Journal of Universal Computer Science, vol. 11, no. 1 (2005), 37-55
submitted: 15/7/04, accepted: 6/12/04, appeared: 28/1/05 © J.UCS

Sliding Window Protocol for Secure Group Communication in Ad-Hoc Networks

In Joe Khor
(Oklahoma State University Tulsa, USA)

Johnson Thomas
(Oklahoma State University Tulsa, USA
jpt@cs.okstate.edu)

Istvan Jonyer
(Oklahoma State University Tulsa, USA
jonyer@cs.okstate.edu)

Abstract: Existing ad hoc routing protocols are either unicast or multicast. In this paper we propose a simple extension to the Dynamic Source Routing Protocol (DSR) to cater for group communications where all node addresses are unicast addresses and there is no single multicast address. The proposed sliding window protocol for multiple communications results in significant improvement in total packet delivery. Due to the high frequency of mobility, attrition and reinforcement in ad hoc networks, in order to preserve confidentiality, it becomes necessary to rekey each time a member enters or leaves a logically defined group. We compare our group rekeying rate on sliding window protocol versus other kinds of Rekeying algorithms. The proposed sliding window protocol performs better. The proposed sliding window is therefore simple and improves both communications and security performance.

Keywords: Ad Hoc Network, DSR Routing Protocol, Re-keying performance, Secure Group Communications
Categories: C.2.0, D.4.6, I.6

1 Introduction

Mobile ad hoc networks have properties such as no infrastructure, arbitrary movement, scarce resources and limited power. These properties determine that ad hoc networks need special protocols. Although multicasting in ad hoc networks has been proposed, a user may wish to individually communicate with several distinct users at the same time. For example, rather than a multicast communication concurrently with a remote group of students during office hours, a professor may wish to communicate with a group of students in an interleaved fashion, such that the message to each student is private and not seen by other students. There is therefore no single multicast group address as each node has its own unique address. This is a form of group communications as the nodes all belong to a group (such as a 'office hours' group) where there is interleaved multiple individual communications between multiple destinations and the same single source, all having different unicast addresses where each node may receive and send different messages. A node may therefore need to discover routes to multiple nodes at the same time and multiple nodes may join or leave the group at the same time. Existing routing protocols are either multicast or unicast. We propose a mixed protocol which makes use of unicast addressing for group communication.

Typically, Ad Hoc networking research focuses on either communications performance (Quality of Service) or security exclusively. Very rarely, has the impact of a protocol on both performance and security been reported in the literature. In almost all cases, an improvement in one aspect is at the expense of the other. Our objective is to propose a routing protocol that:

- builds on existing mobile ad hoc routing protocols for interleaved group communications where there is no multicast address. For deployment and other practical reasons, our objective is not to propose a complex new protocol, but a simple protocol based on current protocols.
- will improve communication performance when compared to existing unicast routing approaches.
- will improve security in terms of rekeying overhead in group communications as opposed to current methods for group security.

The significance of this work lies in the simplicity and minimal modifications to existing protocols, and the enhancement of two key characteristics of mobile group communications - performance and security. Our proposed protocol is an extension of the Dynamic Source routing Protocol (DSR) to include group communication. We propose a pseudo-sliding window protocol for multi-communications in an ad hoc environment. We extend the sliding window approach to secure group multi-communications. We show that the sliding window paradigm is effective for both communication performance and security rekeying in group multi-communications. After a brief overview, we introduce in section 3 the sliding window protocol in a multi-communications group environment. In section 4, the performance of the sliding widow protocol is evaluated. In section 5, we extend the sliding window protocol to incorporate an inter-group rekeying strategy that reduces the performance penalty caused by nodes moving between groups.

2 Related Work

Ad hoc mobile protocols must deal with the limitations of high power consumption, limited resources, low bandwidth and high error rates. Relatively few papers have addressed the issue of reliable multicasting in a MANET. The Reliable Adaptive Lightweight Multicast protocol [Tang, 2002] is a multicast transport layer protocol that achieves relatively high packet delivery by throttling traffic based on congestion experienced by a feedback receiver. [Gopalsamy, 2002] describes RMA, a Reliable Multicast Protocol based on the assumption that senders know the identities of all receivers and achieves reliability by explicit ACKs from all receivers. [Shu, 2002] discusses how to assure packet delivery in MANETs using error correction codes. Bagrodia et al. [Bagrodia, 2000] simulated several multicast routing protocols developed specifically for MANET, some tree-based, some based on a mesh structure. On-Demand Multicast Routing Protocol (ODMRP) ODMRP [Gerla, 2001] is mesh based, and uses a forwarding group concept.

A number of routing protocols have been proposed. These protocols can be classified into three different groups: global/proactive, on-demand/reactive and hybrid. In proactive routing protocols, the routes to all the destinations (or parts of the network) are determined at the start up, and maintained by using a periodic route update process. Examples include DSDV [Perkins, 1994] and WRP [Murthy, 1995]. In reactive protocols, routes are determined when they are required by the source using a route discovery process. Examples include AODV [Das, 2002] and DSR [Johnson, 2002]. Hybrid routing protocols combine the basic properties of the first two classes of protocols into one. That is, they are both reactive and proactive in nature. Examples include ZRP [Haas, 1999] and DST [Radhakrishnan, 1999]. A good review of routing protocols can be found in [Abolhasan, 2004].

Our approach is similar to [Gopalsamy, 2002] in that it assumes that senders know the identities of all receivers and achieves reliability by explicit ACKs from all receivers. However, these works focus on multicast group communications, whereas our work is novel as it proposes individual interleaved multi-communications using unicast addresses.

A number of key management algorithms have adopted a hierarchical structure [DeCleene, 2001], [Wong, 2000], [Harney, 1999]. Broadly, these rekeying algorithms operate by hierarchically dividing the key management domain into smaller administratively scoped groups. Throughout the domain, a Domain Key Distributor (DKD) generates the data key used by the session for encrypting the data. Whenever a new member joins a current session or an existing member leaves a session, a new data key must be generated and distributed to ensure both forward and backward confidentiality. The domain is further divided into disjoint groups. A group is unique in that movement within the group does not require any additional signalling with regard to rekeying. A group can be either logically or geographically defined. Within each group, a Group Key Distributor (GKD) is responsible for distributing the data key to members within that group. Because the distribution of the data key within a group must itself be secure, group-local keys are used by the GKD to distribute a new data key to members within the group. Approaches for intra-group rekeying include Public Key Infrastructure (PKI), secure multicast, and logical tree-based algorithms such as [Rodeh, 2000], [Lazos, 2003]. Mobility impacts performance only when members cross between groups. Without GKD reassignment, rekeying messages must cross heterogeneous network boundaries resulting in additional performance degradation. Consequently, member movement between groups requires a coordinated transfer of the security relationships. To illustrate, consider two partners providing broadcast services for users in two overlapping geographic groups. Users moving within each group are managed by their local GKDs and require no coordination between the two partner broadcasts. On the other hand, when a user crosses from one group into another, then the security relationships must be transferred between the partners.

3 Sliding Window Protocol

We extend the DSR protocol for our work. DSR is a well studied protocol and there are a number of improvements to DSR proposed. We propose a pseudo-sliding window protocol for multi-communication that would serve many route requests to different destinations within a certain timeout period while avoiding network collisions or interferences and congestion. We have been liberal with the usage of the term ‘sliding window’ as strictly speaking it is a sliding window scheme where the window moves in multiple units rather than single units. Henceforth we refer to the sliding window DSR protocol as sliding window and to the normal DSR as DSR.

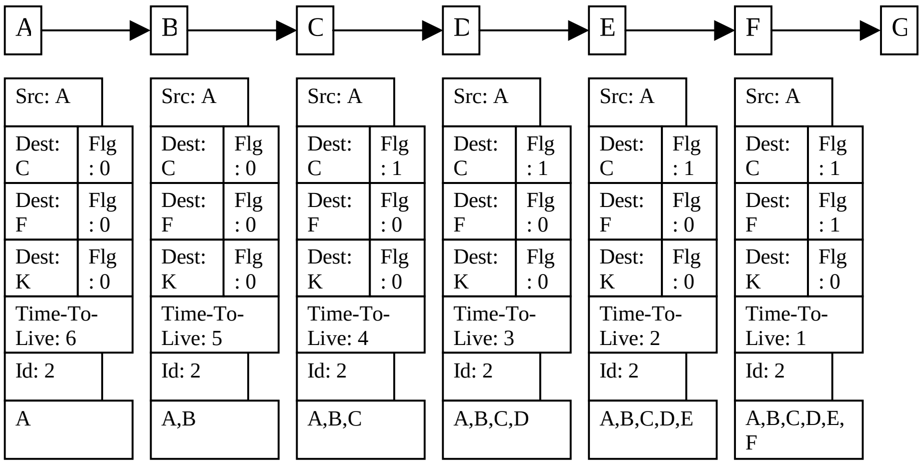
3.1 Outline of Pseudo-sliding window protocol

It is beyond the scope of this paper to present the sliding window protocol in detail. We therefore informally describe the sliding window scheme for route discovery only. There may be different windows for data transfers. Multiple route requests are sent at the same time. Valid route replies are stored in a buffer. With each new valid reply or replies, the window advances. If a new reply arrives within the time-out period for a destination that is already in the window, then there are two possibilities. If the new reply is a longer route than the one currently in the window, the new reply is buffered and only used if the current route fails. If the new reply points to a shorter path, it replaces the old entry in the window.

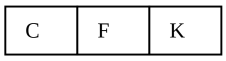
A route request is a broadcast packet that is received by all nodes within range of the node transmitting the request. In outline each request contains the following information: the initiator of the request, the destinations, the time-to-live parameter and a unique request id. Each route request also contains a record listing the address of each intermediate node through which this particular copy of the route request has been forwarded. A timer is started when a route request is transmitted. If a timeout occurs before a route reply is returned, the route discovery for the affected nodes is retransmitted (see below)

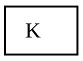
In fig. 1(a) below, routes are requested for destinations C, F and K. The sliding window will have three entries, C, F and K (fig 1(b)).The figure shows the flow of route discovery packets in one path in the network. When another node receives this Route Request, it checks if it is a target of the Route Discovery. If it is not (such as node B below), it decrements the Time-to-Live value. It next checks the Time-to-Live value and if it is greater than 0, the request is forwarded. If the node is a target of the route request (such as node C below), it sets the flag for node C in the route request packet, decrements the Time-to-Live value and forwards the request, if the value is greater than 0. The request is not forwarded if the Time-to-Live is 0 or if all the destination nodes flags have been set in the route request of if a route request with the same id had been received earlier by the node.

If a route request is not forwarded, a "Route Reply" is returned to the initiator of the Route Discovery, giving a copy of the accumulated route record from the Route Request; when the initiator receives this Route Reply, it processes the route record and caches the routes in its Route Caches for use in sending subsequent packets to the destinations. The route A,B,C will be cached for destination C and A,B,C,D,E,F for destination F. The sliding window at the origin will remove C and F (Fig 1(c)). As in the DSR protocol, in order to reduce the overhead from Route Discoveries for nodes which may not be reachable, a node should use an exponential back-off algorithm to limit the rate at which it initiates new Route Discoveries for the same target.



(a) Route requests for destination C, F and K





(b) Window when request has been sent

(c) Window after reply has been received

Figure 1: Route Discovery in Sliding Window

3.2 Determination of time-to-live.

Time-to-Live is the number of hops a packet is allowed to traverse before it is discarded and a route reply returned. A large Time-to-Live value will result in route requests travelling for long distances resulting in reduced performance. A small Time to Live may not generate the routes to some destinations which are further away. We consider the network as a random graph $G(n, p_l)$, a graph of n nodes for which the probability that a link exists between any two nodes is p_l. Erdos and Renyi showed that for monotone properties of a graph $G(n, p_l)$, there exists a value of p_l over which the property exhibits a “phase transition”, i.e. it abruptly transitions from “likely false” to “likely true” [Chan, 2003]. Hence, it is possible to calculate some expected degree d for the vertices in the graph such that the graph is connected with some high probability p, where p = 0:999, for example. Eschenauer and Gligor [Eschenauer, 2002] calculate the necessary expected node degree d in terms of the size of the network n as:

$$d = \left(\frac{n-1}{n}\right)(\ln(n) - \ln(-\ln(p)))$$

Since the models of connectivity are probabilistic, there is always the chance that the graph may not be fully connected.

We assume a star connectivity model with degree d in our work. Each node is at the centre of a star topology whose connectivity is determined by the degree d. Given the degree *d* and the network size *n* (*n* number of nodes), the number of hops *h* may be calculated using the following recurrence relation:

$$n'_h = (n'_{h-1} - n'_{h-2})(d-1) + n'_{h-1}$$

with the initial conditions $n'_1 = 2$ and $n'_2 = n'_1 + (d-1)$.
where *n'* is the number of nodes being evaluated. If there are only two nodes in the network the number of hops is 1 (initial condition $n'_1 = 2$). For example, a network consisting of 60 nodes and $d = 4$ gives a *h* value of 5. As the network is mobile a precise Time-to-Live value cannot be determined. Our approach is therefore simply a heuristic to guide the selection of the Time-to-Live value.

We have assumed here that each node has a degree *d* with a certain probability. All the nodes in the network may have only two neighbours and therefore the hop distance becomes *n*-1. Alternatively at the other extreme, the hop distance is one when all nodes a re immediate neighbours of another node. Therefore for our purposes we doubled the *h* value for the time to live. For our simulations this proved to be more than sufficient. Other approaches may be used to determine the Time-To-Live value.

If a route is not returned, we used an exponential back-off algorithm to limit the rate at which it initiates new Route Discoveries for the same target. In other words, the algorithm doubles the timeout between each successive Discovery initiated for the same target. A similar approach is used to increase the Time-To-Live. The Time-To-Live is doubled for each successive Discovery initiated for the same target.

3.3 Advantages of Sliding Window Protocol

Space limitations prevent a detailed description of the protocol. The proposed protocol has a number of advantages. This protocol reduces the number of Route request messages at the expense of larger route request packets. This approach results in fewer collisions and the discovery of routes to multiple destinations with a single route request.

Each node maintains a route cache where it caches the source routes that it has learned. We measure the system performance in terms of cost which are comprised of the system resource cost (R$/packet) and the delay cost (D $/time unit). The system resource cost indicates the cost of processing the route request and that of transmitting them from the source to the destination. The delay cost indicates how much time the nodes wait for the source to start data transmission.
The total cost of a single route reply is:

$$C = \lambda * R * s + D * (t_1 - t_0)$$

where λ is the time associated with resource usage, s is buffer size in bytes and t_1 is time at which data transmission can commence for the first node (after first route reply received) and t_0 is start-up time when the route discovery process starts. This cost is replicated for each new destination. The total cost for *n* destinations therefore becomes:

$$\sum_{i=1}^{i=n} \left[\lambda Rs + D(t_i - t_0)\right]$$

In the sliding window scheme the sender broadcasts its multiple route requests in one packet. Multi route replies are sent back to the source by either the destination node(s) or another node(s) that knows the route to the destinations. The source node could start the transmission as soon as a route is available. Since sliding window protocol is capable of handling multi replies in one time, the delay cost drops to the cost of a single route reply. Now the total cost of route replies for *n* destinations is

$$\sum_{i'=1}^{i'=n} \lambda Rs + D(t_{i'} - t_0)$$

$t_{i'}$ is the time taken before data transmission can commence; this is defined by the time for a route reply containing paths to multiple destinations. This is dependent upon the Time-to_live value (usually, but not always, as all paths may be discovered or the node may have received a packet with the same id earlier). Therefore as long as $D(t_{i'} - t_0) < \sum_{i=1}^{i=n} D(t_i - t_0)$, the cost will be less with the proposed protocol. Even if on average $t_{i'} > t_{i}$, as multiple routes are obtained in one route request, the total cost for multi-communication is less than for the normal scheme. Clearly, this approach is attractive only for applications which require the establishment of routes to multiple destinations concurrently. The proposed protocol is an extension to the Dynamic Source Routing Protocol (DSR), thus satisfying one of our objectives.

4 Experiments

In DSR, Route Discovery and Route Maintenance each operate entirely "on demand". The route discovery process typically involves network wide flooding of a route request and waiting for a route reply. When a node with a route to the destination (or the destination itself) is reached a route reply is sent back to the source node using link reversal or by piggy-backing. Caching provides a mechanism for generating a route reply from an intermediate node en route to the destination. Marina et. al [Marina, 2001] identified three deficiencies with the Dynamic Source Routing (DSR) protocol and proposed a number of extensions to DSR to overcome these deficiencies. The different schemes proposed by Marina are as follows:

Base DSR – In the basic DSR protocol, the sender knows the complete hop-by hop route to the destination. These routes are stored in a route cache. The data packets carry the source route in the packet header. It uses Route Discovery to determine a route it doesn’t have, and route reply is routed back to the original source. Route error packet is generated if the source route is broken resulting in failure in data transmission over a link. Route error is unicast back to the source.

Negative cache – In this extension to the base DSR scheme, every cache caches the broken links seen recently via the link layer feedback or route error packets. If a node is to forward a packet with a source route containing a broken link, the packet will be dropped and a route error packet will be generated.

Wider Error In this approach, route errors are transmitted as broadcast packets instead of unicast packets which is what the traditional Base DSR does. The node initially detecting the link breakage broadcasts the route error packet containing the broken link information. Upon receiving a route error, a node updates its route cache so that all source routes containing the broken link are truncated at the point of failure.

Adaptive route expiry A timer based approach is based on the hypothesis that routes are only valid for a specific amount of time from their last use. Each node in a cached route now has an associated timestamp of last use. The timestamp is updated each time the cached route is seen in a unicast packet being forwarded by the node.
DSR+. The variant of DSR with all three techniques, Negative cache, Wider Error and Adaptive route expiry combined.

We implemented the above five schemes for the DSR-based sliding window protocol. In other words, DSR-based Sliding Window with Base DSR, DSR-based Sliding Window with Negative Cache, DSR-based Sliding Window with Wider Error, DSR-based Sliding Window with Adaptive route expiry and DSR-based Sliding Window with DSR+. We then compared these protocols with each other and with the five different versions of the DSR protocol implemented by Marina [Marina, 2001]. In particular we investigated the packet delivery percentages for the DSR protocols with the sliding window protocols. Moreover, the window sizes and the timeout were varied to find out how window size and timeout periods affect the performance of the sliding window protocols. The protocols were simulated on Unix platform and the Java programming language was used for implementation purposes. The simulation program has two main components. Event Producer (EP) and the Reply Processing Protocol (RPP). The EP is responsible for generating the stream of events. Two key performance metrics were evaluated:

1. Packet delivery percentage. This is the percentage of data packets that are received at the destinations over those sent at the source.
2. Retransmission throughput. This is the percentage of the data packets delivered to the destinations after previous failed attempts to deliver the packets.

The packet delivery percentage is the most important metric to evaluate the performance of an ad hoc routing protocol [Lou, 2002]. We therefore evaluate the number of packets delivered at the first attempt and also those delivered requiring more than one attempt. It is beyond the scope of this paper to present all the results in detail. We summarize the main results. The number of destination nodes was randomly generated, that is, the number of concurrent multiple destinations for which routes need to be discovered ranged from 1 to a maximum range. This allowed us to test the robustness of the algorithms with different number of destinations. If the number of destinations were the same for each route discovery, the performance of the algorithms would be much more impressive, but that would be an unrealistic scenario. In the real world, at different times there may be a varying number of destination nodes for route discovery as nodes lose contact due to mobility, join and leave groups etc.

4.1 Packet delivery ratio for Sliding Window and DSR

We compared the packets delivered with varying pause time for the five different algorithms with respect to the sliding window protocol and the DSR protocol [Marina, 2001] (Marina et al). A low pause time indicates a highly mobile network whereas a high pause time indicates a relatively static network. Results show that packet delivery percentage decreases when the network is in high mobility status (pause time is low). On the other hand the packet delivery percentage increases at low mobility status (pause time is high). The simulations show sliding window sends over 7.5%, 12.5 % and 8.7% more packets than DSR [Marina, 2001] (Marina et. al) for base DSR algorithm when pause time is 10 seconds, 100 seconds, and 200 seconds respectively. As for negative cache, sliding window delivered over 6.9%, 13.5% and 6.7% more packets than DSR when pause time is 10 seconds, 100 seconds, and 200 seconds respectively. Sliding window sends over 10.4%, 15.2%, 11.6% and 4.2% more packets than DSR for DSR+ when pause time is 10 seconds, 100 seconds, 200 seconds and 300 seconds respectively. As for wider errors, sliding window delivers over 9.8%, 16.2%, 9.3% and 1.43% more packets than DSR when pause time is 10 seconds, 100 seconds, 200 seconds and 300 seconds respectively. As for adaptive route expiry, sliding window sends over 11.7%, 15.1%, 10.6% and 3.3% more packets than DSR for base DSR algorithm when pause time is 10 seconds, 100 seconds, 200 seconds and 300 seconds respectively. See fig. 2

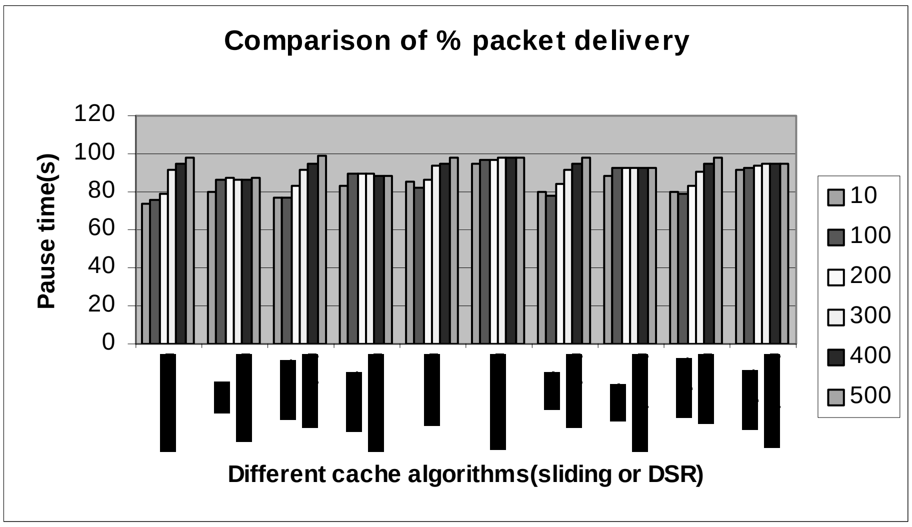


Figure 2: Comparison of delivery fraction with varying pause time for five different algorithms with respect to sliding window protocols and DSR [Marina, 2001]

	Average percentage improvement of Sliding Window over DSR
DSR+	10.35%
Adaptive route	10.18%
Base DSR	9.60%
Wider errors	9.20%
Negative cache	9.03%

Table 1: Average improvement of total packet delivery using sliding window scheme

To summarize, results show that using sliding window protocol helps to improve the total packet delivery rate when compared to DSR [Marina, 2001]. Our simulation results show improvement of total percentage of packet delivery when using a DSR sliding window over a non-sliding window scheme. The results of improved total packet delivery percentage when compared to DSR [Marina, 2001] are shown in Table 1.

4.2 Other performance measures for Sliding Window

We outline some of the performance measures obtained for the five sliding window algorithms. We experimented with Retransmission throughput, window size and timeout for the different algorithms of the sliding window protocol. Simulation results show that the routing retransmission throughput increases when the network is in high mobility status. Comparing the results for pause time 10 seconds to 100 seconds, the retransmission throughput is almost doubled for DSR+ and Adaptive Route. The percentage retransmission increases with a factor of more than 0.3 and 0.4 for both DSR and wider errors respectively. The other algorithm shows an increment of retransmission with a factor around 0.2 These results indicate that the sliding window protocol does increase the retransmission throughput for the different algorithms when the network is at high mobility status. The effective period of using sliding window protocol is significant at pause time 10 seconds to 100 seconds.

We also investigated retransmission throughput for different window size when using the sliding window algorithms. The window sizes were varied from 10 slots to 100 slots. All these algorithms show the same pattern of behaviour in that the retransmission throughput increases as the window size increases. Simulations show that retransmission throughput is higher when the window size is bigger, with the most distinctive results coming from Base DSR, Negative cache, Adaptive route and wider error. The improvement in retransmission throughput are 65.2%, 30.9%, 18.4% and 15.5% respectively, considering the variables of initial window size of 10 slots to the final size of 100 slots. As DSR+ is a more stable caching algorithm, the retransmission throughput does not seem to be affected by the window size. Base DSR algorithm is an unstable algorithm as the window size highly affects its retransmission throughput.

Finally we investigated the retransmission throughput versus timeout used in sliding window protocol. The timeout simulation ranged from 5 seconds to 50 seconds. All these algorithms show the same pattern of behaviour in that the retransmission throughput increases as the timeout period increases. Base DSR, Negative cache, Adaptive route and wider error show increment percentages of 78.6%, 44.8%, 64.0% and 43.5% respectively. DSR+ is a more stable caching algorithm; the retransmission throughput does not seem to be affected much by variation of timeout used. The increment of percentage retransmission for DSR+ is only 16.66% from initial 5 seconds to final 50 seconds. Base DSR algorithm showed the highest improvement in the percentage of retransmission (78.6%) as the timeout period increased.

To summarize, these results indicate that the retransmission throughput is better at high mobility, a bigger window size results in a better retransmission throughput and a bigger timeout value also results in a better retransmission throughput. Furthermore DSR+ performs the best of these algorithms and Base DSR the worst.

5 Sliding window for group security

We extend the sliding window scheme to secure group communications. In group communications, a critical element in controlling information is to ensure that only the appropriate individuals have the cryptographic keys that enable them to decode the disseminated information. Therefore to maintain forward confidentiality, when a member leaves a session or group, the remaining members must be rekeyed to ensure that the departing individual cannot listen in on the future communications. Similarly, backward confidentiality requires rekeying when a new member joins an existing session or group. Otherwise, the new member would be able to decrypt any past archived exchanges for which he/she was not authorized. Since data cannot be exchanged while a member's data keys are being updated, the challenge for any key management system is how to generate and distribute new keys such that the data remains secure while the overall impact on system performance is minimized. Hierarchical group key management schemes [DeCleene, 2001] have been proposed for scalable networks. As there are a number of such group key management schemes, we base our work on [DeCleene, 2001]. Our objective is not to describe a new hierarchical key management scheme, rather it is to show that the sliding window improves key management for hierarchical key management schemes. Due to space limitations, the details of the key management scheme can therefore omitted and can be found in [DeCleene, 2001]. Each group *i* has a key distributor, the Group (or area) Key Distributor (GKDi).

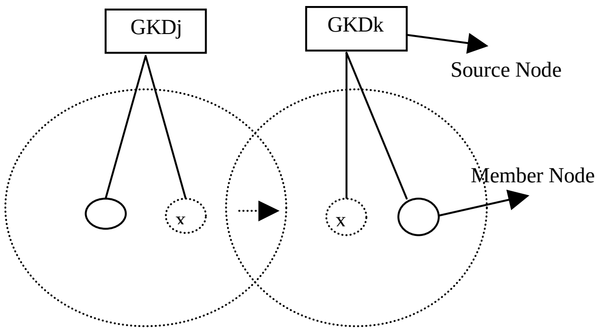


Figure 3: Mobility model with baseline rekeying

In Baseline Rekeying [DeCleene, 2001] [Zhang, 2002] (fig 3) a member leaving the group notifies the local Group Key Distributor (GKDj), which halts the current data transmission. Next the local GKDj updates the new group key for the remaining members by securely unicasting based on their pairwise shared ID key. Even though the member left the group, it still holds the old group key but it is invalid since the group key has been updated. Once this is updated securely, a new data key can be broadcast to all members in the same group. At this point, data transmission resumes using the data key. A member entering the group notifies the local Group Key Distributor (GKDk), which halts the current data transmission. Next the local GKDk unicasts the new group key to newly joined member. Then the local GKDk broadcast the new group keys to all members in the same group. See [Zhang, 2002] [DeCleene, 2001] for details.

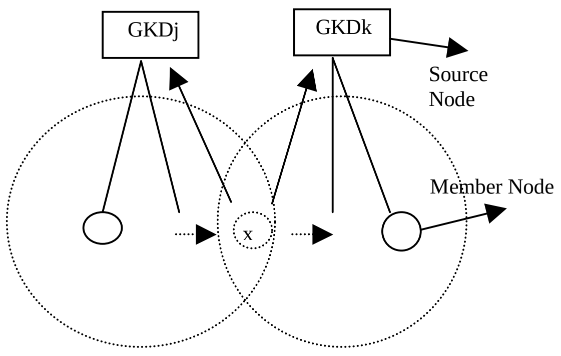


Figure 4:Mobility model with immediate rekeying

Immediate Rekeying [Zhang, 2002] [DeCleene, 2001] (fig 4) extends the baseline algorithm by adding explicit semantics for a hand-off between groups. The member initiates a transfer by notifying the affected groups. Each group only updates the local key upon the moving node arriving at the subset group (if there is one) of both groups. No data key is generated and the data transmission continues uninterrupted.

When a node x wants to leave an group, it sends a "transfer" message to GKDj. GKDj unicasts new group key to remaining members of group j. GKDk unicasts new group key to node x and GKDk sends the new group key to existing members.

Both baseline and immediate rekeying algorithms rekey the local groups as soon as members transfer. As a result, a member that moves rapidly between two groups may cause repeated local rekeying.

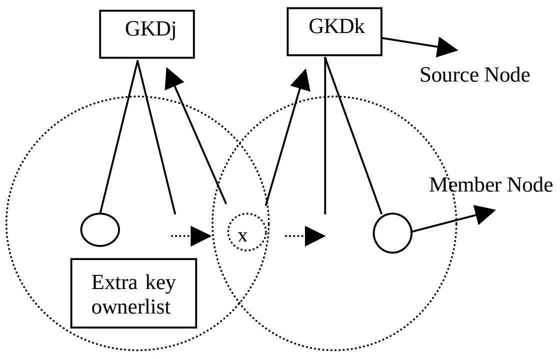


Figure 5: Mobility model with delay rekeying

Delayed algorithms [Zhang, 2002] (fig 5) postpone local rekeying until a particular criterion is satisfied (such as after a specific period). Members moving between multiple groups may accumulate multiple group keys and reuse these keys when they return to a previously visited group. In delayed rekeying, each GKD maintains a list of members that have left the group but still hold valid keys for the group. When a member transfers, the group that the member is entering is rekeyed to prevent from falsely transferring into a group to get access to the old keys (backward confidentially). For the departed group, GKD does not rekey but instead adds the member to the Extra Key Owner List (EKOL). When a member returns to a group, it is checked against EKOL and no new keys are generated if it is on the list. The list is reset whenever a local rekey occurs such as the arrival of a new node not in the EKOL list. Data transmission stops when the list is reset.

5.1 Sliding Window Rekeying

In a group environment multiple nodes may join and leave a group at the same time. When one or more nodes inform the GKDi that they are joining a group, the GKDi uses the sliding window protocol to distribute the group and data keys. In other words, multiple destinations are targeted in one packet as described in section 3.1 and a sliding window is kept at the GKDi.

The sliding window can handle multiple leaves from and multiple entries to a group. With the sliding window, the time-out (section 3.1) will make sure local keys remain valid only for a fixed period of time. There is therefore no need to reset the list and stop data transmission when nodes arrive or leave. If nodes leave a group, the keys simply expire at timeout and there is no need to stop transmission. Similarly, when new nodes arrive without EKOL entries, there is no need to stop data transmission and reset the list. Instead at the end of the timeout period, the EKOL list is updated. The delayed rekeying handles only one request at a time and only updates the EKOL list when the number of owners on the Extra Key Owner List exceeds a specified threshold or the number of keys held by a particular node exceeds a given threshold. In the delayed rekeying scheme, if the membership on the EKOL or number of keys held by a particular node do not exceed the specified threshold over a long period of time, the whole network becomes more insecure since the local keys will remain valid for a long period of time. The Extra Key Owner List timeouts using the sliding window protocol, thus making the network more secure. In the proposed Sliding window (fig 6), rekeying nodes x and y send “transfer” message to GKDj saying they are leaving the group. GKD_j adds node x and y to its local sliding window buffer. (as the sliding window is able to handle more than one input at one time).GKD_k (the group x and y are joining) checks if node x and node y are on its local buffer

- If yes, then do nothing. Nodes already have a valid key for the group.

- If no, at timeout reset Extra Key Owner List and rekey all members in the group.

- The Sliding window works on a time-out and issuing a new group key will ensure that no member outside the group hold a valid key after a fixed period of time.
- The new local key could be added anytime. The old local key could be searched through anytime as long as it is within the same timeout period. The EKOL is reset at the end of the timeout period.
- There is no need to stop data transmission if new nodes arrive or nodes leave a group.
- Multiple transfers can be handled by the sliding window.

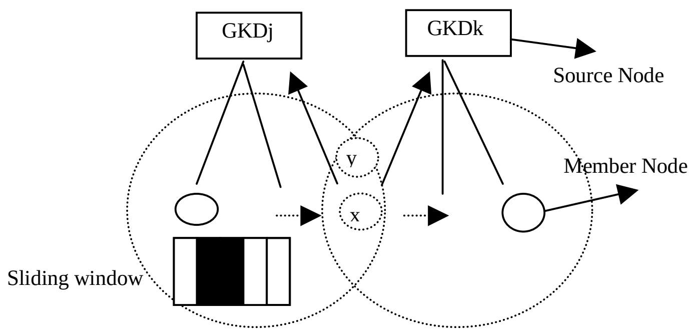


Figure 6: Mobility model with sliding window protocol

The behaviour of the algorithms is depicted in Table II where nodes leave and join per a Poisson process (γ). Upon departing a group, a node has a probability of (p) of transferring to another group. As nodes are independent., each GKD behaves as an M queue. Let T denote the period for rekeying and data transmission. We consider the following metrics:

- Rekeying rates (Rd, Rg) measures the rates at which the data and group keys are generated respectively.
- Mean number of extra-keys (Km, Kg) measures the average number of valid extra keys held by a member outside the group; and the average total number of valid keys held by all members outside the group respectively.

The baseline algorithm performs worst whenever there is any amount of mobility, i.e. p>0. The sliding window protocol has reduced rekeying group rate Rg from 2γ /M +1 to 2γ / (M+1/T). This is a significant improvement on rekeying rate.

	Data Rekey Rate (Rd)	Group Rekey Rate (Ra)	Km	Kg
Baseline	2γ(1-p)	2γ / [M * (1-p)+ (1/T)]	0	0
Immediate	2γ	2γ / [M * (1-p) + (1/T)]	0	0
Delayed	2γ	2γ / M	> 0	> 0
Sliding window	2γ	2γ /(M+ 1/T)	>0	>0

Table 2: Performance comparison of Rekeying Algorithms

The sliding window keying strategy reveals a better rekeying rate. The ability of sliding window to handle multiple requests simultaneously means that it scales well. In the case of thousands of nodes involved in the group communication, where nodes leave and enter dynamically, the baseline and immediate rekeying strategy may not be able to accommodate the workload due to high rekeying rate. The Extra Key Owner

List (EKOL) cache scheme with delayed rekeying will become less secure because of the large number of nodes (and hence keys) involved.

5.2 Secure sliding window protocol experiments

In the simulation of different security algorithms below we assume that arrivals are described by a Poisson process, that time spent in a group is exponentially distributed, and that members traverse groups in a probabilistic manner [Zhang, 2002]. We compare the baseline, immediate, delayed and sliding window rekeying algorithm respectively.

5.2.1 Pause time

We first consider the rekeying rates for the different security algorithms versus pause time where the members move randomly from one group to another group (inter-group). When there is little mobility (high pause time), the members do not get to join and leave as frequently as when there is high mobility (low pause time).

Baseline algorithm and immediate algorithms show the highest rekeying rate as they do not have an EKOL facility. Thus every movement between groups need to be group rekeyed once. As pause time decreases, the rekeying rate increases. The baseline and immediate algorithms perform the worst of the four algorithms. Even at very low mobility rates, the baseline and immediate algorithms perform slightly worse than the other two algorithms. When the pause time is low, the sliding window and delayed rekeying perform best. It is observed that the sliding window performs better than the delayed rekey algorithm at higher mobility (62.5% at 0.01 second pause time and 15% at 0.1 second and 5% at 0.5 second). As mobility decreases, the differences between the two approaches decrease. The rekeying rate shows little difference between the two algorithms after pause time 0.1s.When the pause time is low, the sliding window protocol shows a lower rekeying rate than all the other algorithms due to its ability to handle multiple transfers. When mobility increases, there are lots of nodes joining and leaving between groups of the network. The number of keys on the EKOL could exceed the specified threshold due to many nodes joining and leaving. Thus delayed rekeying needs to stop the data transmission when the EKOL is full and needs to be updated.
A lot of rekeying needs to be done at the beginning when the EKOL is empty. As the sliding window algorithm uses timeout to control the validity of the keys hold in EKOL, not many periodic rekeying updates are necessary when compared with the delayed algorithm. Sliding window is running as a dynamic EKOL and the time-out will make sure no local keys remain valid for more than a fixed period of time and there is no need to stop transmission if nodes arrive or leave the group. Thus the sliding window performs best of the four.

5.2.2 Queue

We next consider security algorithm performance for rekeying rate versus queue. The members move randomly from one group to another inter group. For our simulations, the number of groups range from 5 to 5000. The rekeying rate is at the high side for all four algorithms when the number of groups visited is high. The rekeying rate for baseline algorithm and immediate algorithm is far higher (31% more at around 500 groups visited per node) than the delayed and sliding window.

This scenario could be explained because both baseline algorithm and immediate algorithm lack the EKOL facility. As expected, we observe that the keys held by the EKOL even if the node is outside the group helps to keep the rekeying rate low. Baseline and Immediate algorithms need to rekey multiple groups when the node visits many groups. Comparing the sliding window protocol and delayed protocol, the Sliding window algorithm has a better rekeying because it does not stop the transmission frequently to update the list. The sliding window algorithm does not stop the transmission because the key updates are done with the periodic timeout mechanism. Hence the rekeying rate for sliding window is the lowest among the four algorithms.

5.2.3 Arrival rate

We next consider security algorithm performance for rekeying rate versus arrival rate lambda. The members move randomly from one group to another (inter-group). The arrival rate of each node varies from 0.1/s to 100 /s.

We assume that the arrival time is described by a Poisson distribution. Lambda is a rate per unit time or arrival rate. Figure 7 shows that when arrival rate is on the high side, the rekeying rate tends to be on the high side too. The rekeying rate for immediate algorithm and baseline algorithm is around 50% more than rekeying rate for sliding window algorithm and delayed algorithm. We also notice that the rekeying rates are almost similar in both sliding window algorithm and delayed algorithm when the arrival rate gets higher.

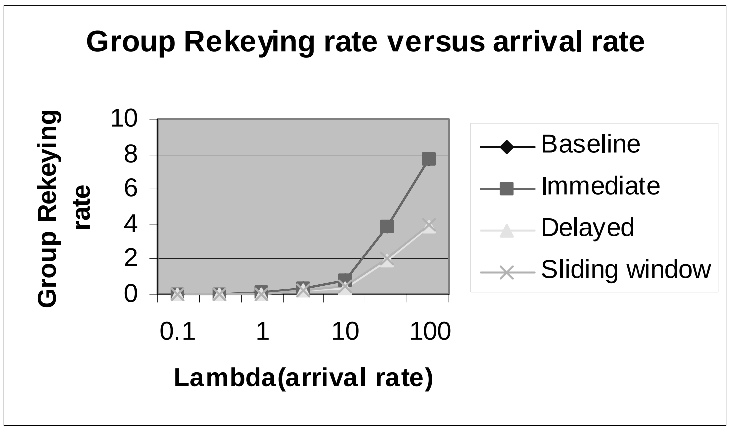


Figure 7: Group rekeying rate versus the arrival rate

The Immediate algorithm interrupts the data transmission when a node leaves and joins the group. If many nodes arrive at the same time, many data transmissions will be interrupted and the rekeying rate will go high. This is because the validity of the key in the group has expired. On the other hand, in the Sliding window the window is running continuously, the add and remove operation is running continuously and it can handle multiple leaves and enters within the same timeout period. The timeout mechanism ensures the validity of keys in the EKOL list. As keys are updated, they are updated in a fixed period of time. According to DeCleene [DeCleene, 2001], the measure of insecurity increases as a function of lambda. Figure 7 shows that both delayed and sliding window are more secure, particularly at high arrival rates as they provide better rekeying rates. A more detailed investigation reveals that the sliding window algorithm shows a slightly lower ratio (5% at 50/s and 100/s) of rekeying rate compared with the delayed algorithm. Therefore the sliding window is more secure and efficient in terms of group rekeying rate than the delayed algorithm.

6 Conclusions

In this paper we have proposed a sliding window mechanism for group multi-communications. The proposed protocol is a simple extension to the DSR protocol. The proposed protocol, the performance of the protocol and the security characteristics of the protocol have been outlined. The proposed scheme improves both routing performance and increases the security of the ad hoc network, particularly at high node mobility. This is significant due to the increasing difficulty in reliable packet delivery and secure communications as mobility increases. Taking the average of the five versions of the sliding window scheme shows an average improvement of total packet delivery of 9.7% over the DSR protocol. This is initial research and further work is needed on determining the optimum network size, degree *d*, time-to-live value etc. Moreover in our simulations we have included route discoveries for single source to single destination as well as single source to multiple destinations (with different number of destinations for different route discoveries). If route discoveries were limited to single source and the same number *n* of destinations for each route discovery, the performance results would be much better. Further performance analysis on using adjustable parameters including receiver data rates, packet loss rate, delays and node geographical location as well as a real life workload study for group model is also needed. The sliding window approach should also be extended to other protocols besides DSR.

The sliding window also improves re-keying performance for secure group communications. The results show that the rekeying rate for the sliding window protocol is lower than the other three rekeying algorithms, especially when mobility is high. The comparison had been done for different parameters, including pause time, Queue (group visited per node) and data packet arrival rate. Table III is a summary of average improvement of rekeying rate using sliding window algorithm over delayed algorithm which performs the best of the other three algorithms.

Parameters	Average group rekeying rate improvement
Pause time	27.5%
Queue	7.50%
arrival rate	5.00%

Table 3: Average improvement of rekeying rate using sliding window scheme

According to DeCleene [DeCleene, 2001], the measure of insecurity increases as a function of the arrival rate. The results show that the rekeying rate of the sliding window scheme is around 50% and 5% lower than rekeying rate of the immediate algorithm and delayed algorithm respectively when the arrival rate is at 100/s. The measure of security is therefore increased using sliding window scheme. Future work would investigate an optimum timeout period such that security is not compromised. If the security system is implemented on a wide logical group in ad hoc networks, where the EKOL may be responsible for holding keys from nodes which may be long distance, then, the entries should be stored in a distributed database management system (DBMS) to decrease the times in storing and retrieving key entries when nodes enter and leave groups.

References

[Abolhasan, 2004] Mehran Abolhasan, Tadeusz Wysocki and Eryk Dutkiewicz, “A review of routing protocols for mobile ad hoc networks”, Ad Hoc Networks, Vol. 2, No. 1 , January 2004, Pages 1-22.

[Bagrodia, 2000] R Bagrodia, M Gerla, J Hsu, W Su, W. S Lee, A performance comparison study of ad hoc wireless multicast protocols. *Proc. of the 19th Annual Joint Conf. of the IEEE Computer and Communications Societies*, Pages(s): 565 –574. March 2000.

[Chan, 2003] Haowen Chan, Adrian Perrig and Dawn Song , “Random Key Pre-distribution Schemes for Sensor Networks”, *Proc.2003 IEEE Symposium on Research in Security and Privacy*. pp 197-213, 2003.

[Das, 2002] S. Das, C. Perkins, E. Royer, Ad hoc on demand distance vector (AODV) routing, Internet Draft, draft-ietf-manet-aodv-11.txt, work in progress, 2002.

[DeCleene, 2001] B DeCleene, L Dondeti, S Griffin, T Hardjono, D Kiwior, J Kurose, D Towsley, S Vasudevan, C Zhang, ”Secure group communications for wireless networks”, *Proceedings of MILCOM*. 2001.

[Eschenauer, 2002] Laurent Eschenauer and Virgil D. Gligor, :A Key Management Scheme for Distributed Sensor Networks*”, Proc. 9th ACM conference on Computer and communications security*, 2002.

[Gerla, 2001] M Gerla, M, S-J Lee and W Su, On-demand multicast routing protocol (ODMRP) for ad hoc networks. *Mobile Network and Application*, 2001

[Gopalsamy, 2002] T. Gopalsamy, M. Singhal, D. Panda, and P. Sadayappen, A Reliable Multicast Algorithm for Mobile Ad Hoc Networks, *in Proc. of Distributed Computing Systems Workshop*, pp. 563-570, Vienna, Austria, July 2002.

[Haas, 1999] Z.J. Hass, R. Pearlman, Zone routing protocol for ad-hoc networks, Internet Draft, draft-ietf-manet-zrp-02.txt, work in progress, 1999.

[Harney, 1999] H. Harney and E. Harder, “Logical Key Hierarchy Protocol,” Internet draft, draft-harney-sparta-lkhp-sec-00.txt, March 1999.

[Johnson, 2002] D. Johnson, D. Maltz, J. Jetcheva, The dynamic source routing protocol for mobile ad hoc networks, Internet Draft, draft-ietf-manet-dsr-07.txt, work in progress, 2002.

[Lazos, 2003] Loukas Lazos, Radha Poovendran, “Energy-Aware Secure Multicast Communication In Ad-Hoc Networks Using Geographic Location Information”, *Proceedings ICASSP* , 2003.

[Lou, 2002] W J Lou and Y G Fang,, Predictive Caching strategy for on-demand routing protocols in wireless Ad hoc networks *Wireless Network, 2002.*

[Marina, 2001] M. K. Marina and S. R. Das, "Performance of Route Caching Strategies in Dynamic Source Routing", *Proceedings of the 2nd Wireless Networking and Mobile Computing (WNMC),* Phoenix, April 2001.

[Murthy, 1995] S. Murthy J.J. Garcia-Luna-Aceves, A routing protocol for packet radio networks, in: *Proceedings of the First Annual ACM International Conference on Mobile Computing and Networking,* Berkeley, CA, 1995, pp. 86–95.

[Perkins, 1994] C.E. Perkins, T.J. Watson, Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers, in: *ACM SIGCOMM’94 Conference on Communications Architectures,* London, UK, 1994.

[Radhakrishnan, 1999] S. Radhakrishnan, N.S.V Rao, G. Racherla, C.N. Sekharan, S.G. Batsell, DST—A routing protocol for ad hoc networks using distributed spanning trees, in: IEEE Wireless Communications and Networking Conference, New Orleans, 1999.

[Rodeh, 2000] O Rodeh, K. Birman, D. Dolev, “Optimized Group Rekey for Group Communication Systems”, *Proceedings of Network and Distributed System Security Symposium* 2000.

[Shu, 2002] L. Shu and D. Poppe, *Assuring Message Delivery in Mobile Ad Hoc Networks with Packet Erasure Recovery*, in *Proc. of Distributed Computing Systems Workshop*, pp. 14-19, Vienna, Austria, July 2002.

[Tang, 2002] K. Tang, K. Obraczka, S.-J. Lee, and M. Gerla, A Reliable, Congestion-Controlled Multicast Transport Protocol in Multimedia Multi-hop Networks, in *Proc. of the 5th International Symposium on Wireless Personal Multimedia Communications*, pp. 252-256, Honolulu, USA, October 2002.

[Wong, 2000] Chung Kei Wong, Mohamed Gouda, Simon S. Lam, “Secure Group Communications Using Key Graphs”, *IEEE/ACM Transactions on Networking,* Vol. 8, No. 1, February 2000.

[Zhang, 2002] C Zhang, B DeCleene, J Kurose, D Towsley, “Comparison of Inter-Area rekeying Algorithms for Secure Wireless Group Communications”, *Performance Evaluation*, Vol. 49, Sept. 2002.