Fault-tolerant Authentication and Group Key Management in Mobile Computing *

Bharat Bhargava Sarat Babu Kamisetty Sanjay Kumar Madria

Center For Education and Research in Information Assurance and Security
and
Department of Computer Science
Purdue University
West Lafayette, IN, U.S.A.
{bb, kamisesb, skm}@cs.purdue.edu

Abstract

Survivability and secure communications are essential in a mobile computing environment. In a secure network, all the hosts must be authenticated before communicating, and failure of the agents that authenticate the hosts may completely detach the hosts from the rest of the network. In this paper, we describe two techniques to eliminate such a single point of failure. Both of these approaches make use of backup servers, but they differ in the way they are organized and deployed. We evaluate our proposed architectures with a prototype system that we built. We also identify various security threats and performance issues in group (multicast) communications in mobile computing environments. We propose a scheme for efficient key distribution and management using key graphs to provide secure multicast service.

Keywords: authentication, fault-tolerance, key management, mobile

*Portions of this work were supported by sponsors of the Center For Education and Research in Information Assurance and Security (CERIAS) and NSF under CCR-9901712 grant

Contents

1 Introduction

Providing security services in the mobile computing environment is challenging because it is more vulnerable for intrusion and eavesdropping. Most of the existing wireless network models assume the presence of stationary base stations which is not quite true. For example, in the tactical mobile networks, base stations also move from one network to another network. Consider the following scenario shown in figure 1 where these traditional network models fail completely. In the figure 1, the base station BS provides routing (or packet-forwarding) services to all the hosts in the network N1. To get the service, the hosts have to authenticate themselves with BS. Therefore, each packet contains authentication information apart from the actual data. Once the authentication is successful, the packet-forwarding is done. All the hosts in N1 have a default route to BS in their routing tables i.e. all the packets originating from the hosts in N1 will go to BS no matter where they are destined. Now let us assume that the base station BS moves to a foreign network. Since the hosts in N1 are not aware of this, they still keep sending their packets to BS. Since BS is not in the home network currently and there is no other base station that could forward packets destined to BS to the foreign network where BS is present currently, all the packets that originate from any host in N1 are dropped. Essentially, now all the hosts in N1 are isolated from the Internet. This disruption of service is caused by the movement of base station which the traditional networking protocols cannot handle. Two simple approaches to handle the above problem are described below.

First approach is to set up proxy base station in the network and change the default route

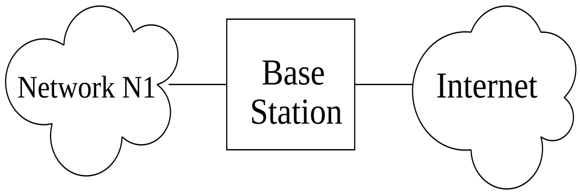

Figure 1: A network with a mobile base station

in all the hosts to point to this proxy base station. This solution is practically unacceptable for the following reasons:

- Routing tables of all the hosts should be updated manually. This becomes tedious if the number of hosts in the network are large. Moreover, manual configuration is error prone.
- Until the routing tables are updated properly, applications running on the hosts will not function properly. The currently running applications need to be restarted. Since the tables are updated manually, it is a time consuming process and hence provision of service is disrupted. Ideally, applications should be unaware of the base station's movement.

The other approach is having another base station that forwards all the packets that originated within the LAN to BS when BS is visiting a foreign network. However, the communication delays that are introduced by this solution are totally unacceptable. Consider a packet originated from a host in N1 that is destined to a host in network N2. As shown in figure 2, when BS is in the home network, the route R1 is taken to reach the destination. If BS is visiting a foreign network then the first the packet is forwarded by the new base station BS1 to BS via route R1 and from there, the packet is sent to the destination via route R2. In most of the cases, the path taken in the second scenario is longer than the one in the first case. Apart from the communication delays, this approach poses a potential security threat since all the packets that are destined to BS contain authentication information which is going outside the home network making attacker's life easier.

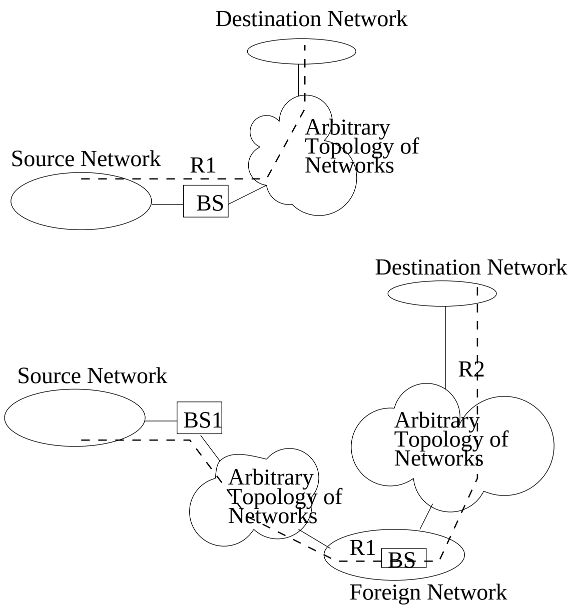

Figure 2: Solution using another base station BS1

The basic problem here is that there is a single point of failure (in our example, it is BS). To overcome this, we need an architecture that eliminates this single point of failure and provides smooth (i.e. without any disruptions) service to the hosts while still allowing the mobile hosts to move in and out of the home network in an ad-hoc manner. In the above example, we discussed about only one kind of service namely authentication. In general, it could be any form of service like secure database access.

In this paper, we propose techniques for providing uninterrupted service to the mobile hosts while still allowing the service providing agents to move or fail In an ad-hoc manner. The tecniques we provide eliminates the manual configuration problem as there is no need to update any routing tables. It does not add any communication delays as the packets will not follow different paths in any situation. It does not impose any security threats as the packets having the security information are never allowed to leave the local network. This solution is "smooth" since the applications can be totally ignorant of this problem and they need not be restarted. A description of this approach [5] will appear in the proceedings of

IC'2000 conference.

The organization of the rest of the paper is as follows. In section 2.1, we describe the primary components which make up the mobile wireless systems and discuss the Mobile IP protocol that provides connectivity in mobile environments and security requirements of the protocol. In section 3.1, we point out some challenges in providing multicast service to the mobile hosts and then identify security issues that should be handled to provide secure multicast service in mobile environment. In section 4, we discuss the issue of providing authentication service in an environment where node failures are frequent like mobile military networks. In section 5, we focus on the key management issues in group communications. In section 6, we propose two solutions to achieve fault-tolerant authentication and compare them. In section 7, we propose a solution for achieving efficient key management for secure multicast sessions. We conclude and discuss the future work in section 9.

2 Background

In this section, we briefly describe the services that the Mobile IP protocol provides and identify some drawbacks of Mobile IP from security point of view.

2.1 Mobile IP

Mobile IP[27][33] is intended to enable nodes to move from one IP subnet to another. It is suitable for mobility across homogeneous media(for example, ethernet to ethernet) as well as mobility across heterogeneous media(for example, ethernet to wireless LAN). Mobile IP introduces the following architectural entities:

- **Mobile Node or Mobile Host(MH):** A Mobile Host is a host or router that changes its point of attachment from one network or subnetwork to another.
- **Home Agent(HA):** A Home Agent is a router on a mobile node's home network which tunnels datagrams for delivery to the mobile node when it is away from home, and maintains current location information for the mobile node.
- **Foreign Agent(FA):** A Foreign Agent is a router on a mobile node's visited network which provides routing services to the mobile node while registered.
- **Correspondent Node(CN):** A peer with which a mobile node is communicating is called a Correspondent Node. A Correspondent Node could be either mobile or stationary.

Mobile IP is a protocol for transmitting IP datagrams between a Mobile Host and its Correspondent Nodes as the Mobile Host changes its point of attachment on the Internet. The Home Agent of a Mobile Host captures the IP datagrams that are sent to the Mobile Host's permanent IP address and tunnels it to the Mobile Host's Care-of-Address(COA) when the MH is away from the home network. This COA acts as the exit point of the tunnel and it could be the Foreign Agent's (FA) IP address or it could be the IP address acquired by the Mobile Host by using DHCP(Dynamic Host Configuration Protocol)[10] or PPP(Point-to-Point Protocol)[32] etc. The HA and FA together track the movement of the MH by making use of the registration messages among themselves and the Mobile Hosts. Based on the registration process, the HA keeps track of the locations of the Mobile Hosts and serves as an entry point of the tunnels.

A more sophisticated version of Mobile IP, called route-optimized Mobile IP[15] was proposed where the Care-of-Address of a Mobile Host can be disclosed to the CNs and to the FAs. As a result the CNs may tunnel their IP datagrams directly to the MH's Care-of-Address and similarly the previous FAs may also forward IP datagrams destined to the MH to the current COA. This way, the datagrams need not follow the triangular route (i.e going to HA of the MH first and then to the MH at the foreign network). Consequently, the performance of the Mobile IP is enhanced.

2.2 Mobile IP Security

In this section, we describe the basic security services provided by Mobile IP and identify the features that it lacks.

While Mobile IP promises un-interrupted IP connectivity when the Mobile Hosts roam around in the Internet, it also increases the risk of causing remote redirection of internet traffic[4] by simply introducing bogus registration and binding update messages. This is also termed as denial-of-service attack. In addition, the presence of Mobile Hosts in the foreign networks may cause security problems to both the home network and the foreign networks. The two goals of Mobile IP security protection are

- to allow a Mobile Host to enjoy similar internet connectivity and safety when it visits a foreign network as it is in its home network and
- to protect both the home and the foreign networks from passive and active attacks while the Mobile Host roams in the Internet.

In order to frustrate the remote traffic redirection attack mentioned above, registration messages must be protected with data integrity, origin authentication and anti-replay attacks. Hence these messages include 64-bit identification tag for detecting replay attacks and one or more authentication extensions [16][17] to provide message integrity and strong authentication using a Message Authentication Code(MAC). Although the use of MAC and an anti-replay tag addresses the security services cited above, the current Mobile IP lacks a scalable key management scheme for dispatching cryptographic keys needed to support these services. In order to protect the registration messages, keys must be shared atleast among the Mobile Hosts and their Home Agents.

For the purposes of network protection and resource management, it is desirable that the FAs in co-operation with the HAs can verify the identity of a MH before allowing it complete its registration and establish the attachment point on the visiting subnets.

The traffic to and from a Mobile Host while it is away from its home network generally will traverse the public Internet, as well as the visited foreign networks. Using these communication paths greatly increases the risks of the passive intrusions such as eavesdropping and active attacks such as packet alteration, insertion and deletion. Consequently, both the foregin networks and the home network of the Mobile Node may require data integrity, data origin authentication and possibly confidentiality for the redirected packets.

In order for the home network to have the same level of trust and hence provide the same amount of connectivity to a Mobile Node when it roams among the foreign networks as if it is residing at home, home network will require secure tunneling to and from the Mobile Host. Similarly, in order for the foreign network to pass traffic for the Mobile Host, the foreign network will require the traffic to be redirected by an authenticated and trusted entity and trusted entity in Mobile Host's home network such as the HA that manages the MH. Note that to achieve this the authentication mechanism between the Home Agent and the Mobile Hosts should be fault-tolerant, otherwise the failure of the Home Agent will prevent the Mobile Hosts from communicating with the outside networks. This problem is discussed in more detail in section 4 and two different solutions to this problem are presented in section 6. The secure tunnels can be implemented by using the IP security protocols (IPSec) in tunneling mode. The protocols will transform each original IP datagram using authentication and encryption mechanisms negotiated by the communicating parties beforehand and then encapsulate the datagram in an IPSec header and an external IP header that specifies the end points of the IPSec tunnel. IPSec makes use of the Internet Security Association and Key Management Protocol (ISAKMP)[20] for automated key management.

The organization of the rest of the paper is as follows. In section 3.1, we point out some challenges in providing multicast service to the mobile hosts and then identify security issues that should be handled to provide secure multicast service in mobile environments. In section 4, we discuss the issue of providing authentication service in an environment where node failures are frequent like mobile military networks. In section 5, we focus on the key management issues in group communications. In section 6, we propose two solutions to achieve fault-tolerant authentication and compare them. In section 7, we propose a solution for achieving efficient key management for secure multicast sessions.

3 Multicast issues in Mobile Environments

3.1 Challenges in Providing Multicast Service to Mobile Hosts

Multicast operation on the Internet is now supported for the fixed hosts through IP multicast[8]. Examples of multicast applications currently used on the Internet include resource discovery, as well as desktop audio/video conferencing. The provision of multicast services to mobile hosts proves to be a very challenging problem for several reasons. First, even the unicast routing for mobile hosts is a difficult problem, since the routing of datagrams intended for a mobile host changes whenever the mobile host changes the location. Second, all the existing multicast routing proposals like DVMRP[28], MOSPF[24], CBT[3] and PIM[12] implicitly assume stationary hosts when configuring the multicast delivery tree. The delivery trees established for static multicast cannot be changed easily or efficiently in all cases due to the propagation of these trees to many routers, and potentially large cost associated with making changes to the tree's structure. In addition, the movement of a mobile node after the tree is constructed can create problems. Finally, the mobile computing environment itself adds additional complexity to the problem. For example, in most wireless implementations of mobile computing, network bandwidth is scarce, error rates are higher, movement can be ad-hoc and the changing point of the network attachment for a mobile user may mean that a multicast router is not always directly accessible. In mobile environment, the network not only must manage multicast group membership and establish necessary routes, but must also contend with the fact that the established routes are transient in nature. The fact that the network must deal with the dynamic group membership and with the dynamic locations of mobile hosts makes multicast in a mobile environment a challenging problem. For a mobile host that wishes to receive multicast datagrams, the routing problem is slightly different from the unicast mobile ip routing problem. This is because the multicast datagrams are sent to group addresses and not to any network address. In addition, the foreign network that the mobile host visits may not have a multicast router and a mobile host may experience unacceptable packet losses when resubscribing. There are currently two solutions proposed to tackle this problem. But both of them do not solve the problem completely.

- **Remote Subscription:** Subscription on the foreign network is the simplest solutions for obtaining the multicast service as it has no special encapsulation requirements and operates using only existing protocols. The problem with this approach is that packets will be lost owing to the set-up time associated with multicast subscription. In addition, this approach may come at a high price for networks involved and the multicast routers that must manage the multicast tree.

- **Bi-directional Tunnelled Multicast:** This method is designed to solve the problem of topologically incorrect source addresses in datagrams by requiring traffic from the mobile host to be routed back to the home network through a foreign agent to home agent tunnel. With this approach the mobile routing agent on the home network must also be a multicast router. When the mobile host is away from home, a bi-directional tunnel to the home agent is set up. A major disadvantage with this scheme is that if multiple mobile hosts on the same foreign network belong to the same multicast group then duplicate copies of the multicast packets will arrive at that foreign network. This problem negates some of the advantages of using multicast in the first phase.

  Any proposed scheme should meet the following goals for providing multicast service for mobile hosts.

  - **Scalability:** The approach should work well even when the number of mobile nodes in the multicast group is large.
  - **Robustness:** The disruption of multicast service due to the movement of a host from one network to another must be negligible.
  - **Routing Algorithm Independence:** Ideally, the approach should be independent of the how the underlying multicast service is provided in the Internet.

It should also be independent of the unicast routing algorithm.

– **Interoperability:** The scheme should interoperate with existing Internet protocols and mechanisms, with as few changes as possible.

In addition, there are a number of infrastructural characteristics in mobile environments that makes security provision more difficult. We identify some characteristics below.

– **Autonomy:** In Local Area Networks(LAN), we usually assume that the communicating end-points and the intermediate nodes are all a part of the same organization. This assumption simplifies the security provision for a LAN. In Wide Area Networks(WAN), this assumption does not hold good since typically the communicating parties belong to different organizations which are governed autonomously. The security policies of these organizations vary widely from one another.

– **Network Partitions:** Mobility has another implication on security. Mobile Hosts move between the cells, and need to be authenticated upon entering each new cell. Currently, each authentication requires communication with Home Agent, which could be across the globe. In case of network partitions, this could be a considerable problem.

– **Clock Synchronization:** Most of the currently available security protocols assume some form of clock synchronization. This is not a valid assumption and providing clock synchronization is challenging in itself. Especially, in mobile environment where a mobile host can move across different time zones without changing its clock.

Providing confidentiality to a multicast session is another challenging issue. The scope of the multicast group can be limited by restricting the routing of its IP datagrams. By manipulating the time-to-live field in each IP datagram, hosts can limit the scope of their traffic by controlling the number of hops a datagram travels before the routers discard it. By restricting the time-to-live field of a datagram, we create basic form of confidentiality for the group by limiting the potential audience of the data. This may be considered a very weak form of confidentiality.

3.2 Security Issues in Multicast

Providing multicast security is much more difficult problem than providing just the multicast service for a mobile environment. In the previous section, we discussed how difficult it is to provide uninterrupted multicast service to a mobile host. In this Section, we discuss some issues in providing a secure multicast channel.

Potential security threats to multicast communications are similar to those encountered in unicast communications. Threats include unauthorized creation, alteration, destruction and illegitimate use of data. In the case of multicast traffic, because of the inherent broad scope of a multicast session, the potential for attacks is greater than for unicast traffic. It is desirable to secure these vulnerabilities while maintaining some of the efficiency and and performance benefits of multicast service. Multicast sessions could be classified as private or public according to the level of session access control required to receive or transmit data within the multicast group. Eavesdropping could quickly become a problem because of the potentially broadscope of a session. Session confidentiality can be provided through encryption. A secure multicast session could be defined as a private session with encryption of data content.

In order to counter the common threats to multicast communications, we can apply several of the fundamental security services, including authentication, integrity and confidentiality. A secure multicast session may use all or combination of these services to achieve the desired level of security. The amount or type of service required is dictated by the specific security policy defined for the session.

We now discuss security issues in various multicast networking functional components.

- **Session Advertisement:** Session advertisement is an important part of the overall design consideration for supporting secure multicast sessions. A generic advertisement mechanism can communicate security requirements and parameters for a secure session to its potential group members. Ideally, it is better to adapt methods already established for both secure and non-secure multicast sessions like Session Description Protocol(SDP), Session Advertisement Protocol(SAP) and Session Initiation Protocol(SIP).

  The session advertisements need to be protected as they could be modified or blocked on the way to the potential group members. It might not be possible to protect them with full guarantees, but atleast an effort should be made to protect the security parameters for that group. The problem is more significant in case of mobile hosts as they can move rapidly from one network to another network and hence the session advertisements also have to be traversed through many tunnels before reaching the mobile host. This greatly increases the risk of exposure of the information in the session advertisement packets.

- **Admission Control:** Admission Controlling component decides whether to accept a join request of a participant or not. This component of the system should be protected and anybody masquerading this should be detected immediately. Otherwise a masquerader can admit a participant (even though the actual security policy of the group does not allow this) who starts sending data which could be easily decrypted by this masquerader. There is a need for mutual authentication whenever data is sent across a network to have maximum security.

- **Multicast Routing Protocols:** In order to deliver the multicast IP datagrams to group members, routers may use one of the several routing protocols that define the network routing topology. In general, it is desirable to design multicast security mechanism that is independent of any particular routing approach. As discussed in section 3.1, the multicast routing information is difficult to keep track especially in mobile environment. Using these multicast routing protocols to distribute multicast information and group keys is a difficult problem in itself.

- **Multicast Reliability Mechanisms:** There are many multicast applications that require a more reliable transport delivery mechanism[2] than available through the generic and unreliable combination of UDP/IP. Key distribution is one area that benefits greatly from the introduction of efficient and reliable multicast transport methods. The overall coherence of a secure multicast session depends upon the successful distribution of keys to the secure multicast group. Existing unicast design solutions do not scale well to the multicast case and often present considerable efficiency concerns, exploding state maintenance and processing burdens. There is no single reliable transport protocol like TCP that can service all classes of multicast applications. In addition, the reliability mechanisms used for real-time and non-real-time applications[31] may differ because of timing constraints.

  The security policy will dictate what reliable multicast transport mechanisms should be used to ensure that key material is delivered to all participants. In particular, the policy will dictate whether the key distribution mechanisms should be sender or receiver reliable. Receiver reliable mechanisms place the burden of receiving the required key material on the receiver. Sender reliable mechanisms place this burden with the distributor of the key material.

  A complete solution to multicast security addresses all security issues discussed above. As observed in [18], we may reduce many of our security concerns to a key management problem and distribution problem. The most widely used criteria to evaluate various key management architectures is listed below.

  1. **Scalability:** While designing a key management scheme, one should address the scalability problem. Scalability of a solution to a secure multicast session is an important issue, as non-scalable solutions are only of theoretical interest. A solution like manual key distribution is not scalable for more than a couple of people. On the other hand, highly scalable solutions demanding lots of processing power and other resources both on the initiator's side and participant's side are not practical at all.
  2. **Efficient Rekeying Operation:** As the multicast session proceeds, it may be necessary to issue new keys to the some participants or to the whole group (i.e. group key). For example, after a new participant joins an existing group or after a participant leaves a group, new group key needs to be issued. This is called re-keying. Re-keying should also be performed when the group key is compromised. Based, upon the size of the group and the architecture being used this re-keying may be an expensive operation. In some architectures where each participant can have multiple keys, if one key is compromised all the other keys that the participant holds might have to be re-keyed. The efficiency of re-keying

is an important issue as this should be done without disrupting other on-going communications. Ideally, a compromised participant should be deleted or given new key(s) without other participants noticing it. This results in a smooth session which is highly desirable.

3. **Storage and Computational requirements:** It is challenging to design an efficient group key management scheme that keeps the storage and processing requirements of the participants at a minimum. Depending upon the group communication architecture being deployed, each participant might need to have multiple keys and additional processing power. The storage requirements and computing power needed for the participants of the group have to be kept at a minimum especially in the mobile environment. The resource requirement of initiator of the group should be well defined as it plays an important role of setting up and maintaining the group.

4 Fault-tolerant Authentication

Authentication is the mechanism by which the receiver of a message can ascertain its origin [30]; an intruder should not be able to masquerade as someone else. Most of the authentication protocols that have been proposed in the past requires a trusted third party which generates the secrets keys for the communicating parties. There are some problems with this approach. For example, if the number of communicating parties are more, then the third party will be overloaded and becomes a bottleneck for communication. It also becomes an attractive spot for attackers. If a malicious guy breaks into the trusted party's secret database, then all the keys are compromised. Especially in mobile environment, where the bandwidth is of great demand, third party solution does not work out well. In a decentralized system where the secret keys are distributed among a group of entities, there will not be any communication bottlenecks. Moreover, this increases the survivability of the communication system.

In mobile networks, when a mobile host wants to securely communicate over the network with other hosts, it has to be first authenticated by the Home Agent. When there is a single Home Agent, this becomes a single point of failure i.e, when the Home Agent fails, all the mobile hosts in the home network cannot communicate with the outside world. A simple but powerful solution to this problem is to have back up Home Agent(s), which assumes the responsibility of a Master Home Agent when the current Master fails. Failure of a Home Agent can be detected by sending "HELLO" packets to it or listening to the agent advertisements. If the current Master is not responding or advertising since a fixed amount of time, then it can be declared to be dead and one of the backups take up the responsibility of the Master Home Agent. This solution requires that the secret key database is fully replicated on all the backup Home Agents too. This introduces a potential security threat as there are several sites that could be attacked now and other database consistency issues. In section 6.1, we propose a more refined solution using this idea that achieves controlled and smooth transitions between the Master Agent and Backup Agent. This is an extension to the idea specified in[29]. In section 6.2, we propose a solution to this authentication issue using an entirely different approach. The basic idea is to logically arrange these back up servers in a hierarchy that represents the communication flow. In case of a Home Agent failure, a node at higher level in the hierarchy can authenticate the mobile hosts of that network and provide network services uninterruptedly.

5 Multicast Key Management Issues

In this section, we identify some key management issues in group communications. Supporting dynamic groups implies that newly joined members must not be able to understand past group communications, and that leaving members must not follow future communications. Key changes are required for all group members when a leave or join occurs, which poses a scalability problem if groups are large. Any approach to provide secure group communication service should allow group members to establish a mutually shared secret, which can be used to provide group privacy, message authenticity, or any other property relying on a shared secret. Ideally, transitions from one key management approach to another in a running system are possible. Another challenge is to design a key management scheme that can offer perfect forward secrecy and still demands small computational power and storage capacity from the participants, and avoid investing trust into third party components such as routers. Existing protocols for secure multicasting are limited to distribute session keys in static and/or small groups. It is desirable that changes to the group membership are possible with minimal involvement of dedicated nodes and group members, limiting the size of messages and computing resources needed. The approach should also cope with several properties inherent to multicast environments like unreliable transport channel, out-of-order delivery of packets. For example, multicast application layer data is typically encapsulated by the transport layer UDP protocol. The combination of UDP and IP protocols create an unreliable datagram service without error correction capabilities. While third party entities such as routers are entrusted with forwarding secured data, they are not allowed to gain access to actual keying material or plain-text payload. Additionally, any third party recording ongoing transmission and later capturing the secrets held by a participant must not be able to understand its recordings. A naive solution to achieve this is to have a lifetime for the keys after which they cannot be used to decrypt any information.

Through the use of encryption and digital signatures, we can achieve desired levels of confidentiality, integrity and authentication. We may generally assume that cryptographic algorithms cannot be broken and hence all security lies in the key material. With this in mind, a secure multicast session can be defined by its Class D IP address and the required keying material for the session.

The key management architecture used and the encryption mechanism used determines the size, type(symmetric or asymmetric) and the number of keys required to secure a multicast session. In general, session participants may use a common group traffic encryption key to encrypt session data. The initiator of the group can use group key-encryption-keys to encrypt the future session keys. For private multicast sessions, access to these keys must be restricted to maintain the security of the overall session. In addition, the keying architecture should prevent collusion by a group of disbanded members from generating or recreating the new group key. The group traffic encryption keys, key encryption keys should changed periodically, otherwise if the multicast session is long, it is vulnerable to attacks. So, the keys should be issued or changed periodically. This is called "rekeying" of a session. Rekeying should also be done in the event of a key compromise, voluntary exit of a participant from an established session etc. The challenging task here is to design a key management architecture that supports rekeying operation in an efficient and scalable way. If the rekeying operation has to be done whenever a member joins or leaves a group, then the key management scheme should support rekeying in a rather natural way. The problem becomes more difficult in mobile environments, as the mobile hosts can move sporadically between the networks and still want to have seamless multicast service. Imagine the group initiator himself is mobile, then he has to hand-off all his responsibilities to another host in the current network. This implicitly assumes that all the hosts in an intranet could be trusted which is not a realistic assumption always. In addition, when the initiator is away from the home network, the host that assumes the responsibility of the initiator, must have processing and storage capacities comparable to that of the initiator. This could be achieved by configuring the mobile hosts at boot time by assigning priorities to the hosts that can take up responsibilities according to the resources they have access for.

In section 7, we propose a solution for achieving efficient key management in secure multicast sessions.

6 Proposed Schemes for Fault-Tolerant Authentication

In this section, we propose two different schemes for achieving fault-tolerant authentication. The first approach uses an abstract entity called Virtual Home Agent and the second solution requires that different Mobile Agents be arranged logically in a tree structure. Fault-Tolerant Authentication is essential in tactical mobile military networks where the base stations are subject to failure.

6.1 Virtual Home Agent Scheme

We define the following entities that we use in the scheme.

1. **Virtual Home Agent(VHA):** A VHA is an abstract or virtual agent that is identified by a network address (eg: IP address). All the hosts send their authenticating requests to the VHA's network address. VHA acts as a default Home Agent for the Mobile Hosts in a LAN. VHA's responsibilities include authenticating the Mobile Hosts by using the Shared Secrets Database.

2. **Master Home Agent(MHA):** A MHA is a Home Agent that is currently assuming the responsibilities of a VHA. For a VHA, at any given point of time, there should be only one MHA assuming that VHA's responsibilities. A MHA intercepts and processes all the packets destined to VHA's network address.

3. **Backup Home Agent(BHA):** A BHA is a Home Agent that backsup a given VHA. There could be more than one BHA for a given VHA, in which case, each BHA will be assigned a priority. In the case of failure of the MHA, BHA having the highest priority becomes the MHA.

4. **Shared Secrets Database:** Each of the Mobile Hosts share a secret with a VHA (i.e. the secret is only known to the Mobile Host and the VHA.). Typically, a secret could be a password or a symmetric key like DES key. All these secrets are stored in a separate database called Shared Secrets Database.

5. **Shared Secrets Database Server:** The server that protects and processes the queries and updates to the Shared Secrets Database is called Shared Secrets Database Server. The VHA will send requests to this server while authenticating a Mobile Host or when a new shared secret is issued to a Mobile Host. To frustrate the impersonating attacks by malicious hosts, the VHA has to authenticate itself with this server.

Figure 3 illustrates the above discussed entities in a typical scenario. In the figure, the

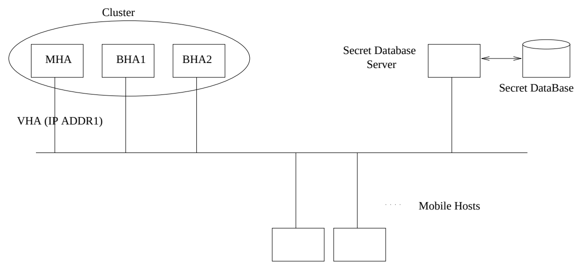

Figure 3: A LAN with a VHA, MHA and BHAs

MHA is the Master Agent for VHA identified by the IP address IP ADDR1. BHA1 and BHA2 are the Backup Agents for the VHA. Only the MHA contacts the Shared Secrets Database Server. The protocol functionality is described below.

Periodically the MHA sends advertisements on the network to a pre-determined or pre-configured IP multicast address. All the BHAs and MHA join this multicast group. The advertisements sent by the MHA have a ttl (time to live) field set to 1 to prevent the packet forwarding to outside networks. Each BHA is assigned a priority. This priority indicates the administrator's preference for a BHA to become MHA if the current MHA fails. The MHA has the lowest value for the priority than all the BHAs. Each advertisement is a packet that contains the following items.

- VHAs IP Address
- MHAs Priority
- Authentication information

The authentication information is necessary as any body could inject bogus advertisements if it were not there. This could be an encrypted password which is shared between the BHAs and the MHA. We denote this advertisement as a tuple <VHA's IPAddr, Priority, Auth-Info>. This advertisement is transmitted periodically every few seconds and this time period is called Advertisement Interval. All the BHAs listen to these advertisements. If the advertisements are not heard for some period of time, then the election of a new Master starts. Typical election protocols require the backups to exchange their configured priorities or any other information required for running the election protocol, and then elect the new Master. This approach has several drawbacks. First drawback is that it introduces more traffic into the network. Second drawback is that this election protocol does not provide smooth transition from Backup to the Master because communication over networks especially on wireless media is time consuming. This slow transition disrupts the service provision as the authenticating requests that arrive in the transition period are essentially lost. Third disadvantage is that the priority values could be manipulated by malicious nodes on the network biasing the election result. This is disastrous, since if a BHA is compromised and its priority could be configured to be higher than others, it could be made the Master and from then on all the communication is exposed.

Below we propose a simple scheme that overcomes all these disadvantages. Each BHA sets the Down Interval Timer as described below. When the Down Interval Timer expires, the BHA transitions to the Master state.

$$DownIntervalTime = 5 * AdvertisementInterval + (BHA'sPriority \div MHA'sPriority)$$

The MHA's priority is obtained from the 3-tuple described above. Each of the BHAs reset the Down Interval Time whenever an advertisement is received on the multicast channel. There are two things that need special attention regarding the Down Time Interval. First of all, it's value is atleast five times the advertisement interval, so the election process will not start until the MHA fails to send five consecutive advertisements. Note that a BHA might not receive some advertisements even though the MHA actually sends them due to packet losses, but five or more consecutive losses of the same packet is very unlikely. Secondly, the Down Time Interval is a function of BHA's configured priority. If the BHA's priority value is less, then its Down Interval Time is less. This means that the BHA having the lowest priority value will have the lowest value for the Down Interval Time. When this Timer fires, then that particular BHA becomes the Master and starts sending the advertisements from then.

The advantage of this election scheme is that there is no communication between the BHAs once the MHA is down. This eliminates security threats, time delays and does not use any extra bandwidth. Also, it is guaranteed that only the Down Interval Timer of the BHA having the highest priority fires earliest and hence there is no confusion and no additional computations are required. The downside of this algorithm is that there is a possibility of partitions. Consider the following scenario. Once a BHA's Down Time Interval fires, it sends an advertisement to the multicast address announcing its presence as the Master. But the packet reached only a subset of other BHAs which now are aware of the new Master.

The other subset of the BHAs have not received the advertisement and one of them declares itself to be a Master. Now there are two Master's leading to chaos. But this is not such a serious problem in a LAN as packet loss is very less and especially broadcast networks support multicast in a natural way - only a single copy of a packet will be transmitted in a LAN even though it is a multicast packet. So, if the packet is lost, none of the BHAs will receive.

The state transitions are shown in the figure 4 and the functions of a Mobile Agent in each state are described below.

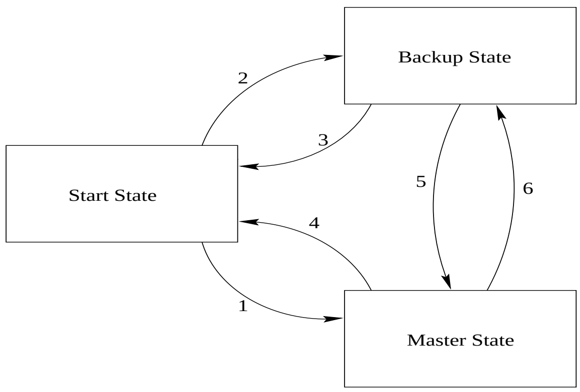

Figure 4: State Transition Diagram for the proposed scheme

6.1.1 State Machine Description

Any BHA or MHA will be in one of the states depicted in the above picture. In each state, a Mobile Agent performs certain functions. We give brief description of these functions below.

1. **Start State:** In this state a Mobile Agent performs certain initialization tasks before becoming a MHA or a BHA. If the administrator configures an Agent as a Master, it

   - sets it's Advertisement Interval to a configured value.
   - joins the multicast group where all the BHAs listen to the advertisements that the MHA sends.
   - sends an advertisement to the multicast group.
   - transitions to the Master State.

   It the Mobile Agent is configured as a Backup, it

   - joins the multicast group where all the BHAs listen to the advertisements that the MHA sends.
   - computes and sets the Down Interval Timer by the above given formula.
   - transitions to the Backup State.

2. **Master State:**

   In this state a Mobile Agent performs the following tasks:

   - Whenever the Advertisement Interval Timer expires, it sends an advertisement packet to the multicast group and resets the Advertisement Interval Timer.
   - It assumes the responsibilities of VHA i.e. it processes the authenticating requests sent to the VHA's network address.
   - If it receives an advertisement with a priority less than it's priority it transitions to Backup State. Note that this is useful in administratively bringing down the current Master.

3. **Backup State:** In this state, a Mobile Agent performs the following tasks:

   - It never sends any advertisement packet unless the Down Interval Timer expires or it is made a MHA administratively.
   - It never responds or processes the packets sent to the VHAs network address.
   - If an advertisement is received, it resets the Down Interval Timer.
   - If the Down Interval Timer expires, it transitions to Master State and sends an advertisement to the multicast group address so that other BHAs know about the new MHA.

The state transitions 1 and 2 depicted in the figure 4 occur once the initialization phase in the Start State is over. The state transitions 3 and 4 occur when the Mobile Agents are brought down either for administrative purposes or because of some exceptional conditions like server failures. The state transition 5 occurs when the Down Interval Timer of a BHA expires indicating the failure of the current MHA. It could also occur if a system administrator wants to make a BHA to be MHA if the current MHA's processing capacity is not satisfactory. The state transition 6 could occur when the current Master is made a Backup Agent and another MHA is installed or a BHA is made a MHA.

The proposed scheme has only 3 states and is easy to implement. In each state there are very few tasks that need to be performed and hence the protocol overhead is negligible. Even though the scheme is simple it is powerful and flexible enough. This scheme allows the existence of multiple VHAs on the same LAN. This feature is highly desirable when the node failures are frequent. An Agent can be a MHA for a VHA and at the same time could be a BHA for another VHA. The advertisements should be encrypted and the sender of the advertisement should authenticate itself with the BHAs to prevent bogus advertisements on the network. For this purpose, the advertisements contain Authentication data. The database requests sent by a MHA to the Shared Secrets Database Server should also be encrypted and contain authentication information so that secret database is not leaked out by bogus requests. Note that when we say authentication, we mean mutual authentication i.e. sender has to authenticate itself to the receiver and vice versa. The database server has been described here as a separate entity for the sake of explanation. It is possible to integrate the database server functionality into the MHA itself. This means that the secrets database needs to be fully replicated on all the BHAs also, as a BHA could become a MHA at any time. Database consistency should be maintained using techniques similar to the ones used in a fully-replicated distributed Database Management System.

6.1.2 Enhancements

Even though the proposed solution in the previous section is sufficient for most of the common scenarios, optimizations are possible by utilizing the Backup Home Agents appropriately. In this section, we describe some modifications to the scheme described earlier.

Encryption and decryption are expensive operations and doing it for every packet on the network introduces unacceptable delays in the communication. In the scheme described in the previous section, if the network is busy MHA becomes a bottleneck. Since every packet has to be decrypted, if the network is busy lot of packets will be queued up at MHA and once the queue becomes full, it starts dropping the packets worsening the congestion. If the LAN is very large, then this might happen frequently. So, the solution is not scalable with just one MHA. Moreover, the BHAs are just listening to the MHA's advertisements. They do not service any of the requests and hence their processing capacity is not utilized properly. Even though, the default BHA has been replaced by a virtual entity to eliminate single point of failure, the central database, might still be an attractive target to the attackers. We can replicate the database fully on all the BHAs and the MHA, eliminating the central database server, but this requires additional storage capacity on each of the BHAs and the MHA. Also, any updates to the database have to be carried out on all the BHAs and MHA which is again an expensive operation.

The scheme can be extended by forming a cluster consisting of MHA and the BHAs. A cluster is a group of servers acting as a single server. This gives the effect of a multiprocessor machine. A cluster is identified by a single IP address. A cluster consists of a front end machine and one or more backends. Only the identity of the front end machine is known to the outside world. When a client sends a request to the front end, the front end forwards that request to one of the backends based on various factors discussed below. The backend, the services the request. Front end does not process any of the request, but just routes the request to an appropriate backend. So, the front end will not be a bottle neck. Typically, there will be more than one backend and hence the throughput of the system increases dramatically. In our modified scheme, MHA is the front end and BHAs become the back ends. So, BHAs are now used to process the requests instead of just listening to the MHAs advertisements. This increases the scalability and efficiency of the system, especially when the backends are dedicated solely for servicing the authentication requests. The front end has to do load balancing to avoid overloading particular backend by forwarding too many requests to it. It also has to keep track of which backends are active at any given instant. Note that the back ends in this scheme remain anonymous like in the previous scheme. A variation of this scheme could be achieved by doing request redirection instead request forwarding. In request redirection, when a client contacts a front end, the front end chooses a backend and asks the client to contact that backend to get service. So, the client now knows the backend's identity too. This is not desirable for security purposes and moreover it places an additional burden on the client, introduces additional communication delay and the number of messages exchanged are more, thus congesting the network.

The request forward decision by the front end depends on different factors. We mention some of them here. It should be noted that a more complete set of dependent factors should be tailored with the application under consideration.

- **Content Based Request Forwarding:** The front end looks into the request and decodes what kind of service the client wants. Based upon the content of the request, the front end decides which backend is best to service that request. For example, if packets from particular host can be authenticated by only by a particular back end that has the appropriate key, then front end uses this technique.

- **Load Based Request Forwarding:** The front end forwards the request to a minimally loaded back end at any given point of time. The back end have to report their load to the front end periodically so that it can keep track of the most recent load on every back end. This technique ensures fairness among the back ends. Note that load reporting implicitly signals the front end that a particular backend if alive.

- **Locality aware Request Forwarding:** The front here keeps track of the cache contents of the back ends. When a request arrives from a client for some data, the front end determines which back end has the item requested in its cache and forwards the requests to that back end.

Using only one particular technique might not achieve maximum efficiency. For example, if a cluster has ten backends out of which one of them has a document that is requested by many clients and if the front end does locality aware request forwarding, only that back end is busy and all others will be idle. In this case, load based and locality aware request forwarding have to be done. In case of load based request forwarding, the challenge is to identify various parameters that represent the load on the back ends. These parameters are reported to the front end using load reports. Below we identify some of the load parameters that need to be reported periodically. The reports are sent to the same multicast address that is used to listen to the MHAs advertisements. This enables building third party load monitors that just listen to these reports, analyze them and log the statistics.

- Queue length
- Requests received
- Requests dropped
- Requests received during the last interval
- Requests dropped during the last interval
- Locality cost

The secrets database now can be partitioned among the nodes in the cluster and so the cluster can be viewed as a distributed database system. So, even if one of the nodes fails, the other can still service the requests. If the database updates are advertised through the multicast address, all the nodes in the cluster can log the changes and when a particular node where some data items are maintained fails, others can quickly process their logs and build the missing data items locally. This enhances the survivability of the system greatly. This also helps in smooth transition from a Backup State to the Master State because the BHAs have the updated database (possibly constructed from their log) and their caches are already built up.

6.2 Hierarchical Authentication Scheme

In this section we propose another solution for achieving fault tolerant authentication using a tree based organization of the Home Agents in the LAN. For the purpose of the discussion we use figure 5. On the right side of the figure there are two Local Area Networks (LAN) and on the left side of the figure we show the logical organization of the Agents in LAN1. The dotted lines show the communication link between the two LANs. This could be wired or wireless media. In LAN1, Mobile Hosts D, E, F and G are Mobile Hosts that are not routers. A, B and C are the Agents that could provide service to the Mobile Hosts. The particular service that we discuss here is packet-forwarding, i.e. these Agents can act as routers and forward the packets that originate from the Mobile Hosts. One could think of large number of services that Agents could provide, for example, access to weather reports is another popular service: the hosts have to authenticate themselves with the Agents to gain access to the weather reports. Subscribed clients will be given a key-list so that he can authenticate and get access to the services.

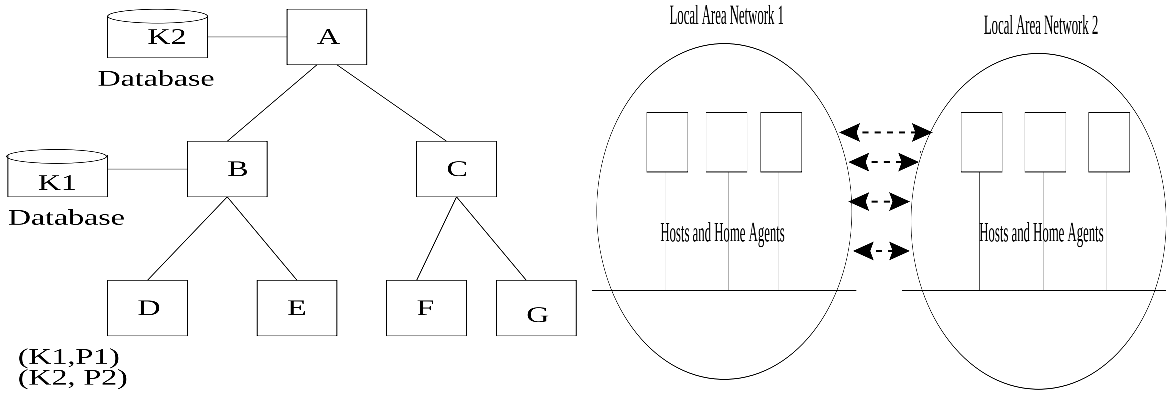

Figure 5: Example using tree-based approach

The proposed architecture is described below. In LAN1, the Agents are logically arranged in a hierarchy forming a tree like structure. The dumb hosts (or the clients that need service) are at the leaf level. Intermediate levels are occupied by Agents (or the service providers). A leaf node shares a secret with each of the Agents that lie in the path from itself to the root of the tree. In the figure, Mobile Host D shares a secret key with every host in the path from D to A i.e with B and A; with B it shares K1 and with A it shares K2. With each of the secret key, a priority is assigned. The key having the highest priority will be used for authentication purposes before the key with the next highest priority is used. Here, K1 has priority P1 and K2 has a priority P2. These priorities could be assigned based upon different factors. Some of the factors are listed below:

- **Communication Delays:** The lower the communication delay with the Agent, the higher should be the priority of the key shared with that Agent.
- **Processing Speed:** If a Agents processing speed is more, the key shared with that Agent is the preferred one.
- **Secret Key Usage:** Usage of the same secret many times over the network may leak the secret. So, as the number of times the key is used increases, it's priority should decrease.
- **Life Time of the Key:** Having an infinite life time for a key is not safe. If the key is compromised at any point of time, all the (recorded) communication that took place in the past could be decrypted and so is the future communication. So, each key should expire after certain period of time. This way, if a key is compromised after it expires, it cannot be used to decrypt ongoing and future communications although past communication for certain period of time could be decrypted. Hence as the life time of a key is going close to expiration time, its priority should decrease; otherwise it might happen that while sending a request a key is valid but after the request is received on the other side request becomes invalid. Note that the Agent also should keep track of the lifetime of the secrets in its database and delete the records for which the lifetime expires.
- **Availability of Secret Key information to an Agent:** If the secret key database is partitioned, it might happen that an authenticating agent might not find the secret key information in its database to process a request from a client. So, it needs to contact other agents that provide the service to fetch the database record corresponding to the request. This might add more delays to process a request. So, the client should update a key's priority every time it uses it.
- **Configurable Priorities:** The system administrator should be given flexibility to assign priorities to the keys. This way they can make a key unusable immediately if a key-compromise is detected.

A given key's priority should be a function of all these factors. Each factor could be assigned a weight factor W_{f_i}, where f_i represents the i^{th} factor. So, if P_{K_j} represents the priority of Key K_j, then P_{K_j} is computed as $P_{K_j} = \sum_{i=1}^{n} W_{f_i} * P_{f_i}$. In the formula, P_{f_i} is the priority of the key computed taking only factor f_i into consideration. In this example, we assign priorities according to communication delays assuming that the delay is proportional to the distance between the nodes in the tree. All other factors are ignored for the sake of explanation. So, P1 is greater than P2 as B is nearer than A to D. Now, when D wants to communicate with any host in LAN2, it has to first authenticate itself with B or A (both are service providing agents as they are not at the leaf level in the tree). Since P1 is greater than P2, key K1 and Agent B is chosen. D sends a authentication request packet to B sending the secret K1 either directly or indirectly. For this any of the well known authentication schemes can be used. B sends back either a positive or negative acknowledgement based upon whether the authentication is successful or not. Once authentication with B is successful, D can communicate with any outside host as B provides the packet-forwarding service. For more security, each packet that gets service from the Agent should be authenticated.

If the agent B fails, then D will not get any acknowledgement from B. After sending the requests for fixed number of times, it now uses the secret key K2 to authenticate itself to A. At this point, D has two choices. It either can discard the key K1 assuming that node B has gone down permanently (for example, in battle field) or D could reduce the priority of K1 to a value less than P2 and any other keys it has, assuming that B's failure is only temporary (for example in commercial networks or in mobile environment where the agent itself is mobile). Once D authenticates itself to A, it now will be able to communicate with the outside world unlike when there was a single point of failure.

New keys are established after a key's priority reaches below a threshold value (for example, when a key's life time expires). To avoid confusion as to which key is in use at any given point of time, version identification numbers could also be attached to the key material used. Further experimentation is needed to exactly quantify the key priorities as a function of the above mentioned parameters.

It is interesting to observe the storage requirements of each Agent like A, B or C. B needs to store only two keys (one for D and one for E), but A needs to store 4 keys(for nodes D, E, F and G) assuming that the intermediate nodes like B and C are dedicated nodes that provide service and no applications are run on top of them. In general, an Agent has to store the secret keys for all the leaf nodes that are under it in the hierarchy. The size of the database maintained by all the Agents at the same level in the hierarchy will be more or less same (if the tree is not a complete n-ary tree).Even though, each record's size is not big, the size of the database might grow if the there are large number of nodes in the whole hierarchy.

To solve this problem, we propose that based upon the Agent's storage capacity a threshold is set, beyond which it cannot store any more secret keys. The rest of the database is stored at the Agent that is located one level up in the tree. For example, in the figure if B cannot store all the six keys, it stores only a part of the database and forwards the rest of it to A. So, when a host wants to authenticate with a service providing agent, the agent looks up for the secret it shares with that particular host in its data base. If it is not there, it fetches the record from the base station above it and then authenticates the host. To achieve higher efficiency, it is better if the base station caches the recently fetched records. Thus, essentially it becomes a distributed database problem. This might increase the communication delays a bit, but secret data could be partitioned cleverly using probabilistic approaches and observing communication patterns so that the secret key records wanted most of the times are readily available. The problem becomes more challenging when the authenticating agent itself is mobile. In such a case, the current authenticating agent should hand-over its responsibilities to any existing backup authenticating agents (and in the process, the whole secret key database might have to be migrated onto the new authenticating agent). It is desirable to do this transition as smoothly and transparently as possible. An approach similar to the one proposed in the previous section could be used to achieve this. Note that in this scheme, an agent that lies above another agent acts as a backup agent implicitly. For example, if B wants to visit a foreign network, then it has two choices. First being, it activates its explicit backup agent by migrating its database onto the backup and hand-over all it's responsibilities. Second and simple choice is to do nothing. In this case, the clients detect that B is down after sensing that B is not acknowledging and then switch over to A by using a different key. We recommend that before visiting foreign networks, the service providing agents inform the clients so that clients need not wait or keep resending the requests until they time out. A special advertisement packet could be used for this purpose.

Virtual Home Agent Scheme	**Hierarchical Authentication Scheme**
Arrangements of Agents is flat.	Agents are logically arranged in a tree structure.
Each host has only one key.	The number of keys thay a host holds depends on the height of the tree which in turn depends on the arity of the internal nodes.
Failure or movement of the base station is completely transparent to the hosts.	Hosts should be aware of the failure of the base stations as which key to use at any given point of time depends which base station is serving it. Failure of the base stations is detected using timeout mechanism.
As each host uses only one secret key, there is no priority assigned with the key.	Each key has a priority associated with it and the priority depends on various factors and changes from time to time. At any point of time, secret key with the highest priority is used for authentication purposes.

Table 1: Comparision between the two proposed approaches

The two proposed schemes solve the problem of providing fault-tolerant authentication in an efficient manner. The architectural and functional differences between the two approaches are summarized in Table 1.

6.3 Evaluation

In this section, we evaluate the approach we proposed in 6.1 along with the enhancements described in 6.1.2. We conducted simulation studies using a prototype system we built.

The studies are carried on a Sun SPARC 5 workstation operating at 170MHz. In all the experiments we carried, the secret database is fully replicated on all the backup agents too and we have not considered the dynamic updates to these database records. This amounts to having infinite life time for each of the encryption keys.

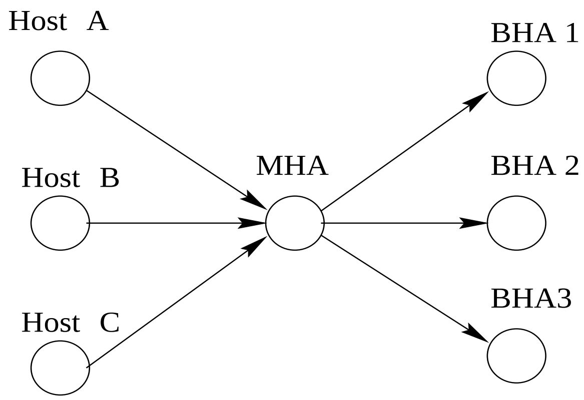

Figure 6: Network Topology in the basic scenario

In the basic scenario, we have set up one Master(MHA) and three Backup(BHA)s. A VHA is identified by an id number instead of an IP address. Similarly, each node in the network is also assigned an unique id. In real world scenarios, either a MAC layer address or an IP address serves the purpose of uniquely identifying a node. The BHAs use this Id number, to decide which key from the database is to be used to properly decrypt the message. MHA does request scheduling in a round robin fashion - the first request goes to the first backup, second one goes to the second backup and so on. The backups periodically send load reports to the current MHA informing about their load. In our implementation, the report contents are - current queue length, cumulative number of requests received and dropped. The current MHA's advertisements contain encrypted password so that the BHAs can be sure of their origin. In our experiments, password is a 64-byte string configured at the startup time and is encrypted using DES [30]. A one way hash of the tuple <VHA Id, MHA priority, Encrypted password> is also appended to ensure integrity of the message.

The network topology is shown in figure 6. The presence of the BHAs if known only to the MHA and all the mobile hosts send the packets to the MHA only. Since, we implemented the proposed scheme at the application layer, all the hosts send their packets to a well known port where the MHA is listening to. To process a request, the following steps are carried out at the MHA - MHA first decides where to route this request and sends to one of the BHAs, then based on the source of the packet, BHA gets the key from its cache and finally the encrypted packet is decrypted using the extracted key. If the decryption operation fails the packet is simply ignored.

Our observation is that the round robin request distribution minimizes the the idle time of the cluster (the cumulative amount of time that the nodes in the cluster are idle). This might not be true in real world scenarios where the database is not fully replicated, there will be cases where required key(s) is not present in the local database. The back up node have few choices in such cases:

- packet is dropped hoping that it will be retransmitted and the MHA will distribute the next request to another backup next time
- the request has to be re-routed to another backup having the required key material.
- the required key material has to be fetched from another backup server

In all the above cases, additional delays are introduced. Note that based upon the underlying transport protocol, each of the above mentioned choices will have different consequences. If the transport protocol provides unreliable transport service like UDP, first mentioned choice is not advisable since dropped packets are never retransmitted. If the transport protocol is like TCP, the tcp state has to be handed-off [23] to other backup server to implement the second choice. In case of the third choice, the backups not only process the requests distributed by the MHA, but also by other backup servers.

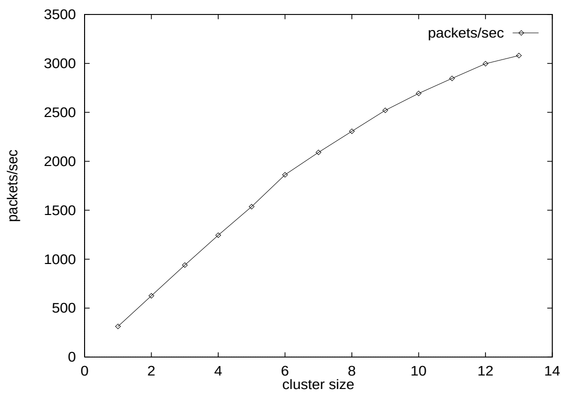

Figure 7: Cluster Throughput

Another observation we made is that the throughput (the total number of requests processed per second by the cluster) increases with the number of nodes in the cluster assuming that the front end is never a bottle neck. This is a fairly reasonable assumption as most of the LANs have far less than a hundred nodes and large cluster sizes are not needed. This assumption might not hold good for a server on the Internet which can potentially receive millions of requests a day. We conducted a simulation to measure the cluster throughput for various cluster sizes using round robin scheduling algorithm. The results are shown in 7. It can be observed from the graph that the throughput is proportional to the cluster size until the size of the cluster is 8, but as the cluster size is increasing, the increase in the throughput is gradually decreasing. This is because the front end becomes a bottleneck as the request frequency increases and cannot route the requests fast enough. Gradually, the front end reaches a saturation point and starts dropping the packets.

Another factor that we did not account for is the time taken for the disk access to fetch the required key material. In our implementation, secret database if fully replicated and resides in the cache of the servers and hence the servers never need to access the hard disk. As observed in [23], server's throughput is limited by the disk access too.

7 Proposed Scheme for Multicast Key Management

In this section, we propose a solution for achieving efficient key management in multicast communications. As pointed out in section 5, the issues that are to be addressed while proposing any approach for secure group communications are scalability, reduction of the amount of trust invested in any third party entity and reduction of computational complexity, storage requirements on the participant's side.

We first describe the entities and their functions in our scheme below:

- **Sender:** Sender is the process or the application that sends the data to the multicast group. For secure communication, sender always encrypts the data with a traffic encryption key.
- **Recipient:** Recipient is the entity that receives the data from the multicast group and decrypts it using a traffic encryption key.
- **Group Manager/Server:** Group Manager entertains the join/leave requests for a group. It also sets and enforces admission control policy before allowing any participant to take part in the multicast session. The Group Manager is also responsible for rekeying of the keys if and when it is necessary (for example, periodically).

In reality, we cannot distinguish between a receiver and a sender, as a any application is free to send and receive the data and so it could be sender and receiver at the same time. In essence, both Sender and Recipient entities could be integrated into the same participant. For some applications, it is crucial to immediately detect the outsiders injecting traffic into a secure session. In such cases, the sender's identification has to be established before sending each packet.

In our architecture, the traffic encryption keys are arranged hierarchically to form a logical tree, with the participants at the leaves of the tree. The keys at the intermediate nodes can be used for sending messages to sub-groups, for example, while rekeying operation. The key that is at the top of the tree is used for sending any encrypted messages to the whole group. Each participant has all the keys that are in the tree along the path starting from the leaf that corresponds to the participant to the root of the tree. Assuming that the tree is a fully balanced, complete binary tree, the number of the keys that each participant holds in a particular session is only logarithmic in the number of hosts in the group. This prevents the implosion of the secret key database that needs to be maintained at each participant. Below we describe, how each operation is carried out during a multicast session.

- **Join Operation:** A join request is processed by the Group Manager. Any potential participant should send a join request to the group manager. This should be a unicast message to reduce the network traffic. The Group Manager process the request by sending the session description to this client. The session description contains all the security parameters and policy information like which encryption protocol to use, what are the key lengths etc.

  If the client agrees with the session description it sends back a positive acknowledgement to the Group Manager. Now, the Group Manager establishes a secret key with this participant. So, all the participants share a secret key with the Group Manager that is known only to the Group Manager and the participant. The Group Manager stores this secret key in the leaf node that corresponds to that particular participant in the key graph. The Group Manager now issues fresh keys for all keys that are along the path from the new participant to the root. These new keys are also sent to the relevant participants and all the participants are informed about the change in the Traffic Encryption Key that lies at the root of the tree. Below we briefly sketch the algorithm.

  Algorithm for Join Operation:

  1. Group Manager generates a secret key for the new member and places this key at the leaf level in the key graph.
  2. For each key that is in the path from the new member's node (which is at the leaf) to the root, Group Manager generates a new key.
  3. Each of the newly generated keys are transmitted to all the participants who are effected in a bottom-up fashion (i.e each new key is encrypted with the keys that are immediate children to this new key in the key graph and then sent over the network).

- **Leave Operation:** The leave operation will be performed by the Group Manager for one of the following reasons.

  1. A participant voluntarily wants to leave the group.
  2. A participant violates the group's policy and is forced to leave the group.
  3. A participant's keys are compromised and so he should (temporarily) leave the group for securing the ongoing communications.

  To exclude a member, all the keys known to it need to be replaced with entirely new keying material. The Group Manager sends out a message with new keying material which can be decrypted only by the remaining participants but not the member which just left. To achieve this, the keys known to the participant being excluded from the group should be viewed as compromised and they have to be replaced in a bottom-up manner. The Group Manager chooses a new key for the lowest node, then transmits it encrypted with appropriate child keys. In general, the new node keys are encrypted with all appropriate underlying node or leaf keys and sent out to the participants.

Since, we are proceeding from the bottom up, each of the replacement keys will have been replaced similar to the join operation that was described already. The algorithm is sketched below:

Algorithm for Leave Operation:

1. The Group Manager determines the set of keys that the leaving participant knows.
2. For each key in set, Group Manager generates a new key.
3. Each of the newly generated keys are transmitted to rest of the participants in a bottom-up fashion as described above.

If the height of the key graph is more, the number of messages containing updated keys will be more. This is because, if the height of the tree is more, it means that number of nodes in tree are more and since each node corresponds to a key in our scheme, the number of keys that need to be transmitted are proportional. This might might increase the message complexity if proper care is not taken while building the tree. We strongly recommend to increase the arity of the tree (which reduces the height of the tree) and also multiple keys could be sent in one message instead of sending many short messages. This strategy works well in mobile environments where the Group Manager himself is mobile. All it needs to do before leaving the home network is to transfer the Key Graph(encrypted) onto a backup server. This will not congest the network as each key is typically only 128 or 256 bits. This scheme nicely suits some applications that has inherent hierarchy of message transmission like tactical mobile military networks.

- **Periodic Re-keying:** In Periodic Re-keying, we need to replace all the keys that lie at the interior nodes (including the root). The algorithm to for re-keying is same as that described in the Join/Leave except that we replace all the keys in the tree.

8 Related Work

Even though numerous authentication protocols have been proposed and are widely in use on the Internet, the issue of fault tolerance and scalability has not received much attention. With the existing systems, if a server goes down a client will not be able to get services from the server. Our paper addresses the issue of fault tolerance on Local Area Networks and scalability of the system. Our architecture is flexible to accommodate any authentication scheme. Virtual Router Redundancy Protocol [29] addresses the issue of eliminating single point of failure on LANs but the scheme is not scalable for very busy networks. For example, secure transfer of audio or video databases on the Internet take huge amount of bandwidth and hence authenticating each and every packet is time consuming and the Master quickly becomes a bottleneck. Our contribution towards this issue is the proposal of the use of a cluster of nodes rather than a single node. Web servers based on clusters of workstations are being widely used in the corporate networks [1] to service the HTTP requests. We extend this scheme and provide fault tolerance to provide un-interrupted services to the clients.

Existing protocols for secure multicasting are limited to distribute session keys statically and for small groups. Research has been going on in this area to provide efficient solutions for dynamic multicast groups. Below we briefly describe the popular techniques:

Extensions to Diffie-Hellman Key Distribution: There are many extensions proposed [14][6][13] to the 2-party Diffie-Hellman key exchange protocol that was initially proposed in 1976 [9]. Unfortunately, some of the results are only theoretical interest and are completely impractical. The security of some other protocols are not proven at all. A better protocol that uses this idea was discussed in [21]. But this protocol does not provide authentication of the participants and does not handle periodic re-keying which is critical to replace compromised keys.

Iolus: In this system[22], large group is decomposed into several small groups. This reduces the number of participants effected by a key change. Each sub group is independent of other groups and has its own subgroup keying material. Thus, when a member joins or leaves, it joins or leaves only its local sub group. As a result, only the local keys needs to be changed. In this scheme, relay nodes perform admission control and packet rekeying. The drawback of this proposed approach is that it requires full trust on these relay nodes and relay node failures are not dealt with.

Multicast Trees: A tree based approach similar to what we proposed was mentioned in [11]. This is currently an Internet RFC. This scheme completely ignores the inherent computing characteristics and problems of mobile networks. This document also ignores the periodic-rekeying operation. It does not define what exactly the key material consists of, when the keys expire and has no concept of assinging priorities to the keys which might lead to confusion as of which key to use if a user has more than one key at a time. Similar approaches are discussed in [26][25].

Secure Lock: This scheme[7] is based on the Chinese Remainder Theorem. In this, only the keys of the authorized users can retrieve the group key and only one copy of the encrypted message is sent. The disadvantage of this approach is that retrieving the group key is an expensive operation.

Group Key Management Protocol: Group Key Management Protocol(GKMP) was proposed in the Internet draft [13]. In this protocol a group controller processes the join requests and sends the group key individually to the newly admitted members. This protocol is like a unicast key distribution protocol and does not handle the dynamic membership changes.

Cliques: The approach described in[19] improves the system capability to distribute the session keys, but the solution is not scalable because of large message complexity consuming large bandwidths.

9 Conclusions and Future Work

In this paper we presented two schemes for achieving fault-tolerant authentication in mobile environments. The two techniques presented are based on the the same basic philosophy to handle failures – using backups, but architecturally, they are different; one is a flat model and the other one is a tree-based model. We summarized the differences between the two approaches in table 1. To address the scalability of the proposed system, a cluster based enchancement to the discussed model it presented and evaluated with the simulation studies conducted.

Even though the proposed solutions solve the problem, some issues need further study. In the tree-based model, key priorities need to be computed based on various factors like communication delays, processing speeds etc. Further experiments need to be conducted to discover the set of parameters that effect the performance of the system and study how the priorities depend on these factors. Another issue that needs to be addressed is, how should the secret database be partitioned and replicated, if the need arises, so that system performance is optimal.

We discussed a framework for efficient key management in dynamic multicast sessions. We are planning to build a prototype system to study the performance of the scheme and assess it's feasibility in large secure multicast sessions. We are exploiting tools like **ns** to generate large topologies to simulate scenarios for measuring the re-keying operation discussed.

References

[1] S. D. G. A. Fox and et al. Cluster-based scalable network services. In *In Proceedings of the Sixteenth ACM Symposium on Operating System Principles*, October 1997.

[2] A. R. A. Mankin and et al. Ietf criteria for evaluating reliable multicast transport and application protocols. *University of Southern California*, July 1998.

[3] A. Ballardie. Core based trees multicast routing architecture. *Internet Draft*, May 1997.

[4] S. Bellovin. Security problems in tcp/ip protocol suite. *ACM Computer Communications Review*, Mar. 1989.

[5] B. Bhargava, S. Kamisetty, and S. K. Madria. Fault tolerant authentication in mobile computing. In *International Conference on Internet Computing (IC, 2000)*, June 2000.

[6] M. Burmester and Y.Desmedt. A secure and efficient conference key distribution system. In *Advances in Cryptology, EUROCRYPT*, 1994.

[7] G. Chiou and W. Chen. Secure broadcasting using the secure lock. *IEEE Transactions on Software Engineering*, Aug. 1989.

[8] S. Deering. Host extensions for ip multicast. *Internet Draft*, Feb. 1997.

[9] W. Diffie and M. Hellman. New directions in cryptography. In *IEEE Transactions on Information Theory*, November 1976.

[10] R. Droms. Dynamic host configuration protocol. *RFC 1531*, Oct. 1993.

[11] D. W. E. Harder and R. Agee. Key management for multicast: Issues and architectures. *RFC 2627*, June 1999.

[12] D. Estrin and et al. Protocol independent multicast-sparse mode. *Internet Draft*, Mar. 1997.

[13] H. Harney and C. Muckenhirn. Group key management protocol (gkmp) architecture. *RFC 2094*, July 1997.

[14] D. T. I. Ingemarsson and C. Wong. A conference key distribution system. In *IEEE Transactions on Information Theory*, September 1982.

[15] D. Johnson and C. Perkins. Route optimization in mip. *Internet Draft, draft-ietf-mobileip-optim-03, IETF Mobile IP Working Group*, Nov. 1995.

[16] S. Kent and R.Atkinson. Ip authentication header. *Internet Draft, draft-ietf-ipsec-auth-header-07.txt*, July 1988.

[17] S. Kent and R.Atkinson. Ip encapsulating security payload. *Internet Draft, draft-ietf-ipsec-esp-v2-06.txt*, July 1988.

[18] P. S. Kruus. A survey of multicast issues and architectures. *Technical Report, Naval Research Laboratory.*

[19] G. T. M. Steiner and M. Waidner. Cliques: A protocol suite for key agreement in dynamic groups. In *Research Report RZ 2984, IBM Zurich Research Lab*, December 1997.

[20] D. Maughan and et al. Internet security association and key management protocol (ISAKMP). *Internet Draft*, Feb. 1997.

[21] G. T. Michael Steiner and M. Waidner. Diffie-hellman key distribution extended to group communication. In *3rd ACM Conference on Computer and Communications Security, Delhi, India*, pages 31–37, 1996.

[22] S. Mittra. Iolus: A framework for scalable secure multicasting. In *In Proceedings of ACM SIGCOMM '97*, pages 277–288, September 1997.

[23] P. D. Mohit Aron and W. Zwaenepoel. Efficient support for p-http in cluster-based web servers. In *In Proceedings of 1999 Annual Usenix Technical Conference*, June 1999.

[24] J. Moy. Multicast extensions to ospf. *RFC 1584*, Mar. 1994.

[25] G. Noubir. Multicast security. In *Technical Draft, European Space Agency*, September 1998.

[26] G. Noubir. A scalable key-distribution scheme for dynamic multicast groups. In *European Research Seminars on Advanced Distributed Systems*, April 1999.

[27] C. Perkins. IP mobility support. *RFC 2002*, Oct. 1996.

[28] C. P. S. Deering and D. Waitzman. Distance vector multicast routing protocol. *RFC 1075*, Nov. 1988.

[29] D. W. S. Knight and et al. Virtual router redundancy protocol. *RFC 2388*, Apr. 1998.

[30] B. Schneier. *Applied Cryptography.* John Wiley and Sons, 1995.

[31] H. Schulzrinne and et al. Rtp: A transport protocol for real time applications. *RFC 1889*, Jan. 1996.

[32] W. Simpson. The point-to-point protocol. *RFC 1661*, July 1994.

[33] J. D. Solomon. *Mobile IP :The Internet Unplugged.* Prentice Hall Series in Computer Networking and Distributed Systems, 1997.