encapsulate (or tunnel) the IP packet within another IP packet. The new IP packet has a destination address which is the temporary IP address of the mobile host and a source address which is the IP address of the source host. When the foreign agent receives the packet, it de-capsulates the packet and sends it to the mobile host. The acknowledgment is sent back directly to the source node by the mobile host.

The forward path routing may be inefficient since messages must first be routed to the home agent before being sent to the mobile host. If the source host and the mobile host are in the same network, but not in the home network of the mobile host, then the messages will experience unnecessary delay since they have to be first routed to the home network of the mobile host. One way to improve the mobile-IP protocol is route optimization [2] in which a message may be routed directly to the temporary IP address of the mobile host. Route optimization is accomplished by having the source host maintain a binding cache which contains the temporary IP address of the mobile host to which it is sending a message. One disadvantage of route optimization is that it requires the source host to be aware of the mobile host's mobility, whereas in the original mobile-IP protocol, the mobility of a destination host is transparent to the source host. This loss of transparency requires the source host to run additional software and maintain additional caches in order to provide route optimization. As a result, route optimization is optional in IPv4.

Recently, another transparent scheme has been proposed which requires additional agent functionality and is referred to as the External Agent [3]. The benefit of our load balancing scheme is that it is able to handle traffic from sources which do not provide route optimization, as well as initial bursts of traffic from sources which do provide route optimization.

3 Mobile-IP with Multiple Home Agents

In the mobile-IP environment, the load on a home agent consists of a number of TCP connections (e.g. ftp, telnet, etc.) from various source hosts to various mobile hosts. There are a number of possible ways to balance the load among the home agents. The traditional approach of transferring packets from a loaded home agent to another home agent has been studied in great detail in previous works [6]. The drawback of this approach is that the overhead of transferring a large

number of individual packets among the various servers may be high, and may also result in some packets being transferred more than once. Another approach is to balance the number of TCP connections among the various home agents. However, this approach is difficult to implement because in an Ethernet environment, all connections with the same destination IP address will be directed to a single MAC address due to the broadcasting of the gratuitous ARP message. Also, multiple connections with the same source-destination pair are indistinguishable at the IP layer, and separating these connections would require processing at the TCP layer. The broadcast nature of the gratuitous ARP also prevents us from balancing the load based on the source of the connections. Therefore, in this paper, we will consider destination-based load balancing, i.e., balancing the load by appropriately assigning mobile hosts to the various home agents. In a static load balancing scheme, a mobile host will always be served by the same home agent, while in a dynamic load balancing scheme, the mobile host may be transferred among home agents.

3.1 The Proposed Protocol

In the proposed scheme, each mobile host will have the IP addresses of all the home agents in the home network. When the mobile host sends a registration request, it will randomly choose one of the home agents to service the request. As defined in the mobile-IP protocol, we assume that each mobile registration request has an unique identification and a lifetime which defines the time for which the registration is valid. The latter will be referred to as T_{reg}. The registration packet also contains the IP address of the selected home agent, the permanent IP address of the mobile host, and the temporary IP address of the mobile host. As part of the mobility binding, the agent maintains the following information: 1) the unique registration identifier, 2) the permanent IP address of the mobile host, 3) the temporary IP address of the mobile host, 4) a boolean variable PROXY which defines if proxy is on or off (if PROXY is on, then the home agent is acting as a proxy for the mobile host and responds with a proxy ARP reply whenever an ARP query is received for the mobile host, if PROXY is off, a different home agent is serving the mobile host), and 5) T_{reg} which defines the time for which the registration is valid. The structure of the binding cache and sample contents are shown in Fig. 3.

Registration Identification	Permanent IP Address	Temporary IP Address	PROXY (on/off)	T_{reg} (seconds)
012143659341	128.120.56.173	128.200.38.181	ON	300
0605259770704	128.120.56.37	128.200.38.73	OFF	300

Figure 3: Binding cache at a home agent.

With multiple home agents, load balancing will be achieved by allowing a home agent to transfer control of a mobile host to another home agent based on some load balancing algorithm. The load balancing algorithm consists of two parts: 1) a transfer policy which determines when a transfer should take place and 2) a selection policy which determines to which home agent the control should be transferred. The latter will yield an IP address of the destination home agent denoted by $\text{IP}_{Dest-HA}$. The key steps for initiating the transfer are as follows:

1. The source home agent reconstructs the mobile host's registration packet, puts $\text{IP}_{Dest-HA}$ in the registration packet, and sends the registration request to the destination home agent.
2. The source home agent turns PROXY off for the corresponding binding cache entry.

When a home agent receives a registration request it performs one of the following action:

1. If the identification of the registration request is different from the identification of all bindings already in the table, then the registration request is a new request, and the home agent performs one of two actions:

   - If the home agent address in the registration packet matches its own IP address, then it performs the following steps:
     - It creates a mobility binding with PROXY turned on.
     - It sends out a gratuitous ARP. As before this causes all hosts and routers in the home network to update their cache entries corresponding to the mobile host.
     - It sends a separate copy of the registration request to each of the other home agents.
   - If the home agent address in the registration packet is different from its own IP address, the home agent simply creates a mobility binding with PROXY turned off. Thus, all home agents will maintain a mobility binding for a given mobile host, but only one home agent will have PROXY turned on for that particular mobile host.

2. If the identification of the registration request is the same as the identification of a mobility binding already in its table, then the registration request is one that has been forwarded from another home agent. If the home agent address in the registration packet matches its own IP address, the home agent performs the following steps:

   - It turns PROXY on for the corresponding binding cache entry.
   - It performs a gratuitous ARP, causing all of the hosts and routers in the home network to update their cache entries so that the IP address of the mobile host will translate to the link layer address of the home agent. This will cause all packets destined for the mobile host to be redirected to this particular home agent.

When the T_{reg} timer expires, the entry is removed from the table. Finally, from the above description it is clear that for a particular mobile host, there is only one home agent that acts as proxy at any time.

3.2 Load Balancing Policies

The load balancing policy consists of a transfer policy to determine when a transfer should take place, and a selection policy to determine the destination home agent. We discuss these two orthogonal issues in this section.

We consider the following three approaches for determining when a transfer should be made:

1. Timer-Based: In this approach, each home agent maintains a timer for each mobile host that it is serving. The timer value will be referred to as the stream transfer time and will be denoted by T_{stt}. When a home agent acquires control of a mobile host, it starts the timer after the first packet for the mobile host is received. When T_{stt} expires, a new home agent is selected, and a registration request is forwarded to the new home agent following the algorithm described in Section 3.1.

2. Counter-Based: In this approach, the home agent counts the number of packets forwarded to each mobile host. When the counter reaches a specified limit, the home agent transfers the registration of the mobile host to another home agent. The counter value which triggers a transfer will be referred as the stream transfer counter and denoted by T_{stc}.

3. Threshold-Based: For each mobile host, the home agent maintains a count of the number of packets in its queue which are destined for the mobile host. When the number of packets in the queue for a given mobile host exceeds some threshold, the home agent forwards that mobile host's registration to another home agent. The threshold value will be referred to as the stream transfer threshold and will be denoted by T_{sth}.

Each transfer policy incurs some amount of overhead whenever a transfer takes place. The cost of a transfer includes both processing costs at each of the home agents, as well as the cost of additional network traffic. When a transfer takes place, the home agent that is initiating the transfer must generate a registration packet, select another home agent, and transmit the registration packet. The destination home agent must then process the registration, modify the appropriate binding cache entry, and send out a gratuitous ARP message. In this work, we model only the overhead associated with a home agent receiving and processing a registration packet, as we expect this processing time to be the most significant component of the overhead.

The timer-based policy may result in a higher number of transfers than the counter-based policy because in the timer-based policy, a mobile host can be transferred even when the mobile host is not receiving any packets. From a load balancing point of view, the threshold-based scheme is expected to perform the best because the transfers are based on the instantaneous load at each home agent.

The choice of the stream transfer parameter, namely, the timer value, the counter value, or the threshold value, determines how well the load is balanced across the multiple home agent and determines the amount of associated overhead. For example, if T_{stt} is very small, then transfers are done frequently, making it possible to randomly and uniformly distribute the load among the home agents on a packet-by-packet basis, thereby achieving high load balancing gains. However, this approach also incurs high overhead, since each time a stream is transferred to another home agent, the first home agent must generate and transmit a registration packet, while the receiving home agent must process the registration packet and broadcast a gratuitous ARP. On the other hand, if T_{stt} is large, then a mobile host is bound to the same home agent for a long time, which may result in poor load balancing.

The second issue to consider is the discipline for selecting the next home agent. The following common policies are considered:

1. Random Policy: In this scheme, the next home agent is selected randomly from all home agents. (To simplify the analysis, we assume that the selected home agent may include the home agent which is attempting the transfer).

2. Round-Robin Policy (RR): In this scheme, the home agents are logically ordered and the next home agent is selected using a simple round-robin policy.

3. Join the Shortest Queue (JSQ) Policy: In this scheme, the home agent which has the minimum number of queued packets is selected as the next home agent. Similar to the random policy, the current home agent may also be selected as the next home agent. Intuitively, JSQ should provide the best performance over other selection policies for a given transfer policy [5].

The random and round-robin policies are the simplest from an implementation point of view. The JSQ policy is difficult to implement in a shared media LAN, and it requires additional overhead. For JSQ, the queue length (or some equivalent load information) at the home agents must be maintained by all home agents.

In this paper we have developed a queueing model to analyze the performance of the random policy. The RR policy and the JSQ policy are analytically intractable and studied only using simulation.

4 Analytical Model

In order to provide some insight regarding the effect of stream transfer parameters, such as T_{stt}, on system performance, we develop an analytical model for a load balancing scheme with a timer-based transfer policy and a random selection-policy. The analytical model may also be useful for determining how to set parameters when designing and implementing the load balancing scheme. In the analysis, performance of the load balancing schemes is measured in terms of the latency at the home agents, with the cost of load balancing coming from the additional registration packets that are generated and handled by home agents. Costs due to increased network traffic are not considered, since it is expected that registration packets will consist of a small percentage of the overall network traffic.

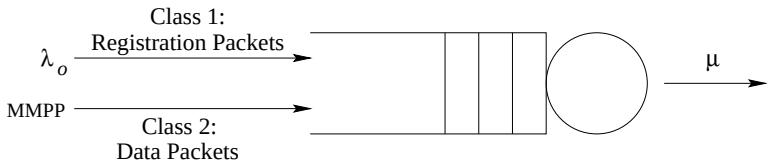

Figure 4: The queueing model for a home agent.

As shown in Fig. 4, each home agent is modeled as a single server queue with two classes of arrivals. Class 1 corresponds to registration requests that initiate a transfer of control, and Class 2 corresponds to data packets to be encapsulated and tunneled to the foreign agent. Class 1 requests have non-preemptive priority over Class 2 request. Since all home agents are identical, we will study the queue at a single home agent and aggregate the effect of all other home agents. We will refer to the selected home agent as the tagged home agent. In the queueing model we will make the following assumptions:

- There are N identical home agents.
- The service time of Class 2 packets is exponentially distributed with rate μ packets per second.
- The service time of the Class 1 packets is exponentially distributed with service rate μ/C packets per second. Thus, by changing C we can model different overhead costs which are represented in terms of Class 2 packet processing times.
- It is assumed that the processing overhead for a registration packet is fairly low, on the order of one data packet processing time.
- Each traffic source in the model represents a stream of packets destined for a single mobile host. Thus, a traffic source may be an aggregate of traffic from a number of hosts or routers which are sending packets to the same destination.

The Class 2 packets are generated by sources which consist of either hosts within the same network or routers that connect the network to the Internet. S denotes the number of identical sources. Each source is modeled as a 2-state Markov Modulated Poisson Process; the two states correspond to the on-state and the off-state. In the on-state, the packet generation follows a Poisson Process with parameter λ packets per second. In the off-state no packets are generated. The source turns on with rate σ_1 and turns off with rate σ_2, both following a negative exponential distribution. In order to take into account other home agents, we modify the model of each source by adding a third state as shown in Fig. 5. The three states are denoted as $S(0,0)$, $S(1,0)$, and $S(0,1)$. When

the process is in state $S(0,0)$, the source is off, and there are no arrivals. On the other hand, when the process is in state $S(1,0)$, the source is on, and arrivals occur to the tagged home agent according to a Poisson process with mean rate l packets/second. In state $S(0,1)$, the source is on, but it is sending its packets to a home agent other than the tagged home agent; consequently the arrival rate to the tagged home agent is zero in this state.

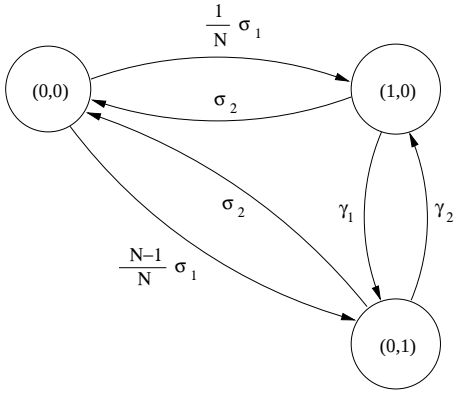

Figure 5: The modified three-state MMPP model of a source.

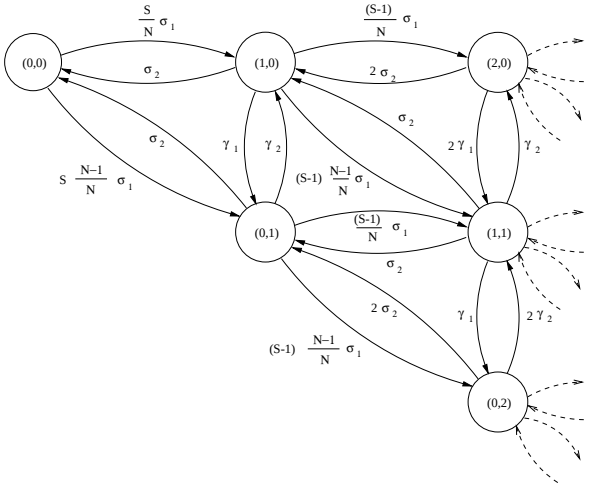

Figure 6: The state diagram of the superposition of S MMPP sources.

The transition rates between the states are also shown in Fig. 5. When a source turns on, it chooses one of the home agents randomly, which implies that the tagged home agent is selected with probability $1/N$. The transition from state $S(0,0)$ to $S(1,0)$ can then be modeled as having rate σ_1/N, while the transition from state $S(0,0)$ to state $S(0,1)$ has rate $\sigma_1(N-1)/N$.

When a source is transmitting to the tagged home agent, it will continue to transmit to that home agent for the duration of the stream transfer time, T_{stt}, after which it will randomly select one of the N home agents and begin transmitting to this new home agent. In order to make the analysis tractable, we approximate the time for which a source transmits packets to a home agent as having an exponential distribution with mean $T_{stt} + (1/\lambda)$. The $1/\lambda$ term results from the fact that the timer isn't started until a packet arrives to the home agent. Also, since the selected home agent may be the same as the tagged home agent, the source transfers to another home agent with probability $(N-1)/N$, and with probability $1/N$ remains with the tagged home agent. When the

source is transmitting to a home agent other than the tagged home agent, it will begin transmitting again to the original home agent after an average period of $N(T_{stt} + (1/\lambda))$ seconds. The transition rates from the state $S(1,0)$ to state $S(0,1)$ and from the state $S(0,1)$ to the state $S(1,0)$ are then given by γ_1 and γ_2 respectively, where

$$\gamma_1 = \frac{N-1}{N \cdot (T_{stt} + \frac{1}{\lambda})} \tag{1}$$

and

$$\gamma_2 = \frac{1}{N \cdot (T_{stt} + \frac{1}{\lambda})}. \tag{2}$$

Using the fact that the superposition of a number of MMPP sources is also an MMPP source, we may combine a number of these sources into a single MMPP. Combining S sources results in an MMPP with $(S+1)^2/2 + (S+1)/2$ states. The state diagram for this process is shown Fig. 6. A state $S(x,y)$ indicates that $x+y$ sources are on, with x sources sending packets to the tagged home agent and with y sources sending packets to other home agents. Ordering the states sequentially from top to bottom in each column, we obtain the following Q matrix for an MMPP with $S = 2$:

$$\begin{bmatrix} -S\sigma_1 & S \cdot \frac{\sigma_1}{N} & S \cdot \frac{(N-1)\sigma_1}{N} & 0 & 0 & 0 \\ \sigma_2 & -(S-1)\sigma_1 - \sigma_2 - \gamma_1 & \gamma_1 & (S-1) \cdot \frac{\sigma_1}{N} & (S-1) \cdot \frac{(N-1)\sigma_1}{N} & 0 \\ \sigma_2 & \gamma_2 & -\sigma_2 - \gamma_2 & 0 & (S-1) \cdot \frac{\sigma_1}{N} & (S-1) \cdot \frac{(N-1)\sigma_1}{N} \\ 0 & 2\sigma_2 & 0 & -2\sigma_2 - 2\gamma_1 & 2\gamma_1 & 0 \\ 0 & \sigma_2 & \sigma_2 & \gamma_2 & -2\sigma_2 - \gamma_1 - \gamma_2 & \gamma_1 \\ 0 & 0 & 2\sigma_2 & 0 & 2\gamma_2 & -2\sigma_2 - 2\gamma_2 \end{bmatrix} \tag{3}$$

We now need to account for Class 1 packets. As mentioned before, Class 1 corresponds to the transfer of registration packets between the home agents to transfer the control of a mobile host. Since these transfers occur during the on-period of a connection, the arrival process of these overhead packets is correlated with the on periods of Class 2 traffic. In this analysis, we assume that the two arrival processes are independent. Furthermore, instead of solving a two-priority queueing model, we will use the shadow server approximation proposed in [9] to analyze two-priority queues. In this approximation, we will aggregate the high priority traffic by appropriately modifying the service time of the Class 2 requests. This can be done by multiplying the service rate of data packets by $(1 - U_o)$ where U_o is the utilization of the server by Class 1 packets and is given by

$$U_o = \frac{\lambda_o}{\mu_o} \tag{4}$$

where λ_o is the aggregate arrival rate of Class 1 packets (overhead packets) to the tagged home agent from all the sources, and the service time for Class 1 packets is assumed to be exponential with service rate $\mu_o = \mu/C$.

At any given time, the expected number of sources in the on-period is given by

$$N_{on} = \frac{S \cdot \sigma_1}{\sigma_1 + \sigma_2} \tag{5}$$

This follows from the observation that the probability that a source is in the on-period is equal to $\sigma_1/(\sigma_1 + \sigma_2)$. In order to compute λ_o, we need to determine the number of overhead packets that are generated during the on-period of a source. The mean duration of an on-period is $1/\sigma_2$. During this period, a source generates overhead packets with rate $1/(T_{stt} + \frac{1}{\lambda})$ packets per second. Therefore, an average of $1/(\sigma_2(T_{stt} + \frac{1}{\lambda}))$ overhead packets arrive during an on-period. However,

an additional overhead packet is generated after the end of on-period (the last timer T_{stt} expires after the source has entered the off-period). On average, this final packet is generated at time T_{stt} after the on-period ends. The rate of overhead packets generated by a single stream, λ_{on}, is given by

$$\lambda_{on} = \frac{(\sigma_2 \cdot (T_{stt} + \frac{1}{\lambda}))^{-1} + 1}{\frac{1}{\sigma_2} + T_{stt}} \tag{6}$$

From Eq. (4) and Eq. (5) and using the argument that each home agent receives only $1/N$ of the overhead packets, λ_o is then given by

$$\lambda_o = \frac{S \cdot \sigma_1 \cdot (1 + \sigma_2 \cdot (T_{stt} + \frac{1}{\lambda}))}{N \cdot (\sigma_1 + \sigma_2)(T_{stt} + \frac{1}{\lambda})(1 + \sigma_2 T_{stt})} \tag{7}$$

The effective service rate is calculated as

$$\mu_{eff} = \mu - \frac{C \cdot S \cdot \sigma_1 \cdot (1 + \sigma_2 (T_{stt} + \frac{1}{\lambda}))}{N(\sigma_1 + \sigma_2)(T_{stt} + \frac{1}{\lambda})(1 + \sigma_2 T_{stt})} \tag{8}$$

Given the matrix Q in Eq. (2), the arrival rate while in each state, and the service time distribution, we can solve the MMPP/G/1 queueing model to find the mean response time. A closed from expression for the delay is not available. Instead we use a numerical approach for solving such models as given in [8].

5 Results

In order to validate the analytical model of the random selection policy derived in the previous section and to investigate the performance of alternative selection policies, we develop a simulation model of the network. The simulation model is based on the same assumptions as the analytical model with two exceptions. First, in the simulation model, each home agent is considered individually as a two-class non-preemptive priority queue. In the analytical model, on the other hand, a shadow-server approximation is used to model two-class traffic, and only a single home agent was considered. Aggregation was employed to consider the impact of other home agent on the specific tagged home agent. Second, in the simulation, the time between transfers is modeled as the sum of two exponential random variables with means T_{stt} and $1/\lambda$, while in the analytical model, this time was approximated as a single exponential random variable with mean $T_{stt} + 1/\lambda$.

We compare the performance of our load balancing scheme with two other schemes, namely, 1) the equal partition scheme and 2) the burst-level random partition scheme. The equal partition scheme is a static load balancing scheme, while the random partition scheme approximates a static random scheme. In all subsequent figures, the mean delay is normalized to the average data packet service time. We consider random, round-robin, and JSQ selection policies. The equal partition scheme and the burst-level load balancing scheme are described as follows:

- Equal Partition: The sources are divided evenly among all home agents, and each source is statically assigned to the same home agent for the duration of the simulation.

- Burst-Level Load Balancing: This scheme corresponds to the case in which T_{stt} is set to an infinite value and is used as a reference point for comparison with the proposed load balancing scheme. A transfer time of infinity would normally correspond to a mobile host being served by the same home agent for the lifetime of the registration. However, in our analytical model, when a burst is over (the source returns to state $S(0,0)$ in Fig. 5) the source does not retain

information about the home agent to which it was associated. Thus, when the source returns to the on-period, it randomly selects a new home agent. Therefore, in our results, a transfer time of infinity corresponds to load balancing on a burst-by-burst basis.

For the purpose of this study, we will define degree-of-burstiness, β, as the ratio of the peak arrival rate to the mean arrival rate. The mean arrival rate to a single home agent, λ_{mean}, is defined as

$$\lambda_{mean} = \frac{S \cdot \frac{\sigma_1}{\sigma_1+\sigma_2}\lambda}{N} \tag{9}$$

Thus $\beta = \lambda/\lambda_{mean}$. In our numerical examples, we will adjust the degree of burstiness by changing the parameters σ_1, σ_2, and λ, while keeping the mean arrival rate constant.

In Fig. 7, we plot delay as a function of T_{stt}. The various parameters are shown in the figure. The degree of burstiness β for this case is 1. The results are shown for the case in which there is no overhead ($C = 0$), and the case in which the overhead is equivalent to the service time of a single packet ($C = 1$). Results for the equal partition case, as well as for the burst-level load balancing scheme are also shown in the figure.

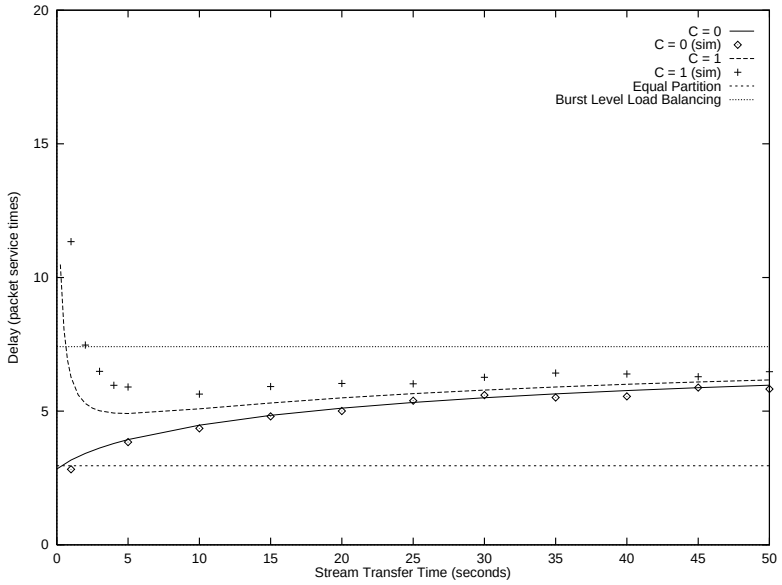

Figure 7: The mean packet delay as a function of the stream transfer time, T_{stt}, for the random selection policy. (S=4, N=2, λ=1 pkts/s, $\sigma_1 = \frac{1}{50}$, $\sigma_2 = \frac{1}{50}$, μ=2.5 pkts/s).

Following are the main observations:

- We see that as the transfer time decreases, the delay increases rapidly. This increase is caused by overhead packets that are generated every time a stream is transferred from one home agent to another. As the transfer time is increased, the delay drops due to less overhead. For high transfer times, the delay increases as the load becomes more unbalanced at the home agents.

- From the figure we also observe that the load balancing mechanism performs worse than the equal partition case, even when there is no overhead. The difference can be explained by the fact that for the random scheme, while on average each home agent is supporting an equal number of sources, there is a non-zero probability that a single home agent will be supporting more sources than the other home agents. This results in a higher variability in the number of packets at a given home agent, and thus a higher average delay.

- The simulation results are also shown in the figure. While the simulation results corroborate the analysis for most of the range, for low values of T_{stt}, the analysis underestimate the mean packet delay. This is due to the shadow server approximation that we have adopted to model the overhead.

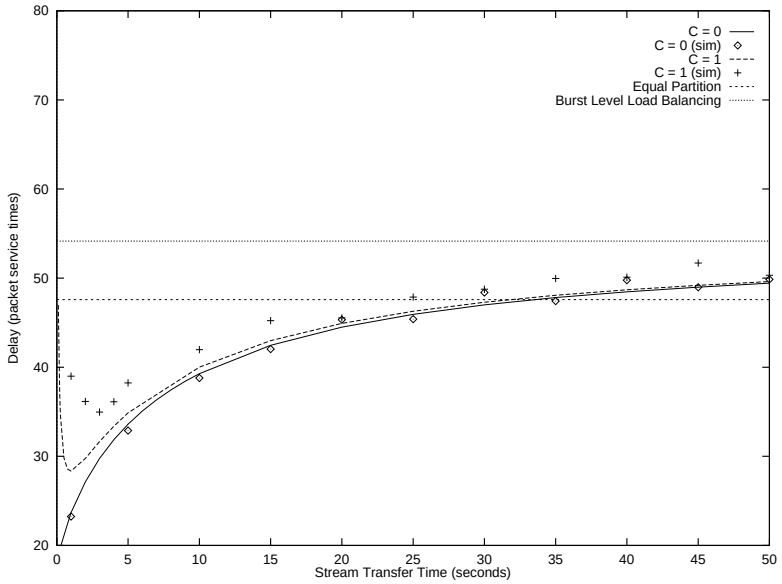

Figure 8: The mean packet delay as a function of the stream transfer time, T_{stt}, for the random selection policy. (S=4, N=2, λ=5 pkts/s, $\sigma_1 = \frac{1}{90}$, $\sigma_2 = \frac{1}{10}$, μ=2.5 pkts/s).

Figure 8 plots the packet delay as a function of T_{stt} for source traffic with higher degree of burstiness ($\beta = 5$). The following are the main observations:

- The load balancing gains are higher than the previous case in which $\beta = 1$. When traffic is very bursty, packets tend to build up in a queue quickly, resulting in high delays. By breaking up a burst and spreading the burst over a number of home agents, the queue at any single home agent doesn't grow as quickly. In general, load balancing gains are high when the peak burst arrival rate is higher than a home agent's service rate, or when there is a high probability that the aggregated arrival rate of multiple sources at any point in time is higher than the service rate.

- Unlike the previous case with $\beta = 1$, Equal Partitioning does not perform as well as the random selection policy even with overhead. Note that even with one source on, the arrival rate is higher than the service rate, resulting in higher delays. As T_{stt} is increased, the equal partitioning scheme performs better than the random selection policy. This is because the random selection policy approaches burst level load balancing as T_{stt} increases. With burst level load balancing, there is a non-zero probability that more than two sources will be sending bursts to a particular home agent, while for the equal partitioning scheme, each home agent is guaranteed to have at most two sources sending bursts.

- The simulation results do not match the analytical results for the case $C = 1$, particularly for lower values of T_{stt}. This is because of the approximation that we have used to model the overhead packets, which have a greater impact for lower values of T_{stt}.

We now plot the load balancing gain of the proposed scheme versus T_{stt}. The percent load balancing gain, G, is defined as the gain over the burst level load balancing case and is given by

$$G = (R_{bll} - R) \cdot 100/R_{bll} \tag{10}$$

where R is the mean packet delay for a given T_{stt}, and R_{bll} is the mean packet delay for the burst level load balancing scheme.

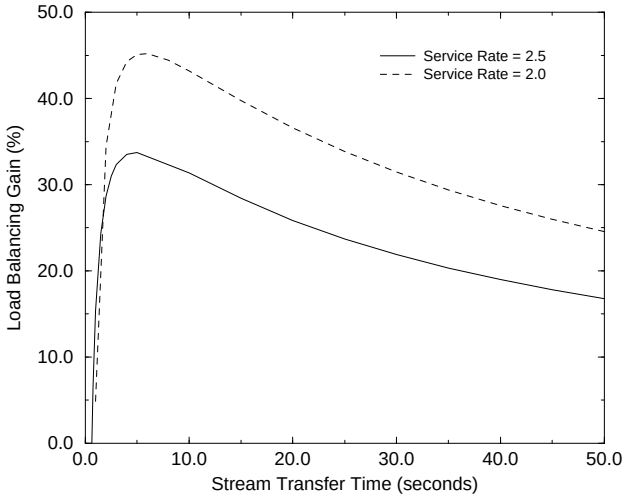

Figure 9: The percent load balancing gain, G, versus stream transfer time, T_{stt}, for different loads with the random selection policy (S=4, N=2, λ=1 pkts/s, $\sigma_1 = \frac{1}{50}$, $\sigma_2 = \frac{1}{50}$, μ=2.5 pkts/s, $C = 1$).

Figure 9 plots the load balancing gain, G, for two different values of load obtained by changing the packet service rate μ. The plots are shown for values of μ which correspond to $\rho(= \lambda_{mean}/\mu)$ of 0.4 and 0.5, respectively. From the plot we observe that higher load yields higher load balancing gains. The maximum gains are reasonably high (35% to 40%) even with a low degree of burstiness. Finally, note that the value of T_{stt} that maximizes the gain does not change much with higher load. However, when the load is increased to a point where all of the home agents are always busy, then we expect that transfers will not result in significant load balancing gains.

In Fig. 10, we plot load balancing gain for two different system sizes - one with 4 sources and 2 home agents and the other with 8 sources and 4 home agents. As the results show, the load balancing gains are higher for larger system sizes. This is because for larger systems, there is a higher probability that burst level load balancing will result in an uneven distribution of sources among home agents, creating higher delays. This situation allows for greater load balancing gains when the sources are allowed to transfer from one home agent to another.

6 Comparison of Load Balancing Policies

In this section, we investigate the effects of various policies for selecting the next home agent and for determining when to transfer a stream.

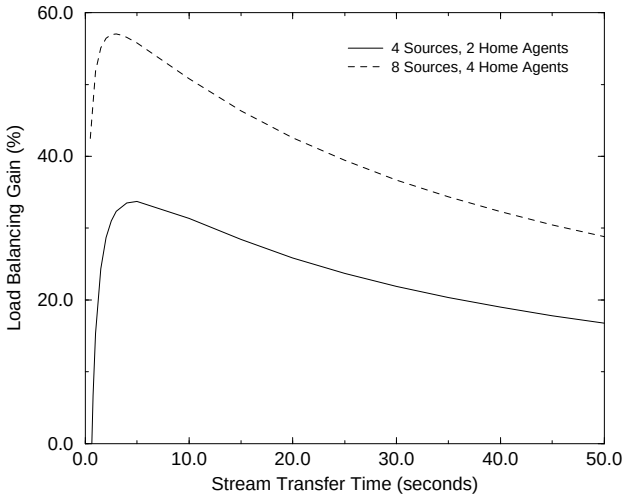

Figure 10: The percent load balancing gain, G, versus stream transfer time, T_{stt}, for different system sizes with the random selection policy (λ=1 pkts/s, $\sigma_1 = \frac{1}{50}$, $\sigma_2 = \frac{1}{50}$, μ=2.5 pkts/s, $C = 1$).

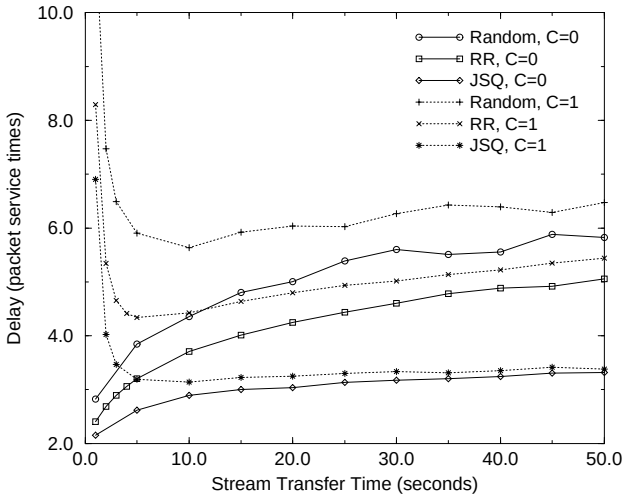

Figure 11: The mean packet delay as a function of T_{stt} for the random, round robin (RR), and JSQ policies ($S = 4$, $N = 2$, λ=1 pkts/s, $\sigma_1 = \frac{1}{50}$, $\sigma_2 = \frac{1}{50}$, μ=2.5 pkts/s).

6.1 Selection Policies

Three common selection policies are random, round-robin, and join the shortest queue (JSQ). Figure 11 plots the mean packet delay for the different selection policies with a timer-based transfer policy. Results are obtained using simulation. As the results show, the mean packet delay under JSQ is significantly lower than the random selection policy. Since the JSQ policy uses instantaneous state information to select the next home agent, the load balancing gains are higher. From the figure it should be noted that the limiting performance of JSQ and the random selection policies are different. This is because, in the limiting case in which T_{stt} is infinite, the random selection policy is equivalent to the burst level load balancing scheme. This implies that each burst is randomly allocated to a home agent. When T_{stt} is infinite, the JSQ policy is also equivalent to a burst level load balancing scheme. However, in the case of JSQ, the bursts are not randomly assigned, instead they are assigned based on the instantaneous queue length information. As a result, as T_{stt} is increased, JSQ converges to a much lower delay than the random selection policy.

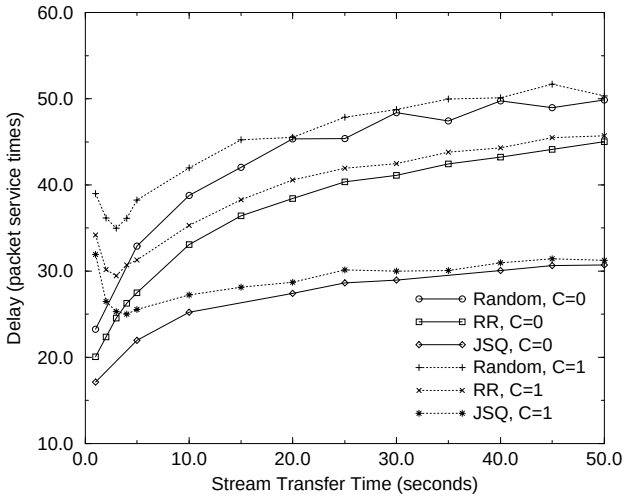

Figure 12: The mean packet delay as a function of T_{stt} for the random, round robin (RR), and JSQ policies ($S = 4$, $N = 2$, λ=5 pkts/s, $\sigma_1 = \frac{1}{90}$, $\sigma_2 = \frac{1}{10}$, μ=2.5 pkts/s).

Figure 12 plots the mean packet delay for a higher degree of burstiness; β is equal to 5. The results are consistent with the previous observation that a higher β implies higher load balancing gains, both for the case of the random selection policy, as well as for the case of the JSQ policy.

6.2 Transfer Policies

In this section, we consider the performance of three different transfer policies with a random selection policy. The transfer policies studied are the timer-based policy, the counter-based policy, and the threshold-based policy. These policies are described in Section 3.2. In Figs. 13, 14, and 15, we plot average delay for the various transfer policies with $\beta = 1$ ($\lambda = 1, \sigma_1 = \frac{1}{50}, \sigma_2 = \frac{1}{50}$) and $\beta = 5$ ($\lambda = 5, \sigma_1 = \frac{1}{90}, \sigma_2 = \frac{1}{10}$).

Note that each transfer policy uses a different transfer parameter on which to base the transfer decision. The timer-based approach uses a timer value (T_{stt}), the counter-based approach uses a counter value (T_{stc}), and the threshold-based scheme uses a queue threshold (T_{sth}). To gain some

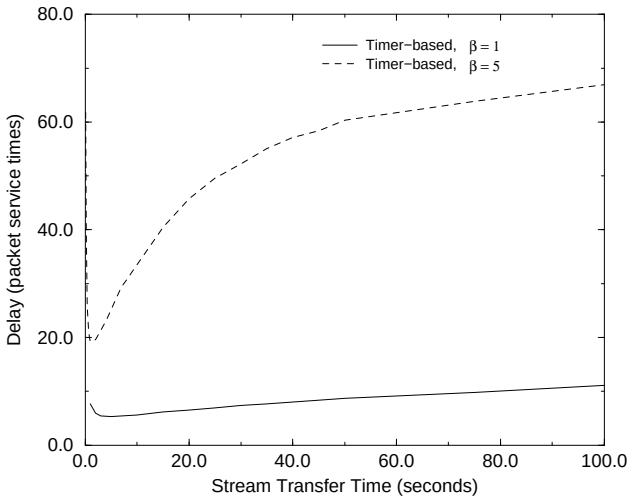

Figure 13: The mean packet delay as a function of T_{stt} for the timer-based random policy ($S = 8$, $N = 4$, μ=2.5 pkts/s, $C = 1$).

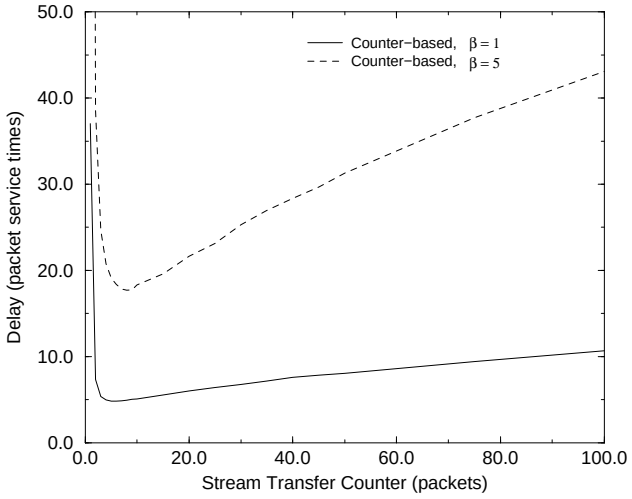

Figure 14: The mean packet delay as a function of T_{stc} for the counter-based random policy ($S = 8$, $N = 4$, μ=2.5 pkts/s, $C = 1$).

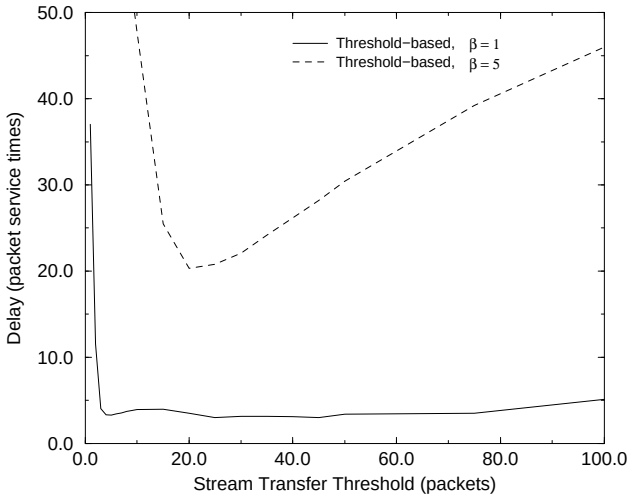

Figure 15: The mean packet delay as a function of T_{sth} for the threshold-based random policy ($S = 8$, $N = 4$, μ=2.5 pkts/s, $C = 1$).

insight as to how the parameters for the timer-based policy and the counter-based policy relate, assume that the arrival rate of packets for a given mobile host is $1/\lambda$. In the timer-based policy, the mean time between transfers is then $\frac{1}{\lambda} + T_{stt}$, while in the counter-based policy, the mean time between transfers is $\frac{1}{\lambda} \cdot T_{stc}$. Thus, the counter-based policy is more responsive to changes in the arrival rate.

For the case in which each source has a low peak arrival rate ($\beta = 1$), the threshold policy offers better performance than the timer or counter-based policies. This performance is explained by noting that the threshold policy will only transfer a stream when the queue becomes overloaded, while the other policies may transfer a stream even if the home agent isn't overloaded; thus the threshold policy not only incurs less overhead, but also tends towards a situation in which streams are equally balanced among the home agents. This result is confirmed in Table 3, which indicates that if the transfer parameter (queue threshold) is set high enough, no transfers take place (after a few possible initial transfers), but a low delay is still maintained (Fig. 15).

T_{stt}	5	10	50	100	500
$\beta = 1$	0.0867	0.0496	0.0136	0.0079	0.0019
$\beta = 5$	0.0240	0.0150	0.0075	0.0055	0.0017

Table 1: Average number of transfers per second for the timer-based policy.

If the peak arrival rate of a single source is higher than the service rate at a home agent ($\beta = 5$), then a single source may be able to overwhelm the home agent, causing its queue to build up during a burst arrival. In this case, even if the streams are equally balanced among home agents, further load balancing gains may still be achieved through stream transfers by spreading a burst among several home agents. We see that for the timer and counter based policies, the transfer parameter should be set to a low value in order to ensure that a burst from a single source

T_{stc}	5	10	50	100	500
$\beta = 1$	0.1000	0.0500	0.0100	0.0050	0.0010
$\beta = 5$	0.0996	0.0502	0.0100	0.0050	0.0010

Table 2: Average number of transfers per second for the counter-based policy.

T_{sth}	5	10	50	100	500
$\beta = 1$	0.0198	0.0023	0.0000	0.0000	0.0000
$\beta = 5$	0.3043	0.1359	0.0043	0.0009	0.0000

Table 3: Average number of transfers per second for the threshold-based policy.

is divided evenly among the home agents. However, for the threshold policy, a low queue threshold incurs significantly more overhead than the other two transfer policies. The reason for this behavior is that if a stream returns to a home agent which it has already visited, then the selected home agent may still have packets from the given stream in its queue, in which case a transfer may take place sooner than expected, leading to a higher number of transfers. This is verified in Table 3.

7 Conclusions and Future Work

When the mobile-IP protocol is deployed, subnets which are supporting a large number of mobile hosts will need to have multiple mobility agents in order to provide an adequate level of service. In this paper we presented a means of evenly distributing the load among multiple home agents in mobile-IP. By providing a mechanism which allows incoming packet streams to be transferred from one home agent to another, we may achieve gains over schemes in which each packet stream is only served by a single home agent. The gains depend on the periodicity with which the transfers are performed and the policy that is that is used to select the next home agent. Results show that even the simple random selection policy can yield modest load balancing gains (30% to 55%) over static schemes. The gains are higher when the traffic is bursty, i.e., when the peak-to-mean ratio is high. As expected, the JSQ policy, which selects the next home agent based on the instantaneous queue lengths at each home agent, performs much better than the random selection policy.

This work may be extended to the case in which the underlying network is an ATM network. In this situation, IP to ATM address translations are performed by an entity known as the Address Resolution Protocol server (ARP server). This centralized server may be used to implement better load balancing schemes by keeping track of ATM connections and by balancing the load on an ATM connection level.

References

[1] C. E. Perkins, "Mobile IP," IEEE Communications Magazine, vol. 35, no. 5, pp. 84-99, May 1997.

[2] C. E. Perkins and D. B. Johnson, "Route Optimization in Mobile-IP," Draft-IETF-mobileip-optim-06.txt, July 1997.

[3] A. Giovanardi, and G. Mazzini, "Transparent Mobile IP: an Approach and Implementation," IEEE Globecom 1997, Pheonix, Arizona, November 1997.

[4] S. Deering, "ICMP Router Discovery Message", RFC 1256, September 1991.

[5] D. Eager, E. Lazowska, and J. Zahorjan, "Adaptive load sharing in homogeneous distributed systems," IEEE Trans. Software Engineering, vol. SE-12, pp. 662–675, May 1986.

[6] R. Mirchandaney, D. Towsley, and J. A. Stankovic, "Analysis of the effect of delays on load sharing," IEEE Transactions on Computers, vol. 38, no. 11, pp. 1513-1525, November 1989.

[7] A. N. Tantawi and D. Towsley, "Optimal Static Load Balancing in Distributed Computer Systems," Journal of the Association for Computing Machinery, vol. 32, no. 2, pp. 445-465, April 1985.

[8] W. E. Leland and T. J. Ott, "Load-balancing Heuristics and Process Behavior," Proceedings of Performance 86 and ACM SIGMETRICS 86, Raleigh, NC, USA, May 1986.

[9] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook," Performance Evaluation 18, pp.149-171, 1992.

[10] K. C. Sevcik, "Priority scheduling disciplines in queueing network models for computer systems," in Proc. IFIP Congress. Amsterdam: North-Holland, 1977.

[11] LAN Emulation SWG Drafting Group, "LAN Emulation Over ATM: Draft Specification - Revision 2", ATM Forum/94-0035R2+, April 15, 1994.

[12] W. Richard Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Addison-Wesley Professional Computing Series, 1994.

[13] E. W. Knightly, "Second moment resource allocation in multi-service networks," 1997 ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 97), Seattle, WA, USA, June 1997.

[14] C.-J. Hou and K. F. Shin, "Implementation of decentralized load sharing in networked workstations using the Condor package," Journal of Parallel and Distributed Computing, vol.40, no. 2, pp. 173-84, February 1997.

[15] C. Huitema, Routing in the Internet, Prentice Hall PTR, Englewood Cliffs, New Jersey 07632, 1995.

[16] J. Ioannidis, D. Duchamp, and G. Q. Maguire Jr., "IP-based Protocols for Mobile Internetworking," ACM SIGCOMM'91, September 1991.