An Efficient Fault-Tolerant Approach for Mobile IP in Wireless Systems

Jenn-Wei Lin and Joseph Arul, *Member*, *IEEE*

Abstract—This paper presents the fault tolerance of Mobile IP in wireless systems. Mobile IP can support wireless users with continuous network connections while changing locations. It is achieved by allocating a number of mobility agents (foreign agents and home agents) in the architecture of a wireless system. If a failure occurs in a mobility agent, the wireless users located in the coverage area of the faulty mobility agent will lose their network connections.To tolerate the failures of mobility agents, this paper proposes an efficient approach to maintaining the network connections of wireless users without being affected by the failures. Once detecting a failure in a mobility agent, failure-free mobility agents are dynamically selected to be organized as a backup set to take over the faulty mobility agent. Compared to the previous approaches, the proposed approach does not take any actions against failures during the failure-free period. Besides, the hardware redundancy technique is also not used in the proposed approach. The overhead of the proposed approach is analyzed using the *M/G/c/c* queuing model. The results show that the proposed approach can effectively resolve the fault-tolerant problem of Mobile IP in wireless systems.

Index Terms—Fault tolerance, Mobile IP, wireless systems, *M/G/c/c* queuing model.

1 Introduction

Due to the rapid progress of wireless communication technology, there is a growing demand for accessing data by wireless systems (wireless data accesses). Mobility is one important characteristic of wireless systems. Each wireless user (mobile node) may change its location several times during the execution of its data service. To avoid interrupting the ongoing data session, the Internet Engineering Task Force (IETF) has defined Mobile IP [1] as one which enables wireless users to maintain ongoing data sessions without interruption while changing locations.

To provide the functionality of Mobile IP in a wireless system, a number of mobility agents are required in the architecture of the wireless system. The mobility agents are classified into two types: foreign agent (FA) and home agent (HA). The FA logically connects with several radio access networks (RANs) to form a wireless data serving area. The HA maintains the current addresses of mobile nodes (MNs). When an MN starts a data session, a data request is first sent to the located RAN of the MN, then to a serving FA and, finally, to a corresponding application server. The application server processes the data request and sends back the packets of the handling results to the MN. The packets sent to the MN will be first intercepted by the MN's corresponding HA. The HA looks up the current address of the MN and tunnels the packets to the FA serving the MN. Then, the FA detunnels the packets and forwards them to the MN. From this scenario of a wireless data session, we can know that, if a failure occurs in a FA, all MNs located in its data serving area cannot perform wireless data sessions again. Likewise, if an HA crashes, the response packets of the data session cannot be sent back to the corresponding MN.

This paper presents an efficient approach to providing fault-tolerant capability in the wireless system with Mobile IP functionality. The fault-tolerant importance is described as follows: For a telecom system, "five nines" (99.999 percent) reliability is a usual requirement for the network design [2]. To achieve such high reliability, the failure rate of equipment in the telecom system may need to be very low. However, the failure rate varies as time, where the relationship can be modeled as a bathtub curve [3]. In the increasing part of the bathtub curve, the failure rate of telecom equipment increases as time increases. This means that telecom equipment will be prone to incur failures after it has been functional for a long period of time. The wireless system is also one telecom system. For supporting the "five nines" reliability, it is necessary to provide fault-tolerant solutions for a wireless system.

The proposed approach is based on the concept of resource sharing to redirect the workloads of a faulty FA (HA) to other *failure-free FAs (HAs)*. Unlike the previous approach [4], [5], the fault-tolerant method of an FA in the proposed approach is different from that of an HA. Once detecting a failure in a FA, one or more failure-free FAs are dynamically selected to be organized as a backup set for the faulty FA. Then, a system-initiated handoff is issued to virtually move the MNs now served by the faulty FA to the serving areas of the FAs in the backup set. The data executable capabilities for the MNs can be continuously supported by the failure-free FAs in the backup set. For the fault tolerance of an HA, if a failure is detected in an HA, one or more failure-free HAs are also dynamically selected to be the backup members of the faulty HA. The selected failure-free HAs, instead of the faulty HA, intercept the

• *The authors are with the Department of Computer Science and Information Engineering, Fu Jen Catholic University, 510 Chung Cheng Road, Hsinchuang, Taipei 242 Taiwan, ROC.*
E-mail: {jwlin, arul}@csie.fju.edu.tw.

Manuscript received 18 Aug. 2002; revised 5 June 2003; accepted 25 June 2003.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number 7-082002.

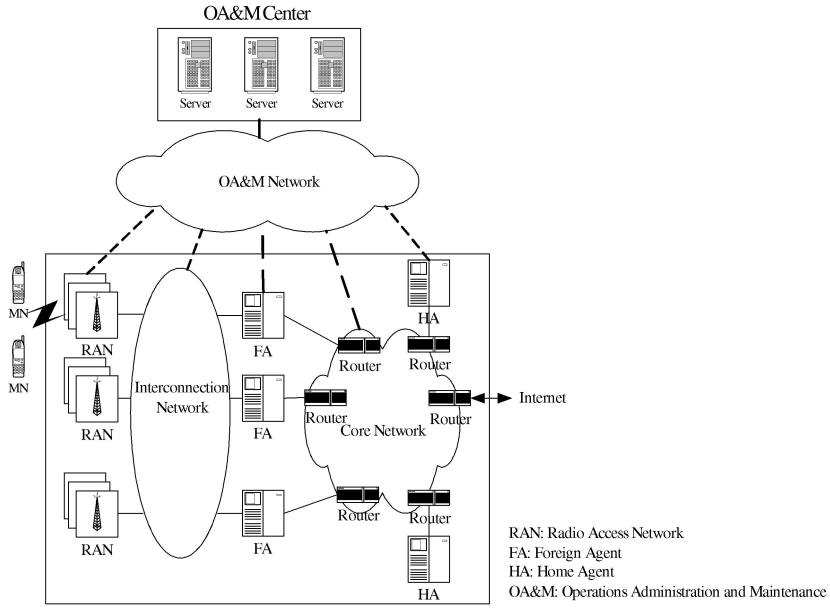

Fig. 1. Wireless data network model.

packets moving toward the faulty HA, and then send the packets to corresponding MNs.

The remainder of this paper is organized as follows: Section 2 presents the background of this paper. Section 3 proposes our approach. Section 4 describes the details of implementing the proposed approach. Section 5 evaluates the overhead of the proposed approach. Section 6 makes the comparison between the proposed approach and previous approaches. Finally, conclusions are made in Section 7.

2 Background

This section describes the background materials of this paper. First, we give an overview of the Mobile IP. Then, the system model is described. Next, the fault assumption is made. Finally, related work is reviewed.

2.1 Mobile IP Overview

Mobile IP is designed to support node mobility by using two types of mobility agents: home agent (HA) and foreign agent (FA). Initially, each mobile node (MN) has a unique address to be managed by an HA on its home network. If an MN moves from its home network to a foreign network, a "care-of-address" (COA) is allocated to the MN to reflect the MN's current point of attachment. The FA on the foreign network also adds an entry corresponding to the MN in its visitor list. Then, a register message is sent to the MN's HA to create or modify the MN's mobility binding which associates the MN's home address, the located FA, and the COA. Packets from a correspondent host (CH) to an MN are addressed to the home address of the MN. If the MN is away from its home network, the HA on the home network can intercept the packets and tunnel them to the located FA of the MN. Then, the FA decapsulates the packets and forwards them to the MN.

2.2 System Model

The system model considered in this paper refers to the architecture of a third generation (3G) wireless system [6], [7], as shown in Fig. 1. The system model contains three major components: mobile node (MN), radio access network (RAN), and core network. The MN interacts with an RAN to obtain radio resources for performing wireless data sessions. The RAN provides the transmission across the air interface. With the core network, it is an IP-based network and contains the following equipment: foreign agent (FA), home agent (HA), and intermediate routers. The FA and HA provide the wireless data sessions with Mobile IP functionality. The intermediate routers assist FAs and HAs to forward packets. In addition, between RANs and FAs, there is an interconnection network to deliver the wireless data requests from MNs to the core network and to send back the response packets from the core network to MNs. Based on the specifications of [8], [9], [10], the interconnection network can be a Frame Relay network, ATM network, or IP network, respectively.

To manage all equipment in the wireless system, operations, administration, and maintenance (OA&M) functions are necessary. The OA&M functions are classified into configuration management, fault management, performance management, and security management [11]. The configuration management configures equipment (e.g., RAN, FA, HA, AAA server, router, etc.) with suitable resource parameters. The performance management measures the resource utilization, loading status, and other concerned values in the equipment. The fault management is capable of detecting and reporting failures in the equipment. The security management monitors the access rights to the equipment.

2.3 Fault Assumption

In this paper, failures are only assumed to occur in mobility agents. To announce the presence of a mobility agent (FA or

HA), the mobility agent periodically transmits an agent advertisement message on its located subnet [1]. When a failure occurs in a mobility agent, the failure can be detected by not receiving an agent advertisement message within a period of time. The fail-stop assumption is also made so that the faulty mobility agent cannot send any agent advertisement message again.

As an FA fails, its in-processing data requests will be lost. For a faulty HA, its in-processing response packets are also lost. However, these lost data requests and response packets can be retransmitted by an end-to-end reliable transport layer such as Transmission Control Protocol (TCP). Several modifications have been proposed to improve TCP in wireless systems [12], [13]. In this paper, we do not discuss how to resend the lost data requests and response packets.

2.4 Related Work

For Mobile IP, many fault-tolerant approaches have been proposed [4], [5], [14], [15], [16], [17]. However, only [4] and [5] deal with the issue of tolerating failures in mobility agents. According to [4], a mobility agent is statically equipped with one or more redundant mobility agents as its backup set. The mobility agent can cooperate with its backup set to work in the standby or load-sharing model. If a mobility agent fails, one member in its backup set will be selected to act as the primary mobility agent. Here, the Address Resolution Protocol (ARP) [18] is used to map the IP address of the faulty mobility agent onto the network link-layer address of the selected backup member. Other backup members still cowork with the new primary mobility agent, but the total processing capability is less one. However, in the approach of [4], there is nontrivial overhead for preserving the mobility binding information. When an MN registers with a mobility agent, the registration is required to be additionally done on all backup members. This possible long registration delay will introduce significant performance degradation.

To avoid the long registration delay, the approach of [5] suggests checkpointing and logging techniques to store the mobility bindings in stable storage. After a mobility agent fails, its backup is also selected from one of its equipped redundancies. Unlike the approach of [4], the backup does not have the mobility bindings of the faulty mobility agent. It needs to restore the mobility bindings from stable storage. The stable storage is assumed to be invulnerable. Basically, the approach of [5] is similar to the approach of [4].

3 The Proposed Approach

This section presents a new approach to tolerating the failures of mobility agents. To avoid significant performance degradation, the approach dynamically selects multiple failure-free mobility agents to form a backup set for the faulty mobility agent. The workloads of the faulty mobility agent are redirected to the failure-free mobility agents in the backup set.

3.1 Fault Tolerance of Foreign Agent

Upon detecting a failure in an FA, the MNs located in its serving area will not be able to execute wireless data sessions. The new arriving MNs are also affected. These MNs are called as *FA_failure-affected MNs*. To resume the data executive abilities of the FA_failure-affected MNs, a system-initiated handoff is performed to dynamically select multiple failure-free FAs to organize a backup set for the faulty FA. Then, the FA_failure-affected MNs are virtually moved to the serving areas of the selected failure-free FAs. Each of the selected failure-free FAs adds a number of visitor entries for the FA_failure-affected MNs that have moved to it and informs those MNs' corresponding HAs of the new serving FAs and the care-of-addresses to update their mobility bindings. After performing the system-initiated handoff, the network connections of the FA_failure-affected MNs can be continuously supported by their new serving FAs. The workloads of the faulty FA are redirected to other failure-free FAs.

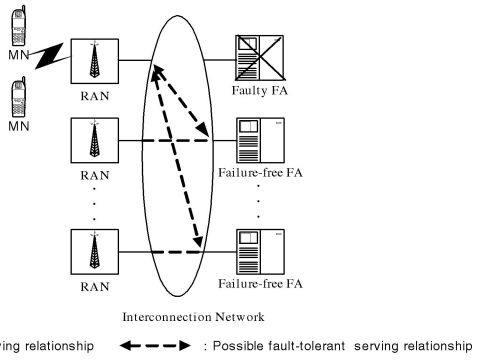

Fig. 2. Remapping the relationship between RANs and FAs for fault tolerance.

Originally, a handoff functions to switch an MN's wireless communication and maintains its network connection when the MN moves from one radio coverage area to another. In the proposed approach, the system-initiated handoff is to redirect the workloads of the faulty FA, but it does not introduce location changes of the FA_failure-affected MNs. This objective is attained by modifying the relationship between RANs and FAs, described as follows:

As shown in Fig. 1, there is an interconnection network to connect RANs with FAs. From the logic point of view, there exists at least one delivery path from an RAN to each FA. Therefore, if an RAN receives a data request from a MN, it can deliver the data request to any FA to process it. The RAN determines which FA to process its received data requests based on its internal *FA-serving* record. Initially, the FA-serving record of an RAN is set to be the identifier of a fixed FA. After detecting a failure in an FA, one or more failure-free FAs are selected to be the backup members of the faulty FA. Then, the FA-serving records of the *failure-affeced RANs* (the RANs which have an initial serving relationship with the faulty FA) are reset as the identifiers of the backup members. By modifying the FA serving records, the serving FAs of the failure-affected RANs become the backup members. Therefore, the FA_failure-affected MNs are now served by the backup members, but their locations are not changed (they still locate in their respective radio coverage areas), as shown in Fig. 2.

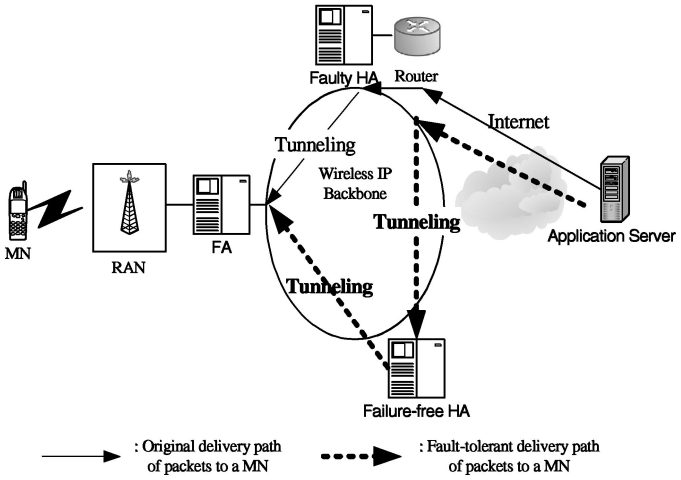

Fig. 3. Packets destined to an HA_failure-affected MN.

3.2 Fault Tolerance of Home Agent

An HA assists a correspondent host (CH) to send packets to an MN based on the following three functions: mobility binding maintenance, packet interception, and packet tunneling. If an HA fails, the *HA_failure-affected MNs* (the MNs managed by the faulty HA) will not be able to receive packets from CHs. To resume the packet receiving abilities of the HA_failure-affected MNs, one or more failure-free HAs are also dynamically selected to be the backup members of the faulty HA. Then, the above three functions of the faulty HA are restored on the backup members, described as follows:

As mentioned in Section 2.1, when an MN enters into a data serving area, the corresponding FA adds a visitor entry in its visitor list to record the following information: the MN's data link-layer address, IP address, and home agent address. When the MN leaves the data serving area, its corresponding visitor entry is made invalid. Therefore, if an MN has a valid entry in an FA's visitor list, the MN is now located in the serving area of the FA. From the FAs' visitor lists, the up-to-date locations of all MNs can be known. This points out that the mobility bindings of the faulty HA can be restored by searching all the FAs' visitor lists. The restored mobility bindings are then distributed to the backup members. With regard to the packet interception of the faulty HA, it is restored on the backup members using the tunneling technique. Usually, one or more routers are collocated with an HA on the same network segment to connect the HA with the core network and the Internet. When an CH sends packets to an MN, the packet is first received by the collocated routers, and is then intercepted by the MN's HA. After detecting a failure in an HA, its collected routers do not forward their received packets to the faulty HA. Instead, the collected routers tunnel the packets to the backup members, as shown in Fig. 3. In such a case, a packet from a CH to an HA_failure-affected MN is sent by twice tunneling. One is to let one backup member intercept the packet. Another is to tunnel the packet to the located FA of the HA_failure-affected MN.

As for the restoration of the packet tunneling, the failure-free HAs selected as the backup members, already possess the tunneling function.

4 Implementation

The implementation of the proposed approach can be integrated into the OA&M. As described in Section 2.2, there are four main functions in the OA&M. When a failure in an FA (HA) is detected, the failure event is sent to the fault management. The fault management initiates the proposed fault-tolerant approach for the FA (HA). At first, it interacts with the performance management to acquire the loading status of the failure-free FAs (HAs). Based on this loading information, multiple failure-free FAs (HAs) with low traffic are selected to be the backup members of the faulty FA (HA). Then, the configuration management is informed to configure the backup members of the faulty FA (HA) by resetting appropriate parameters to some equipment in the core network. Thereafter, the incoming workloads of the faulty FA (HA) are redirected to its backup members.

4.1 Implementation of Foreign Agent

To understand the implementation in more detail, the fault-tolerant procedure for the FA is given in Fig. 4. There are three main tasks in the procedure. The first task

```
Do not receive an agent advertisement message sent from a FA within a period of time
Send a FA failure event to the fault management
/*Collecting the load status of each failure-free FA*/
Ask the performance management to respond with the loading status of each failure-free
FA
/*Calculating the number of FA_failue-affected MNs*/
FA_failure-affected_MNs←0   --number of FA_failure-affected MNs
For each RAN RAN_i managed by the faulty FA (each failure-affected RAN)
    n←Number of associations for data sessions in the RAN
    FA_failure-affected_MNs←FA_failure-affected_MNs+n
End
```

(a)

Fig. 4. The fault-tolerant procedure for the FA. (a) Collecting the loading status of failure-free FAs and the number of FA_failure-affected MNs.

```
MNs_assigned←0   --number of MNs already assigned their new serving FAs
lower_mark←true
For each failure-free FA_i
    available_resources[i] ←the number of available resources in the FA_i
    resources_to_use[i] ←0 --number of resource units to FA_failure-affected MNs
End
Repeat
 For each failure-free FA_i
  If (lower_mark) --select the backup members from the FAs with lower traffic
    If (available_resources[i]≥a threshold)
      Call Be_Backup_Member
    End
  Else --need to select the backup members from the FAs with high traffic
      Call Be_Backup_Member
  End
 End
 lower_mark←false
Until MN_assigned≥FA_failure-affected_MNs
/* Specify a failure-free FA to be a backup member*/
Subroutine Be_Backup_Member
  resources_donated←r × available_resources[i]
  /* r is the ratio of donating available resources for FA_failure-affected MNs*/
  If (resources_donated>0)
     Specify the FA_i as a backup member
     available_resources[i] ←available_resources[i] - resources_donated
     resources_to_use[i] ←resources_to_use[i] + resources_donated
     MNs_assigned←MNs_assigned + resources_donated
  End
End
```

(b)

Fig. 4. (Continued). (b) Selecting the backup members.

is to collect the loading status of each failure-free FA and calculate the number of FA_failure-affected MNs, as shown in Fig 4a. The loading status of each failure-free FA can be collected from the performance management of OA&M. For the number of FA_failure-affected MNs, it can be calculated by counting the number of association entries in the failure-affected RANs. In an RAN's coverage area, if an MN issues a wireless data session, a corresponding entry is generated in the RAN to record the association between the RAN and the FA serving the MN. The second task selects multiple failure-free FAs to be the backup members of the faulty FA, as shown in Fig. 4b. First, the failure-free FAs with lower traffic are selected. If the total resources donated by the selected failure-free FAs are insufficient to serve all failure-affected MNs, some failure-free FAs with higher traffic will be selected to be the backup members. In addition, each selected failure-free FA also estimates the number of FA_failure-affected MNs to be virtually moved to its serving area based on the number of the available resources in it.

```
For each RAN located under the faulty FA's serving area (each failure-affected RAN)
    New_Serving_FA_Liste←null
    MNs_moving←Number of MNs currently located within the RAN's coverage area
    For each backup member failure-free FA_i
       IF (resources_to-use[i]>0)   --resources_to_use[i] has derived from Fig. 4(b)
  /*There are still donated resources in the FA_i for FA_failure-affected MNs*/
          New_Serving_FA_List←Serving_FA_list∪the identifier of FA_i
          IF (MNs_moving ≤ resources_to-use[i])
            /*All FA_failure-affected MNs in the RAN can be served by the FA_i */
             resources_to_use[i] ←resources_to_use[i] - MNs_moving
             MNs_Moving←0
             Exit
          Else
  /*Only a part of FA_failure-affected MNs in the RAN can be served by the FA_i */
              MNs_moving←MNs_moving - resources_to_use[i]
              resources_to_use[i] ←0
           End
       End
    End
    Distribute the MNs located under the RAN to be served by the new FAs indicated in
    the New_serving_FA_list.
    For each FA_i in the New_serving_FA_list
       Add the visitor entries corresponding to the MNs assigned to be served by it.
       Update the mobility bindings of such MNs to record their new serving FAs
    End
End
```

(c)

Fig. 4. (Continued). (c) Performaing the system-iniated handoff.

The third task, as shown in Fig. 4c, performs the system-initiated handoff both to virtually move FA_failure-affected MNs to the serving areas of the backup members and to update the mobility bindings of such MNs. From Section 3.1, we know that the virtual moves of the FA_failure-affected MNs are achieved by modifying the FA-serving records of the failure-affected RANs. However, the resources of a backup member are derived from a portion of the available resources in a failure-free FA. It is possible that the MNs in the failure-affected RAN cannot be fully served by a single backup member. Each failure-affected RAN first needs to specify enough backup members to serve the MNs in its coverage area. Then, it resets its FA-serving record to the identifiers of the specified backup members. Next, each specified backup member adds the visitor entries for the FA_failure-affected MNs served by it. Meanwhile, the HAs of the FA_failure-affected MNs are notified to update their mobility bindings. Thereafter, the failure-affected RANs can deliver their received data requests to the backup members. In addition, the packets destined to the FA_failure-affected MNs can be tunneled to the backup members.

4.2 Implementation of Home Agent

There are also three main tasks in the procedure for implementing the fault tolerance of the HA. Based on the HAs' loading status given by the OA&M, the first task selects multiple failure-free HAs with low traffic to be the backup members of the faulty HA. One of the backup members is specially designated as the HA's backup manager.

The second task restores the mobility bindings of the faulty HA by sending a mobility-reconstruction message to each FA. Upon receiving the message, each FA searches its visitor list to find the visitors entry with the identifier of the faulty HA. Each qualified visitor entries is then reorganized as the form of a mobility binding entry and sends back to the HA's backup manager. The HA's backup manager divides its received mobility binding entries into several groups based on their corresponding MNs' IP addresses and then sends each group to a backup member, as shown in Fig. 5a. If a backup member is designated to maintain the

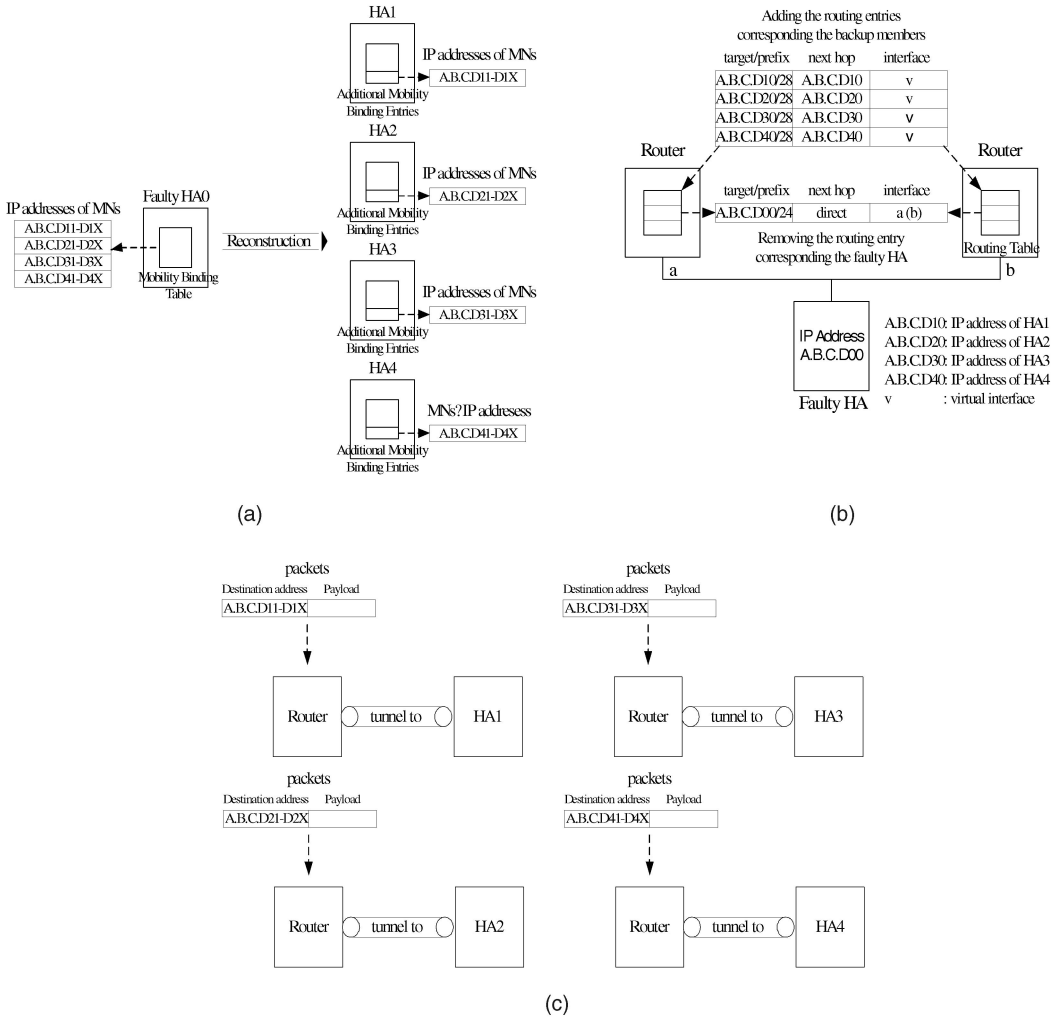

Fig. 5. An example of tolerating the HA failure. (a) Reconstructing the mobility bindings. (b) Changing the packet interceptor. (c) Redirecting the packet interception.

mobility bindings of MNs with IP addresses in the range $[IP_1, IP_2]$, it will take the responsibility for intercepting the packets whose destination addresses belong to $[IP_1, IP_2]$, and then tunneling such packets.

The third task is to change the packet interceptors of the HA_failure-affected MNs from the faulty HA to the backup members. To change the packet interceptors, the collocated routers of the faulty HA must remove their routing entries corresponding to the faulty HA and add the routing entries corresponding to the backup members, as shown in Fig. 5b. However, the backup members may not have direct physical interfaces with the collocated routers. In each routing entry corresponding to a backup member, its interface field is set to a virtual interface. The virtual interface points to a software program to perform the packet tunneling. By tunneling, the packets destined to a HA_failure-affected MN are intercepted by backup members (HA1 - HA4), not the faulty HA (HA0), as shown in Fig. 5c.

4.3 Failure Recovery

When a faulty FA is recovered from failure, the FA_failure-affected_MNs can be served back by the recovered FA. The recovery procedure is shown in Fig. 6a. First, the recovered FA determines the information which RANs are preserved by it (the information about the failure-affected RANs). Then, the failure-affected RANs reset their FA-serving records to the identifier of the recovered FA. The recovered FA gets back to be the common serving FA of all FA_failure-affected MNs. Next, the recovered FA creates a lot of visitor entries for the FA_failure-affected MNs. In addition, the respective HAs of the FA_failure-affected MNs are also notified to update the MNs'mobility bindings.

The recovery procedure for an HA is shown in Fig. 6b. When a faulty HA is recovered, the responsibility of packet interception can be returned to it. By modifying the routing tables of the collocated routers of the recovered HA, the packet interceptors of HA_failure-affected MNs are changed back from the backup members to the recovered HA. In addition, the mobility bindings of the recovered HA are also required to be reconstructed, which can be done by searching all FAs' visitor lists. This reconstruction task is similar to the task of restoring the mobility bindings of the faulty HA (see Section 4.2).

Retrieve the information about failure-affected RANs from the configuration management of OA&M
/* Let the recovered FA be the serving FA of the FA_failure-affected MNs*/
For each failure-affected RAN_i
  Reset the serving-FA record of RAN_i to the identifier of the recovered FA
  For each association of a data sessions in RAN_i
    $MN_i \leftarrow$Mobile Node corresponding to the association
    Add a visitor entry of MN_i in the recovered FA
    Send a mobility binding update to modify the serving FA of MN_i to be the recovered FA
  End
End

(a)

/* Let the recovered HA be the packet interceptor of the HA_failure-affected MNs*/
For each Router Ru_i collocated with the recovered HA on the same network segment
  Add a routing entry corresponding to the recovered HA into Ru_i's routing table
  Delete the router entries corresponding to the backup members from Ru_i's routing table
End
/* Reconstruct the mobility bindings of the recovered HA*/
For each FA FA_i in the wireless system
  Search the visitor entries of FA_i to select the entries with the identifier of the recovered HA being set in their HA-address fields
  Re-organize each qualified entry as the form of a mobility binding entry, and send it back to the recovered HA
End

(b)

Fig. 6. The failure recovery procedure. (a) FA and (b) HA.

5 Evaluation

The proposed fault-tolerant approach redirects the workloads of a faulty FA (HA) to failure-free FAs (HAs). To perform the workload redirection, some control messages are issued, which also introduce a certain overhead on the system performance. This section analyzes the performance degradation of a failure-free FA (HA) and the control message overhead.

To obtain the two desired overheads, the parameter notations are first defined in Table 1. The data requests sent to an FA and the response packets intercepted by an HA are assumed to follow Poisson processes, but the service time of a data request and the processing time of a packet tunneling are not assumed to obey any specific distribution. Based on these traffic distributions, the traffic behavior of an FA and that of an HA can both be modeled as the *M/G/c/c* queuing model [19].

5.1 Performance Degradation of a Mobility Agent

As described in Section 3, the performance degradation of a failure-free FA is due to the system-initiated handoff which virtually moves some FA_failure-affected MNs to its serving area. Therefore, the resources of a failure-free FA are now contended by the MNs virtually moved and the MNs originally located. The performance of a failure-free FA incurs a certain degradation, which can be represented in terms of the following metric:

- Increasing blocking probability $P_{FA\_Blocking}$ that causes a new data request to be more possibly blocked at a failure-free FA in comparison to prefailure.

  The probability $P_{FA\_Blocking}$ is derived as follows: The arrival rate of data requests to a failure-free FA is λ_{FA}. Using the well-known *Erlang's loss formula* from the *M/G/c/c* queuing model [19], the blocking probability of a data request to a failure-free FA is:

$$\frac{\frac{\left(\frac{\lambda_{FA}}{\mu_{FA}}\right)^{c_{FA}}}{c_{FA}!}}{\sum_{i=0}^{c_{FA}} \frac{\left(\frac{\lambda_{FA}}{\mu_{FA}}\right)^{i}}{i!}}. \quad (1)$$

TABLE 1
Parameter Notations

Parameter	Meaning
N_{Agent} (N_{FA} or N_{HA})	Number of mobility agents (FAs or HAs) in the wireless system
R_{Agent}	Number of redundancies in a network segment for the approaches of [4] and [5]
F_{Agent} (F_{FA} or F_{HA})	Number of faulty mobility agents (FAs or FAs) in the wireless system
c_{Agent} (c_{FA} or c_{HA})	Number of resource units for handling data in a mobility agent (FA or HA)
λ_{Agent} (λ_{FA} or λ_{HA})	Arrival rate of data to a mobility agent (FA or HA)
μ_{Agent} (μ_{FA} or μ_{HA})	Service rate of data in a mobility agent (FA or HA)
w_{FA_k} (w_{HA_k})	Ratio of redirecting the workloads of the faulty FAs (HAs) to the failure-free FA_k (HA_k), where $\sum_{k=1}^{N_{FA}-F_{FA}} w_{FA_k}=1$ ($\sum_{k=1}^{N_{HA}-F_{HA}} w_{HA_k}=1$)

After some FAs fail, the data requests to the faulty FAs are redirected to failure-free FAs. Compared to prefailure, the arrival rate of data requests to the failure-free FA_k is larger and becomes as $\lambda_{FA} + F_{FA} \times \lambda_{FA} \times w_{FA_K}$ if there are F_{FA} faulty FAs. The new blocking probability is as follows:

$$\frac{\frac{\left(\frac{\lambda_{FA}+F_{FA}\times\lambda_{FA}\times w_{FA_k}}{\mu_{FA}}\right)^{c_{FA}}}{c_{FA}!}}{\sum_{i=0}^{c_{FA}}\frac{\left(\frac{\lambda_{FA}+F_{FA}\times\lambda_{FA}\times w_{FA_k}}{\mu_{FA}}\right)^{i}}{i!}}. \quad (2)$$

From (1) and (2), $P_{FA\_Blocking}$ can be derived as follows:

$$P_{FA\_Blocking}=\frac{\frac{\left(\frac{\lambda_{FA}+F_{FA}\times\lambda_{FA}\times w_{FA_k}}{\mu_{FA}}\right)^{c_{FA}}}{c_{FA}!}}{\sum_{i=0}^{c_{FA}}\frac{\left(\frac{\lambda_{FA}+F_{FA}\times\lambda_{FA}\times w_{FA_k}}{\mu_{FA}}\right)^{i}}{i!}}-\frac{\frac{\left(\frac{\lambda_{FA}}{\mu_{FA}}\right)^{c_{FA}}}{c_{FA}!}}{\sum_{i=0}^{c_{FA}}\frac{\left(\frac{\lambda_{FA}}{\mu_{FA}}\right)^{i}}{i!}}. \quad (3)$$

Likewise, the performance degradation of a failure-free HA_k is represented in terms of the following metric:

- Increasing blocking probability ($P_{HA\_Blocking}$) that causes an intercepted packet to be more possibly blocked at a failure-free HA in comparison with prefailure.

The derivation of $P_{HA\_Blocking}$ is similar to $P_{FA\_Blocking}$, which is represented as:

$$P_{HA\_Blocking}=\frac{\frac{\left(\frac{\lambda_{HA}+F_{HA}\times\lambda_{HA}\times w_{HA_k}}{\mu_{HA}}\right)^{c_{HA}}}{c_{HA}!}}{\sum_{i=0}^{c_{HA}}\frac{\left(\frac{\lambda_{HA}+F_{HA}\times\lambda_{HA}\times w_{HA_k}}{\mu_{HA}}\right)^{i}}{i!}}-\frac{\frac{\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)^{c_{HA}}}{c_{HA}!}}{\sum_{i=0}^{c_{HA}}\frac{\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)^{i}}{i!}}. \quad (4)$$

5.2 Control Message Overhead

In the proposed approach, the following control messages are issued from the OA&M for assisting the fault tolerance of the FA (HA):

- *FA_Loading*: Collecting the loading status of each failure-free FA for reducing the performance degradation of a failure-free FA.
- *RAN_Mapping*: Remapping the relationship between the RANs and the FAs for virtually moving the FA_failure-affected MNs.
- *Binding_Update*: Updating the mobility bindings of FA_faiure-affected MNs for registering their new serving FAs.
- *HA_Loading*: Collecting the loading status of each failure-free HA for reducing the performance degradation of a failure-free HA.
- *Interceptor_Change*: Modifying the routing tables for changing the packet interceptors of HA_failure-affected MNs.
- *Binding_Restoration*: Searching the visitor list of each FA for restoring the mobility bindings of a faulty HA.

The first three control messages are issued for tolerating the FA failure. The first control message inquires of each failure-free FA its loading status. The introduced cost of the message includes the transmission time ($T_{FA\_Loading}$) of sending the message and the transmission time ($T_{FA\_\mathrm{Re}\,sponse}$) of responding the FA's loading status. The second control message notifies each failure-affected RAN to modify its FA-serving record. Since the modification of the FA-serving record is a single memory access operation, the introduced cost of the message can be only determined by the transmission time ($T_{RAN\_Mapping}$) of sending the message to an RAN. The third control message indicates the backup members of a faulty FA to send the registration messages for updating the mobility bindings of FA_failure-affected MNs. The introduced cost of the message is estimated as follows:

$$\begin{aligned} &T_{Mobility\_Update}+T_{\mathrm{Re}\,g}\\ &=T_{Mobility\_Binding}+(N_{FA\_MN}\times t_{\mathrm{Re}\,g})\times f_{FA\_MN}, \end{aligned} \quad (5)$$

where $T_{Mobility\_Update}$ is the transmission time of sending a command of the mobility binding update to each FA (note that the command can be simultaneously sent to each FA from the OA&M), and $T_{\mathrm{Re}\,g}$ is the total required time for the

failure-free FAs to send the mobility binding updates about all FA_failure-affected MNs. N_{FA_MN} is the total number of the FA_failure-affected MNs, and $t_{\mathrm{Re}\,g}$ is the average transmission time of sending a registration message from an FA to an HA. Since each FA can simultaneously send the registration messages to the corresponding HAs, $T_{\mathrm{Re}\,g}$ is only required a fraction (f_{FA_MN}) of the time $N_{FA\_MN} \times t_{\mathrm{Re}\,g}$ (the time for updating the mobility bindings of all FA_failure-affected MNs using the serial transmission).

The last three control messages are issued for tolerating the HA failure. The fourth control message inquires of each failure-free HA its loading status. The introduced cost of the message includes the transmission time ($T_{HA\_Loading}$) of sending the message and the transmission time ($T_{HA\_\mathrm{Re}\,sponse}$) of responding to the HA's loading status. The fifth control message notifies the collocated routers of a faulty HA to modify their routing tables to add and delete routing entries. Since the routing table of a router is stored in memory, the modification of the routing tables involves several trivial memory access operations. The introduced cost of this control message is only dependent on the transmission time ($T_{Inteceptor\_Change}$) of sending the message to a collocated router of the faulty HA. The last control message indicates each FA to search its visitor list to select the visitor entries corresponding to the HA_failure-affected MNs. Since the FA's visitor list is also stored in memory, the cost for searching the visitor list can be ignored. For example, the searching time of a visitor list with 10,000 entries is only 0.032 sec. The introduced cost of the last control message is:

$$\begin{aligned} &T_{Mobility\_Binding} + T_{Entry} \\ &= T_{Mobility\_Binding} + (N_{HA\_MN} \times t_{Entry}) \times f_{HA\_MN}, \end{aligned} \tag{6}$$

where $T_{Mobility\_Binding}$ is the transmission time of sending a command of the mobility binding restoration to each FA, and T_{Entry} is the total required time for restoring the lost mobility binding table of the faulty HA. N_{HA_MN} is the total number of the HA_failure-affected MNs, t_{Entry} is the average time of sending a qualified visitor entry, and f_{HA_MN} is a fraction. The derivation of (6) is similar to that of (5).

The time metrics: $T_{FA\_Loading}$, $T_{FA\_\mathrm{Re}\,sponse}$, $T_{RAN\_Mapping}$, $T_{Mobility\_Update}$, $T_{HA\_Loading}$, $T_{HA\_\mathrm{Re}\,sponse}$, $T_{Inteceptor\_Change}$, and $T_{Mobility\_Binding}$ are mainly dependent on the transmission delays between the OA&M and the FAs, RANs, HAs, and routers in the core network. The OA&M and the equipment in the core network are connected through an OA&M network, as shown in Fig. 1. The physical interfaces of the OA&M network in a commercial wireless system are usually equipped with high-speed lines (T1 or above). In addition, the sizes of the messages corresponding to the above time metrics are also small. The costs of the above time metrics can be neglected.

To conclude, the control message overhead for tolerating the FA failure and that for the HA failure ($Control\_Over_{FA}$ and $Control\_Over_{HA}$) are mainly dependent on the mobility binding updates of the FA_failure-affected MNs and the mobility binding restoration of the HA_failure-affected MNs, respectively. Both the overheads have been estimated as $(N_{FA\_MN} \times t_{\mathrm{Re}\,g}) \times f_{FA\_MN}$ and $(N_{HA\_MN} \times t_{Entry}) \times f_{HA\_MN}$. With respect to N_{FA_MN} and N_{HA_MN}, they can be represented as the following equations:

$$N_{FA\_MN} = F_{FA} \times \sum_{n=0}^{c_{FA}} n \times P_{FA\_n} \tag{7}$$

$$N_{HA\_MN} = F_{HA} \times \sum_{n=0}^{c_{HA}} n \times P_{HA\_n}, \tag{8}$$

where $P_{FA\_n}(P_{HA\_n})$ is the probability that there are n in-processing data requests (response packets) in a faulty FA (HA). If each MN is not allowed to simultaneously issue more than one data session, $P_{FA\_n}(P_{HA\_n})$ can be represented by the probability that there are n FA_failure-affected MNs (HA_failure-affected MNs) in a faulty FA (HA). If the condition is not supported, the N_{FA_MN} and N_{HA_MN} are less than (7) and (8), respectively. From [19], the formulas of P_{FA_n} and P_{HA_n} has been also given. $Control\_Over_{FA}$ and $Control\_Over_{HA}$ can be further deduced as:

$$Control\_Over_{FA} = \left(F_{FA} \times \sum_{n=0}^{c_{FA}} n \times \frac{\frac{\left(\frac{\lambda_{FA}}{\mu_{FA}}\right)^n}{n!}}{\sum_{i=0}^{c_{FA}} \frac{\left(\frac{\lambda_{FA}}{\mu_{FA}}\right)^i}{i!}} \right) \times t_{\mathrm{Reg}} \times f_{FA\_MN} \tag{9}$$

$$Control\_Over_{HA} = \left(F_{HA} \times \sum_{n=0}^{c_{HA}} n \times \frac{\frac{\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)^n}{n!}}{\sum_{i=0}^{c_{HA}} \frac{\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)^i}{i!}} \right) \times t_{Entry} \times f_{HA\_MN}. \tag{10}$$

6 Comparison

Table 2 makes a comparison between the proposed approach and the approaches of [4] and [5] with respect to hardware redundancy, fault-tolerant range, fault-tolerant overhead, failure-free overhead, and failure-recovery overhead.

The approaches taken in [4] and [5] are based on the hardware redundancy to equip a primary mobility agent with several redundancies in a network segment. There are two alternatives: standby and load-sharing to configure the coworking mode between the primary mobility agent and its redundancies. After a primary mobility agent fails, the two coworking modes in the above approaches take the ARP to select one redundancy as the new primary mobility agent, but they have different degrees of performance degradation on the selected redundancy. In the standby mode, the blocking probability of the selected redundancy before a failure is 0, since it does not handle any workload. After a failure, the selected redundancy alone handles all the workloads in the network segment. Based on (3), the performance degradation imposed on the selected redundancy can be represented as:

TABLE 2
Comparison between the Proposed Approach and the Approaches of [4] and [5]

Comparison Metrics	Approach in [4]	Approach in [5]	Proposed Approach	
Hardware redundancy	Yes	Yes	No	
Fault-tolerant range	Restricting in a network segment	Restricting in a network segment	Extending to the whole system	
Failure-free overhead	long registration delay	Checkpointing and logging	No	
Failure-tolerant overhead	ARP execution and performance degradation	ARP execution, performance degradation, and mobility binding restoration	FA	Mobility binding update and performance degradation
			HA	Mobility binding restoration and performance degradation
Failure-recovery overhead	ARP execution	ARP execution and mobility binding reconstruction	FA	Mobility binding update
			HA	Mobility binding Reconstruction

$$P_{Agent\_Blocking} = \frac{\frac{\left(\frac{\lambda_{Agent}}{\mu_{Agent}}\right)^{c_{Agent}}}{c_{Agent}!}}{\sum_{i=0}^{c_{Agent}} \frac{\left(\frac{\lambda_{Agent}}{\mu_{Agent}}\right)^{i}}{i!}}. \quad (11)$$

In the load-sharing mode, the selected redundancy is responsible for more workloads than other redundancies in the same network segment since it needs to additionally handle the workloads of the original primary mobility agent. The performance degradation imposed on the selected redundancy is:

$$P_{Agent\_Blocking} = \frac{\frac{\left(\frac{2\times\frac{\lambda_{Agent}}{(1+R_{Agent})}}{\mu_{Agent}}\right)^{c_{Agent}}}{c_{Agent}!}}{\sum_{i=0}^{c_{Agent}} \frac{\left(\frac{2\times\frac{\lambda_{Agent}}{(1+R_{Agent})}}{\mu_{Agent}}\right)^{i}}{i!}} - \frac{\frac{\left(\frac{\frac{\lambda_{Agent}}{(1+R_{Agent})}}{\mu_{Agent}}\right)^{c_{Agent}}}{c_{Agent}!}}{\sum_{i=0}^{c_{Agent}} \frac{\left(\frac{\frac{\lambda_{Agent}}{(1+R_{Agent})}}{\mu_{Agent}}\right)^{i}}{i!}}, \quad (12)$$

where $(1 + R_{Agent})$ is the number of mobility agents (primary mobility agent and its redundancies) in a network segment, and $\frac{\lambda_{Agent}}{(1+R_{Agent})}$ is the arrival rate of data to a mobility agent based on the load-sharing mode. After the failure, the arrival rate of data to the selected mobility agent is $2 \times \frac{\lambda_{Agent}}{(1+R_{Agent})}$.

For the fault-tolerant range of the two approaches, if the primary mobility agent and its equipped redundancies fail simultaneously, the redundancies in other network segments (subnets or LANs) cannot be used to tolerate such failure situation since the ARP is only available in the LAN environment [19]. The fault-tolerant range is restricted in a network segment. In addition, for restoring the mobility bindings, the approach of [4] incurs a long registration delay during the failure-free period. The approach of [5] involves a lot of time-consuming operations for saving mobility bindings in stable storage. In the aspect of the failure-recovery overhead, both the approaches use the ARP to activate the recovered mobility agent. The reconstruction of the mobility bindings for the approach of [4] and that for the approach of [5] are done by retrieving the lost mobility bindings from one redundancy and the stable storage, respectively. Basically, the failure-recovery overhead of each of the two approaches is similar to its respective fault-tolerant overhead but the performance degradation.

In contrast to the previous approaches, the proposed approach does not incur the hardware and failure-free overheads. Also, unlike the approaches of [4] and [5], the backup members of a faulty FA (HA) are not necessary to be located with the faulty FA (HA) on the same network segment. In theory, the proposed approach can simultaneously tolerate $N - 1$ faulty FAs (HAs) if there are N FAs (HAs) in the system. However, for tolerating a faulty FA (HA), the failure-free FAs (HAs) selected to be the backup members incur a certain degree of performance degradation due to handling a portion of workloads of the faulty FA (HA). Two additional fault-tolerant overheads are also incurred due to the issues of control messages (see Section 5.2). As for the failure recovery, its procedure is similar to the fault-tolerant procedure (see Section 4.3). The incurred failure-recovery overhead is nearly same as the fault-tolerant overhead except the performancedegradation.

From Table 2, we can further observe that the approaches of [4] and [5] and the proposed approach give a different trade off among the above given comparison metrics. Among all the approaches, the approach of [4] takes the least failure-recovery overhead. The approach of [5] improves the approach of [4] to avoid the long registration delay, but it has to take time-consuming operations to save mobility bindings in stable storage. Comparatively, the proposed approach has obvious advantages on the hardware overhead, fault-tolerant range, and failure-free overhead. In the aspect of the fault-tolerant overhead, it is difficult to make a comparison. Intuitively, the cost of the ARP execution is trivial. However, all three approaches

TABLE 3
Analytical Parameters

Parameter	Values
N_{Agent} (N_{FA} or N_{HA})	10
R_{Agent}	1,2, and 3
F_{Agent} (F_{FA} or F_{HA})	1, 2, 4, and 8
c_{Agent} (c_{FA} or c_{HA})	50
$\frac{\lambda_{Agent}}{\mu_{Agent}}$ ($\frac{\lambda_{FA}}{\mu_{FA}}$ or $\frac{\lambda_{HA}}{\mu_{HA}}$)	10, 50, 100, and 200

have the performance degradation. Next, we perform the analytical comparison to quantify the differences of the three approaches on the performance degradation.

6.1 Analytical Comparison

The analytical parameters used are specified in Table 3. $\frac{\lambda_{Agent}}{\mu_{Agent}}$ denotes the expected number of arrivals per mean service time at a mobility agent (FA or HA); it is used to represent the traffic intensity of a mobility agent. The values of the traffic intensity are, respectively, set to be 10, 50, 100, and 200. The four chosen values of the traffic intensity all supply large workloads to a mobility agent (FA or HA) [20].

As mentioned before, the performance degradation is in terms of the increasing blocking probability of a failure-free mobility agent. Fig. 7 illustrates the comparison of the increasing blocking probabilities in the approaches of [4] and [5] and the proposed approach as a function of F_{Agent}, when $\frac{\lambda_{Agent}}{\mu_{Agent}} = 10,\ 50,\ 100,$ and 200. For the proposed approach, the workloads of faulty mobility agents (faulty FAs or HAs) are evenly redirected to all the failure-free mobility agents since the same traffic distribution is made in each mobility agent. The ratio $w_{FA_k}(w_{HA_k})$ of redirecting workloads to a failure-free mobility agent is $\frac{1}{N_{Agent}-F_{Agent}}$. For the approaches of [4] and [5], there are two coworking modes for the primary mobility agent and its redundancies. From (11) and (12), we know that if the load-sharing mode is adopted, the number of redundancies in a network segment affects the increasing blocking probability. To observe the effects, the number of redundancies equipped in a network segment is respectively set to 1, 2, and 3.

As shown in Fig. 7, it is clear that the increasing blocking probability for the proposed approach increases as F_{Agent} increases. If F_{Agent} increases, the workloads of the faulty FAs (HAs) redirected to the failure-free FAs (HAs) become more. However, the increasing blocking probability for the approaches of [4] and [5] is independent of the variance of F_{Agent} since each redundancy is responsible for taking over only one faulty mobility agent regardless of how many faulty mobility agents in the same network segment. In the aspect of $\frac{\lambda_{Agent}}{\mu_{Agent}}$, the increasing blocking probability for the approaches of [4] and [5] increases as $\frac{\lambda_{Agent}}{\mu_{Agent}}$ increases since only one redundancy is selected to take over one faulty mobility agent. When $\frac{\lambda_{Agent}}{\mu_{Agent}}$ is larger,

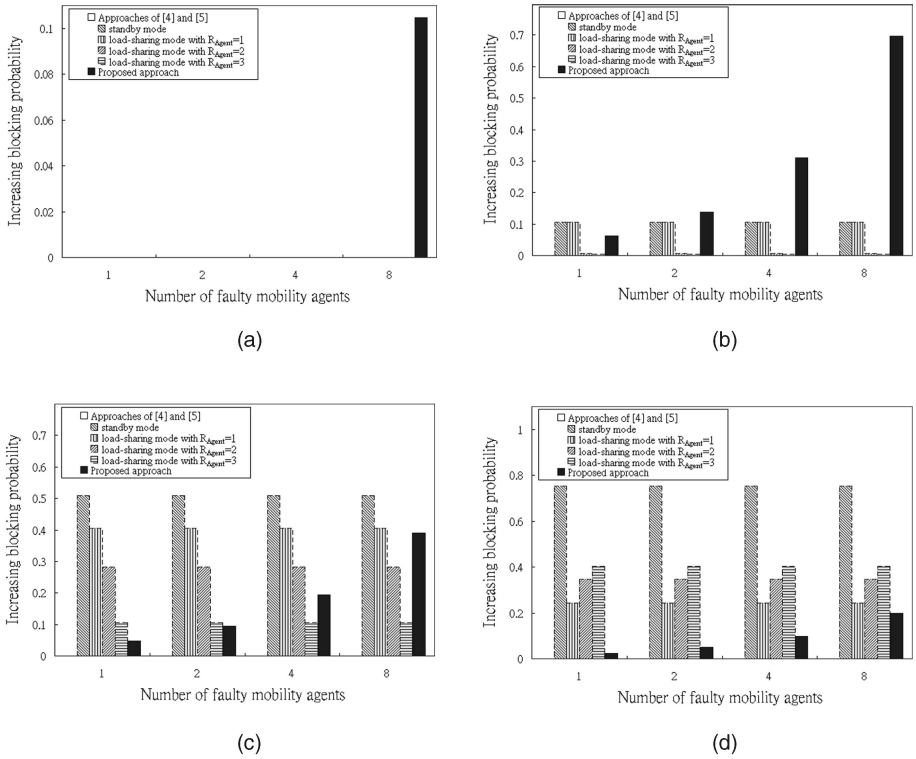

Fig. 7. Comparison of increasing blocking probablity under various numbers of faulty mobility agents. (a) $\frac{\lambda_{Agent}}{\mu_{Agent}} = 10$. (b) $\frac{\lambda_{Agent}}{\mu_{Agent}} = 50$. (c) $\frac{\lambda_{Agent}}{\mu_{Agent}} = 100$. (d) $\frac{\lambda_{Agent}}{\mu_{Agent}} = 200$.

the additional workloads to the selected redundancy become more. For the proposed approach, the increasing blocking probability does not always increase as $\frac{\lambda_{Agent}}{\mu_{Agent}}$ increases. For example, the increasing blocking probability for the proposed approach at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 200$ is less than that at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 50$ and 100. The situation is explained as follows: When $\frac{\lambda_{Agent}}{\mu_{Agent}} = 200$, the blocking probability (0.7516) of a failure-free FA (HA) before a failure is obviously larger than that (0.1048) at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 50$ and that (0.5093) at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 100$. After some FAs (HAs) fail (e.g., $F_{Agent} = 8$), the new blocking probability of a failure-free $FA(HA)$ at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 200$ becomes 0.9501, which is not much larger than that (0.8010) at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 50$ and that (0.9002) at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 100$. Therefore, the increasing blocking probability (0.9501 - 0.7516 = 0.1985) of a failure-free FA (HA) at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 200$ is smaller than that (0.8010 - 0.1048 = 0.6962) at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 50$ and that (0.9002-0.5093=0.3909) at $\frac{\lambda_{Agent}}{\mu_{Agent}} = 100$.

For the comparison results of the increasing blocking probabilities in all the three approaches, when $\frac{\lambda_{Agent}}{\mu_{Agent}} = 10$, the increasing probability for the approaches of [4] and [5] is close to 0 regardless of the variance of F_{Agent}. In this traffic intensity, the resource units in a redundancy are sufficient to handle the workloads to it since c_{Agent} is set to 50. The increasing blocking probability of a redundancy is very small. For the proposed approach in such traffic intensity, if $F_{Agent} < 8$, the increasing blocking probability is also very small. If $F_{Agent} = 8$, the increasing blocking probability of each failure-free FA (HA) is slightly larger since only two failure-free FAs (HAs) handle all the workloads of the whole system, but it is still less than 0.12. When $\frac{\lambda_{Agent}}{\mu_{Agent}} = 100$, if the increasing blocking probability for the approaches of [4] and [5] is required to be less than that of the proposed approach, the two approaches must use the load-sharing mode and equip two or more redundancies in each network segment. When $\frac{\lambda_{Agent}}{\mu_{Agent}} = 200$, even if the number of the redundancies in a network segment is set to 3, the increasing blocking probability for the approaches in [4] and [5] is still larger than that of the proposed approach.

In the proposed approach, there are two additional fault-tolerant overhead for the fault tolerance of the FA and HA, respectively. These two additional overheads are mainly dependent on the average number N_{FA_MN} of the mobility binding updates and the average number N_{HA_MN} of mobility binding restoration (see Section 5.2). Fig. 8 plots $N_{FA\_MN}(N_{HA\_MN})$ under one faulty FA (HA). Initially, $N_{FA\_MN}(N_{HA\_MN})$ is approximately equal to the value of $\frac{\lambda_{FA}}{\mu_{FA}}\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)$. As $\frac{\lambda_{FA}}{\mu_{FA}}\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)$ increases up to a threshold value, $N_{FA\_MN}(N_{HA\_MN})$ increases slowly as $\frac{\lambda_{FA}}{\mu_{FA}}\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)$ increases again. Since the total number of resource units in an FA (HA) is 50, if $\frac{\lambda_{FA}}{\mu_{FA}}\left(\frac{\lambda_{HA}}{\mu_{HA}}\right)$ is already larger than 50, the new data request (response packets) to an FA (HA) is very possible to be blocked. Therefore, at any time, the average number of in-processing data requests (response packets) in an FA (HA) cannot be greater than 50. We have also made the assumption that each MN is not allowed to simultaneously issue more than one data session (see Section 5.2). This also means that the maximum value of the number of FA_failure-affected MNs (HA_failure-affected MNs) is 50 if only one FA (HA) fails. Correspondingly, the maximum number of $N_{FA\_MN}(N_{HA\_MN})$ is 50. We can make the conclusion that the overhead of the mobility binding update (the overhead of mobility binding restoration) is restricted by the total number of resources in an FA (HA). Furthermore, the analytical results plotted in Fig. 8 can be also used to represent the failure-recovery overhead of the proposed approach since the failure-recovery overhead is also dependent on N_{FA_MN} and N_{HA_MN} (see Table 2).

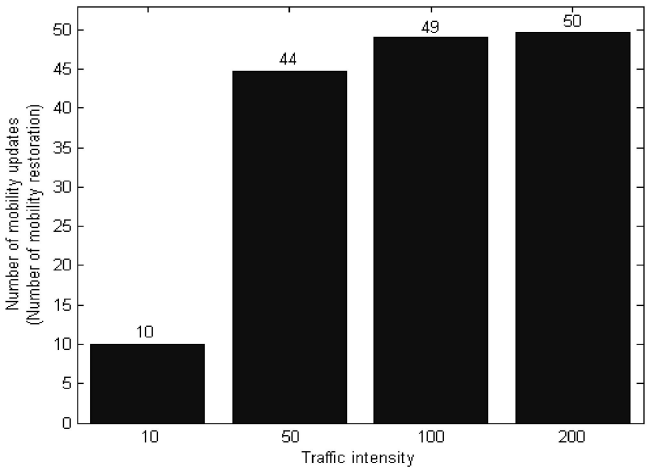

Fig. 8. Average number of mobility binding updates (mobility binding restoration).

6.2 Simulation Validation

To validate the analytical results of the increasing blocking probability for the proposed approach, simulation experiments are performed by extending the given Mobile IP simulation model in ns-2 [21]. In the extended simulation model, there are 10 FAs, 10 HAs, 1000 MNs, and 5 application servers. The size of the packet queue in each FA (HA) is set to 50 units. Each MN randomly moves between the serving areas of the FAs and HAs. The capacities of the communication links in the simulation model are randomly set in the interval 1 to 10 Mpbs. The uniform traffic is generated from five application servers to the 1,000 MNs, and vice versa, where the traffic intensity is, respectively, set to 10, 50, 100, and 200. The failures randomly occur to generate one, two, four, and eight faulty mobility agents during the simulation time. The total simulation time is set to 5,000 minutes.

The simulation results are shown in Table 4 in comparison to the analytical results. The analytical results match closely with the simulation results. In the tables, the

TABLE 4
The Validation of the Increasing Blocking Probability for the Proposed Approach

Number of faulty mobility agents	Analysis	Simulation		Difference rate	
		FA	HA	FA	HA
1	9.3854E-18	0.00014	0.00019	0.00%	0.00%
2	8.5836E-16	0.0011	0.0016	0.00%	0.00%
4	2.3503E-11	0.00766	0.00815	0.00%	0.00%
8	0.10479	0.09504954	0.09861451	9.30%	5.89%

(a)

Number of faulty mobility agents	Analysis	Simulation		Difference rate	
		FA	HA	FA	HA
1	0.062447	0.06052199	0.0621841	3.08%	0.42%
2	0.13751	0.13701561	0.13463776	0.36%	2.09%
4	0.31111	0.34051933	0.28894805	9.45%	7.12%
8	0.6962	0.68987151	0.74187728	0.91%	6.56%

(b)

Number of faulty mobility agents	Analysis	Simulation		Difference rate	
		FA	HA	FA	HA
1	0.047665	0.04759795	0.04589794	0.14%	3.71%
2	0.09581	0.09364215	0.09929257	2.26%	3.63%
4	0.19321	0.17856595	0.17613171	7.58%	8.84%
8	0.39092	0.37069882	0.37780622	5.17%	3.35%

(c)

Number of faulty mobility agents	Analysis	Simulation		Difference rate	
		FA	HA	FA	HA
1	0.024649	0.02463756	0.02270807	0.05%	7.87%
2	0.04935	0.04445464	0.04775267	9.92%	3.24%
4	0.098887	0.10111555	0.09387188	2.25%	5.07%
8	0.19841	0.18492526	0.21738434	6.80%	9.56%

(d)

(a) $\frac{\lambda_{Agent}}{\mu_{Agent}} = 10$. *(b)* $\frac{\lambda_{Agent}}{\mu_{Agent}} = 500$. *(c)* $\frac{\lambda_{Agent}}{\mu_{Agent}} = 100$. *(d)* $\frac{\lambda_{Agent}}{\mu_{Agent}} = 200$.

difference rates between the simulation results and the analytical results $\left(\frac{|\text{Analysis}-\text{Simulation}|}{\text{Analysis}}\right)$ are all below 10 percent. Note that the same workloads to an FA and HA, $P_{FA\_Blocking}$ and $P_{HA\_Blocking}$, have the same value in the numerical analysis based on (3) and (4), but the two probabilities have different values in the simulation experiments.

7 Conclusion

This paper has presented an efficient approach to tolerating the failures of mobility agents in a wireless system. The proposed approach utilizes the available resources in other failure-free mobility agents to dynamically generate a backup set for each faulty mobility agent. Compared to the previous approaches, the proposed approach possesses the following advantages:

- Not requiring hardware support.
- Not incurring failure-free overhead.
- Distributing the fault-tolerant overhead to avoid the significant performance degradation on a single failure-free mobility agent.

However, the proposed approach issues a more complicated fault-tolerant procedure to redirect the workloads of faulty mobility agents to multiple failure-free mobility agents. With the fault-tolerant overhead, the *M/G/c/c* queuing model has been used to make the analytical comparison of the

proposed approach and the approaches of [4] amd [5]. The simulation experiments have been also performed to validate the analytical comparison.

The comparison results show that when $\frac{\lambda_{Agent}}{\mu_{Agent}}$ is enlarged to 200, the increasing blocking probability for the proposed approach is obviously less than that in the approaches of [4] and [5]. This comparison result is also true in most cases of small faulty mobility agents. The approaches of [4] and [5] give the smaller increasing blocking probability when $\frac{\lambda_{Agent}}{\mu_{Agent}}$ is not too large (e.g., $\frac{\lambda_{Agent}}{\mu_{Agent}} = 10$ or 50), but they need to equip several redundancies in each network segment.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers. Their comments have significantly improved the quality of this paper. This research was supported by the National Science Council, Taiwan, R.O.C., under Grant NSC 90-2218-E-030-004 and the Society of the Divine Word under Grant SVD9021.

REFERENCES

[1] C. Perkins, "IP Mobility Support," Technical Report IETF RFC 3220, Jan. 2002.
[2] Goahead Software Inc., "Building Highly Reliable Systems: The Role of Embedded Middleware," technical report, available at http://www.goahead.com/pdf/BuildingHighAvailSys.pdf, 2003.
[3] B.W. Johnson, *Design and Analysis of Fault Tolerant Digital Systems.* Addison-Wesley, 1989.
[4] R. Ghosh and G. Varghese, "Fault-Tolerant Mobile IP," Technical Report WUCS-98-11, Washington Univ., Apr. 1998.
[5] J.H Ahn and C.S. Hwang, "Efficient Fault-Tolerant Protocol for Mobility Agents in Mobile IP," *Proc. 15th Int'l Parallel and Distributed Processing Symp.,* pp. 1273 -1280, 2001.
[6] B. Sarikaya, "Packet Mode in Wireless Networks: Overview of Transition to Third Generation," *IEEE Comm. Magazine,* vol. 38, no. 9, pp. 164-172, Sept. 2000.
[7] P.J. McCann and T. Hiller, "An Internet Infrastructure for Cellular CDMA Networks Uusing Mobile IP," *IEEE Personal Comm.,* vol. 7, no. 4, pp. 26-32, Aug. 2000.
[8] ETSI GSM 08.16: Digital Cellular Telecommunications System (Phase 2+); General Packet Radio Service (GPRS); Base Station System (BSS)—Serving GPRS Support Node (SGSN) Interface; Network Service, 1998.
[9] 3GPP TS 25.410: UTRAN Iu Interface: General Aspects and Principles, 1999.
[10] 3GPP TR 23.922: Architecture for an All IP Network, 1999.
[11] R. Mistry, P. Savill, and A. Tofanelli, "OA&M for Full Services Access Networks," *IEEE Comm. Magazine,* pp. 70-77, Mar. 1997.
[12] R. Caceres and L.R. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments," *IEEE J. Selected Areas in Comm.,* vol. 13, no. 5, pp. 850-857, June 1995.
[13] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, and R.H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," *IEEE/ACM Trans. Networking,* vol. 5, no. 6, pp. 756-769, Dec. 1997.
[14] C. Graff, M. Bereschinsky, M. Patel, and L.F. Chang, "Application of Mobile IP to Tactical Mobile Internetworking," *Proc. Military Comm. Conf.,* vol. 2, pp. 409-414, 1998.
[15] Y. Mun, Y. Kim, Y.J. Kim, and G. Hwang, "IP Mobility Support over Wireless ATM," *Proc. IEEE Int'l Conf. Comm.,* pp. 319-323, 1999.
[16] S. Khurana, A. Kahol, S.K.S. Gupta, and P.K. Srimani, "An Efficient Cache Maintenance Scheme for Mobile Environment," *Proc. 20th Int'l Conf. Distributed Computing Systems,* pp. 530 -537, 2000.
[17] H. Ahn and C.S. Hwang, "Low-Cost Fault-Tolerance for Mobile Nodes in Mobile IP Based Systems," *Proc. 15th Int'l Parallel and Distributed Processing Symp.,* pp. 508-513, 2001.
[18] D.C. Plummer, "Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware," Technical Report IETF RFC 826, Nov. 1982.
[19] D. Gross and C.M. Harris, *Fundamentals of Queueing Theory.* John Wiley & Sons, Inc., 1985.
[20] P. Lin and Y.B. Lin, "Channel Allocation for GPRS," *IEEE Trans. Vehicular Technology,* vol. 50, no. 2, pp. 375-387, Mar. 2001.
[21] Ns-2, http://www.isi.edu/nsnam/ns/tutorial/nsscript6.html, Creating Wired-cum-Wireless and Mobile IP Simulations in ns, 2003.

Jenn-Wei Lin received the MS degree in computer and information science from National Chiao Tung University, Hsinchu, Taiwan, in 1993, and the PhD degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1999. He is currently an assistant professor in the Department of Computer Science and Information Engineering at Fu Jen Catholic University, Taiwan. He was a researcher at Chunghwa Telecom Co., Ltd., Taoyuan, Taiwan, from 1993 to 2001. His current research interests are fault-tolerant computing, mobile computing and networks, distributed systems, and broadband networks.

Joseph Arul received the BSc degree in mathematices in 1981 from Indore Univeristy, India, and the MS degree in computer science in 1994 from De Paul University, Chicago. From 1994-1995, he was a lecturer at Fu Jen Catholic University, Taiwan. He completed his PhD degree in computer engineering at the University of Alabama in Huntsville (UAH) in 2001. He was also a research associate in computer engineering at UAH. He is currently an assistant professor in the Department of Computer Science and Information Engineering, Fu Jen Catholic University, Taiwan. His current research interests are computer architecture, parallel and distributed computing, multithreaded programs, and compilers. He is a member of the IEEE and the IEEE Computer Society.