MENU: Multicast Emulation using Netlets and Unicast

Kalaiarul Dharmalingam and Martin Collier
Research Institute for Networks and Communications Engineering (RINCE)
Dublin City University, Republic of Ireland
E-mail: {arul, collierm}@eeng.dcu.ie

Abstract— **High-end networking applications such as Internet TV and software distribution have generated a demand for multicast protocols to be an integral part of the network. This will allow such applications to support data dissemination to large groups of users in a scalable and reliable manner. Existing IP multicast protocols lack these features and also require state storage in the core of the network which can be costly to implement.**

In this paper, we present a new multicast protocol referred to as MENU. MENU realises a scalable and a reliable multicast protocol model by pushing the tree building complexity to the edges of the network, thereby eliminating processing and state storage in the core of the network. The MENU protocol builds multicast support in the network using mobile agent based Active Network services - Netlets, and unicast addresses. The multicast delivery tree in MENU is a two level hierarchical structure where users are partitioned into client communities based on geographical proximity. Each client community in the network is treated as a single virtual destination for traffic from the server. Netlet based services referred to as Hot Spot Delegates (HSDs) are deployed by servers at "hot spots" close to each client community. They function as virtual traffic destinations for the traffic from the server and also act as virtual source nodes for all users in the community. The source node feeds data to these distributed HSDs which in turn forward data to all downstream users through a locally constructed traffic delivery tree. It is shown through simulations that the resulting system provides an efficient means to incrementally build a source customisable secured multicast protocol which is both scalable and reliable. Furthermore, results show that MENU employs minimal processing and reduced state information in networks when compared to existing IP multicast protocols.

I. INTRODUCTION

IP Multicast [1] is a network level service in which routers disseminate multiple copies of datagrams to interested group members. This approach to logically group dispersed receivers offers operational advantages for content and network providers by minimising network resource demands and end-system overheads. Despite extensive research, multicast routing protocols have not been widely deployed in the Internet.

One of the primary reasons that discourages widespread multicast deployment in the Internet is the lack of a scalable protocol model. Existing IP multicast protocols require routers in the core of the network to store per-flow state information and to support per-flow packet forwarding operations. Performing per-flow operations inside the core of the network affects the network scalability. This is because, as the number of simultaneously operating multicast sessions increases, there is a linear increase in state information which leads to increased packet processing delays and memory requirements. Given the amount of data flowing through the core of the network, any protocol which requires the maintenance of considerable state information is likely to prove impractical.

Some of the other major factors that discourage multicast deployment are the lack of : (i) reliable inter-domain multicast routing protocols; (ii) reliable communication support - the existing multicast model supports best-effort service and hence does not support reliable communication which limits the applicability of multicast in the Internet; and (iii) access control - conventional IP multicast protocols allow any node in the network to send/receive data to the group, facilitating flooding attacks. A detailed discussion of these problems can be found in [2].

Recent research efforts [3, 4, 5] have demonstrated a one-to-many abstraction of the basic multicast model that scales better than conventional IP multicast. Such a model is appropriate for large scale applications such as Internet TV, automatic software distribution, etc. The available single source multicast models e.g., [3,4] have been successful in supporting a secured group communication model and in overcoming the Class D address depletion problem. However, they still lack scalability and reliability.

A solution using unicast to build multicast services was presented in [6]. The hard-wired nature of this approach means that the protocol model is non-extensible. Furthermore, each node on the multicast tree is required to maintain state information, which affects scalability. In [7], a combination of ephemeral states and unicast forwarding was employed to build multicast services. In this approach, receivers use a topology-probing mechanism to identify a graft point on the delivery tree and instantiate an active service at that point to duplicate and distribute incoming datagrams. This approach generates additional traffic and state information at intermediate network nodes to support continuous tree optimisation.

In this paper, we present the MENU protocol which is intended to serve large scale single source multicast applications. MENU builds multicast support in the network using mobile agent based Active Network services - Netlets and unicast addresses. Netlets are autonomous, nomadic mobile software components which persist and roam in the network independently, providing predefined network services. The Netlets network uses the mobile agent paradigm to realise an Active Network architecture. Netlet Nodes offer support for execution of Netlet based network services. A more detailed discussion of the Netlets approach and its architecture can be found in [8, 9, 10].

The rest of the article is organised as follows. In section II we describe the MENU protocol concept. Section III describes the idea of Hot Spot Delegates and their deployment and activation mechanisms. In section IV we describe the working of the MENU protocol. In section V we describe the flow-based approach which is used to build the traffic delivery trees in MENU. Furthermore, the supporting architectural design features required for operation of MENU are presented in section VI. In section VII we present simulation results which evaluates the large scale deployment of the MENU protocol and section IX concludes the article.

0-7803-8784-8/04/$20.00 © 2004 IEEE

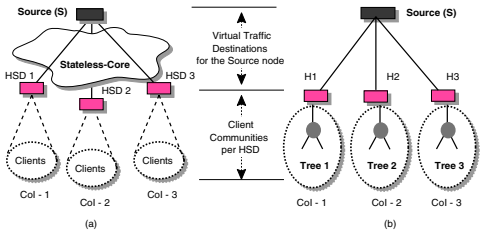

Fig. 1

TWO LEVEL HIERARCHICAL MODEL OF MENU

II. MENU PROTOCOL CONCEPT

The reference architecture shown in Fig. 1-a captures the basic characteristics of the MENU protocol. The multicast delivery tree in MENU is a two level hierarchical structure where users are partitioned into client communities based on geographical proximity, referred to as Communities of Interest(CoI) in this paper (e.g., CoI-1, CoI-2 in Fig.1-a). Each client community in the network is treated as a single virtual destination for traffic from the server. Netlet based services referred to as Hot Spot Delegates (HSDs), are deployed by servers at "hot spots" close to each client community to function as: (i) virtual traffic destinations for the traffic from the server; and (ii) to act as virtual source nodes for all users in the community. The source node, node S in Fig.1-b, feeds data to distributed HSDs (H1,H2 & H3) which in turn forward data to all downstream users through dynamically provisioned traffic delivery trees. Note that MENU realises independent traffic delivery trees from each HSD to all its local receiver nodes (e.g. Tree-1 for CoI-1 and Tree-2 for CoI-2 respectively in Fig. 1-b). MENU follows a data flow-based approach to build such a distributed tree model.

The primary reasons for pushing the tree building complexity to edge networks are as follows. Firstly, the scalability of any protocol which stores state in the core of the network may be poor. Secondly, research findings [6, 11] have shown that close to 70% of nodes in multicast trees have average an fan-out degree of 2 and only function as traffic relay nodes, i.e. performing vanilla packet forwarding as in conventional IP. Furthermore, these relay nodes are found to be located near the source. Therefore there is little demand for packet replication in the core of the network. Finally, since routing nodes present at the network edges are likely to process fewer data flows at lower bit rates than core networks, the expense involved in supporting multicast protocols there will be less.

III. HOT SPOT DELEGATES (HSDS)

1) Hot Spot Nodes: The node on which the HSD service operates is referred to as the Hot Spot Node (HSN). The exact location and the number of "hot spots" present in a network (i.e. with respect to a particular traffic source node) is dictated by the location of the relevant communities of interest in the network, as discussed in section VI. Methods to discover active nodes[1] at hot spot locations in a heterogeneous network environment (i.e. accommodating both legacy and active nodes) such as the Internet can be found in our earlier work [12].

2) HSD Deployment: Note the mobile and autonomous property of service code in the Netlets architecture avoids the need for manual intervention in HSD deployment. A single HSD service is deployed in the network with the address list of nodes requiring service activation. This Netlet then autonomously migrates to each node and installs the service thus avoiding centralised deployment schemes and generating less network traffic [8].

3) HSDs as Virtual Traffic Destinations for Source: The source node, S in Fig.1, maintains an address list of HSDs operating in the network. This address information includes the unicast address of the HSNs and the port on which the HSD receives data from the traffic server. Note that HSDs are not specific to any multicast session. When a server is required to support simultaneous multicast sessions in a network, it informs the HSD of the specific session details.

[1] We refer to those nodes that can be dynamically programmed to host Netlet services as active/Netlet nodes, as in active networking technology.

IV. WORKING OF THE MENU PROTOCOL

A. Anycast Address and Connection Attraction

A multicast session in MENU is identified by a globally unique anycast address [13] (that corresponds to the traffic source e.g., node S in Fig.1) and a source generated port number.

$$multicast\ session =< anycastaddress, portnumber >$$

MENU employs this global anycast address [13] to seamlessly integrate the distributed HSDs in the network with the traditional client-server communication followed in the Internet. The MENU protocol shares this anycast address among the HSDs that act as virtual source nodes and with the traffic source node. For the example in Fig. 1-b, the anycast address is shared among the Source and the HSDs H1, H2 & H3. Thus the distributed HSDs and the server are presented to the rest of the network as a single logical entity. A detailed discussion concerning the activation of HSDs and registration of anycast addresses at Hot Spot Nodes is deferred to section VI.

In MENU, when a user wishes to receive data from a server, the user connects to the server (using the available global anycast address) as in the unicast communication model. Consequently, messages from users that correspond to a server's anycast address are automatically routed to the closest HSD rather than directly to the sender node. If no HSD exists close to a client's location, the messages get automatically routed to the source node itself. HSD services that receive join requests from users, in turn, generate HSD subscription messages to the source for receiving session data. Following this, the source node distributes traffic to the HSDs via unicast connections. Note unicast communication is used here because MENU assumes a stateless core network model, i.e. where storage of flow state entries are not supported.

B. Recursive Packet Replication and Forwarding

The HSDs on receiving the data performs recursive packet replication and forwarding within the network to distribute datagrams to members of the multicast session. To support recursive operation, multicast packets carry unicast destination addresses of immediate downstream branch nodes rather than Class D addresses as in the conventional IP multicast model. Each replicated packet is set with the unicast destination addresses of the downstream receiver member. Note the source address of a multicast packet in MENU is always set to point the actual source node responsible for the session, node S in Fig. 2. However, multicast datagrams carry the address of the node which actually generated it in the IP Source Route Option field.

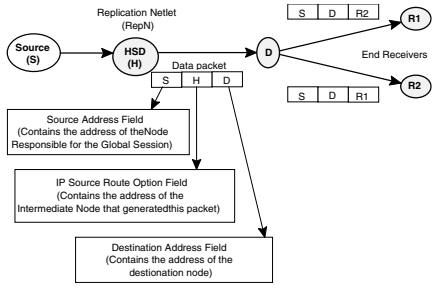

Fig. 2

PACKET REPLICATION AND FORMAT OF DATA PACKETS IN MENU

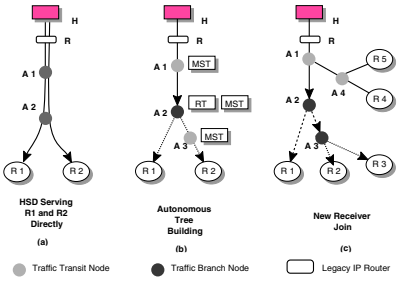

Fig. 3

BUILDING THE TRAFFIC DELIVERY TREE

The inclusion of two source addresses allows a downstream receiver: (i) to know the global session to which the packet belongs; and (ii) to know the local upstream source responsible for packet replication. Packets are recursively replicated until they reach the end user nodes. For example in Fig.2, HSD node, H, addresses session packet to its downstream receiver, node D. Following this, node D performs packet replication and sends two individual copies of the packet to its downstream receivers, R1 and R2.

C. Replication Netlets (RepN)

Replication Netlet services, referred to as RepN, are employed to perform the packet replication and packet forwarding functions in MENU. For example in Fig. 2, RepN service operating at node D performs this function. Additional functions of RepN services operating at on-tree nodes of a multicast delivery tree include:

- recording the list of live multicast sessions that traverse the local node in the Multicast Session Table (MST); the session details include the *global source address of the session* (i.e., the anycast address, S in Fig. 2), *global port number*, *actual source address of the packet (i.e. the address present in IP Source Route field)* and *destination address of the packet*;
- evaluating whether the local node acts acts as a traffic transit or a branch node in the delivery tree;
- if branch node, then maintaining the list of downstream receiver node addresses in the receiver table (RT) for which the local node is the upstream source; and
- listening for join requests that corresponds to any live session present in the MST.

V. FLOW PROGRAMMED TREE CONSTRUCTION

For the purpose of illustration, we describe the tree construction procedure followed in MENU using Fig. 3. First, R1 and R2 subscribe for receiving multicast session data from source S. Due to the use of an anycast address, the session request is routed to the closest HSD, node H. H on receiving session data from the source, replicates and forwards datagrams to both the downstream receivers. The data packets as they travel towards the destination trigger the tree construction process.

1) Flow-based Activation of RepN Services: Data packets in MENU carry the name of the Netlet service that will process them at intermediate network nodes, i.e., RepN in this case. Note such a feature allows custom processing of multicast packets within the network. On arrival of a multicast packet at a network node, the RepN service records the multicast session details in its MST (see section IV-C). Following this, it performs conventional packet forwarding as in IP. For example in Fig. 3-a, when individual data packets are sent from H to R1 and R2, RepN services at A1 & A2 record session details in their corresponding MSTs. Hence, routers on the delivery path automatically become members of the MENU tree.

2) RepN Service at An On-Tree Node: When an RepN service at an on-tree node records multiple flows that belong to the same session in its MST (in Fig. 3-b, both A1 and A2 have two copies of the same flow from H), it initiates a simple procedure to evaluate whether the local node is either a transit or a branch point in the MENU tree. The evaluation procedure is as follows. The RepN service using the local routing table entries checks whether all such flows have a common next hop node;

- if true, the RepN service knows that it is a transit node for the traffic, and does not perform any further evaluation;
- if false, the RepN service knows that it has to act as a branch in the MENU tree and works as described below.

3) RepN at a Branch Node: The RepN service creates a Receiver Table (RT) and adds the addresses of all downstream receivers for which it is the optimal branch point. It then requests the actual source node (as recorded in the MST) currently serving these receivers to handover the session. Following this request, the actual source node adds this requesting node as a downstream member in its RT. Furthermore, the source node hand overs the set of receivers it was handling to this new optimal branch node. For the example in Fig. 3-b, the RepN at A2 on identifying itself as a branch node works as follows. This RepN service by consulting the MST identifies H as the actual source for the session. Next, A2 requests branch node status from H. Additionally, it advises H that it is the optimal branch point for downstream receivers R1 and R2. Next, H adds A2 as its immediate downstream member in its RT and performs handover of R1 and R2 to A2.

Protocol Dynamics

1) Joining an existing session: When a join request from a user to a live multicast session traverses an on-tree node, the RepN service captures the request and then works as follows: (i) if the node is a branch point in the tree, it adds the user as a downstream member in its Receiver Table (RT); or (ii) if the node is currently acting as a transit point for traffic, the RepN service recognises that it has to act as an optimal branch point from now on. It then follows the procedure described above to claim branch node status. For example in Fig. 3-b, A3 is a transit node for the request from R3. After receiving the user join request from R3, A3 in turn requests for a change in status from transit to branch from the actual source of the session, A2. Furthermore, it requests A2 for session handover of R2. This is because the new optimal branch point for R2 is A3.

2) Leaving a session: In MENU, it is assumed that periodic heartbeat messages are issued by users to their corresponding actual source nodes in order to receive session data continuously. Thus, when a user node wants to leave the session, it simply stops sending these heartbeat messages. After an appropriate timeout interval, the source removes the user from its RT. A branch node in MENU changes to a transit node only when the number of downstream receivers for that particular node falls below two. For the example in Fig. 3-d, when R2 leaves, A3 has only a single receiver R3. Following this event, the RepN service at A3 hands over R3, to the upstream source from which it has been receiving data for the session i.e., A2.

3) Sub-optimal Branches for Transient Periods: At times, sub-optimal branches may exist due to race conditions in user joins. For example in Fig. 3-c, when R4 and R5 issue join requests in immediate successions, the branch node at A1 recognises itself as the upstream source for those receivers. However, the optimal branch point is A4. This sub-optimal structure arises because A4 has not been added as a member in the MENU tree. A4 will become a member in the tree only when multicast datagrams traverse it. Hence, during this intermediate transient periods sub-optimal branches may exist. However, once multicast data packets flow across A4, this node joins the tree automatically. It next recognises itself as a branch point and performs tree optimisation as described in section V-.3. Note that during this transient period, receivers may receive redundant data. By employing sequence numbers within MENU packets, receivers will be able to ignore such redundant packets.

VI. Discussion

1) Communities of Interest: A deployment scheme is required for distributing HSD services within the network. Analysis of access logs of various servers have shown the existence of *communities of interest* in the Internet [14,15,16]. These are groups of clients which are responsible for generating a high proportion of the workload on servers and which are geographically close or under common administrative control. Traffic sources should deploy HSD services close to such communities.

In [15], a network-aware method based on prefixes and netmask information gathered from Border Gateway Protocol (BGP) routing table snapshots was used to identify client clusters (referred as communities of interest in this paper) in the Internet. This technique gave good performance even when used with historical snapshot data.

The results from [15] based on globally collected server logs show that 90% of communities have 100% of their clients topologically close to each other. It was also reported that around 5% of communities accounted for the majority of the clients and for generating a high percentage of the workload on the web server. This confirms earlier studies [16] that claim the existence of Zipf-like distributions in a variety of web measurements. By being able to locate communities of interest, servers will be able to serve the majority of the client population using MENU services.

Recall that when no HSD service exists within a domain, the anycast packets are routed towards the traffic source which shares the same anycast address with HSD services. Since there are relatively few clients outside the communities of interest, traditional unicast can be used to serve them. This does not represent a major burden on the traffic source. Note that the inter-domain routing can be implemented in a scalable manner using the method of Global IP Anycast (GIA) [13].

2) HSD Activation: When a HSD service is deployed at a network node, this service requests the local node: (i) to register for receiving client requests that correspond to the anycast address for which the HSD holds the permission; and (ii) to advertise routes for the anycast address. The concept of virtual host and interfaces used by IP aliasing can be used to register HSD services at network nodes. IP Aliasing is simply a mechanism that enables a single physical or virtual network port to assume responsibility for more than one IP address. By using this feature a network node will be able to support multiple HSD services simultaneously. New routes to anycast addresses can be advertised as part of normal routing table updates. Note that HSDs are not specific to any multicast session. When a server is required to support simultaneous multicast sessions in a network, it informs the HSD of the specific session details.

3) Scalability of Unicast Routing Protocol for Anycast Addresses: HSD services employ global anycast addresses to seamlessly integrate the dynamically deployed service points with the traditional client-server based communication model followed in the Internet. When HSD services are deployed within stub networks they behave as local anycast groups to the corresponding stub domain. Due to this specific nature of the Netlets approach, conventional intra-domain routing protocols will be sufficient to route packets destined to anycast receivers local to the domain. For example, distance-vector algorithms, such as RIP inherently provide support for anycast service [17]. Employing unicast protocols for anycast services causes each HSD present within a stub network to take up an entry in the internal routing table. However, this approach is scalable because: (i) the number of HSDs within a network is driven by user demand local to that domain; and (ii) routers in the edge networks have small routing tables and fewer data flows to process at lower bit rates than core networks.

VII. Evaluating Large Scale Deployment of MENU

Active nodes are required to be present in the Internet, in order to host the Replication Netlet (RepN) services, which provide packet duplication support in the MENU protocol. However, due to the large scale nature and heterogeneity of the Internet, such support can only be integrated gradually. Under such circumstances, MENU like protocols will go through phases of partial deployment to ubiquitous availability. Here, we study through simulations the effect of incremental deployment of active node support in the Internet on MENU.

We evaluate this based on: (i) the gain achieved (e.g. reduction in bandwidth) when using MENU based multicast communication services over unicast to support group communication in the Internet; (ii) the packet redundancy level, referred to as the link stress [18], experienced by the network nodes when working with partial deployment of active nodes in the network; and (iii) the forwarding state saving achieved by MENU in comparison to existing multicast protocols.

This analysis was performed using core-stub network topologies generated using the GT-ITM [19] package. The topologies for the study have 20 nodes per stub domain and 10 nodes per core domain. The total number of router nodes present in the generated topologies are 500. Furthermore, 20 user nodes are added at random to each stub domain in the network. Note that at most a single stub router had only two

receivers assigned. For the purpose of illustration, the topology we used for testing the effects of mean hop count (12.5) against multicast gain is presented in Fig. 4. For each simulation cycle, the traffic source was selected randomly from one of the stub domains. We used different randomisation seeds during each simulation cycle for assigning network routers as active. We present the results averaged across 25 simulations.

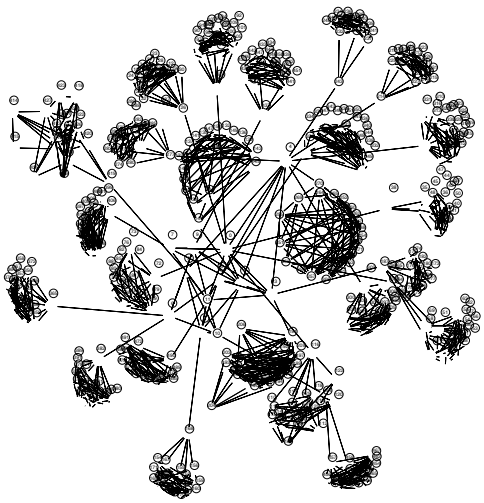

Fig. 4
NETWORK TOPOLOGY USED FOR SIMULATION

Incremental Deployment: Efficiency of employing MENU based multicast communication over unicast for supporting group communication was evaluated using the gain metric defined in [20]. This metric compares the total number of links traversed by multicast packets and unicast packets over a given topology. Note each of the link is referred to as a hop in the path of a single unicast or multicast packet. The multicast gain is defined as [20]:

$$\delta = 1 - \frac{multicasthops}{unicasthops}$$

δ represents the percentage gain in efficiency of MENU in relation to conventional unicast. For δ approaching zero, multicast offers no gain over unicast communication. As δ increases (to a maximum value of $\delta = 1$) multicast communication offers bandwidth savings over unicast.

Here, we studied the multicast gain when stub network domains hosted active nodes i.e. where there were stateful stub domains and stateless core domains. The effect of incremental deployment on MENU was assessed by varying the number of active nodes available within the stub networks. In the experiment, the ratio of active nodes available in each stub network is varied from 0% to 100% in steps of 10. The traffic delivery tree is constructed as described in section V. For each deployment ratio, the number of multicast hops to unicast hops when serving all the receivers in the network was recorded. This experiment was repeated for different mean path lengths (between the source and receiver nodes). This allowed us to evaluate the impact the HSD created on the multicast gain.

Fig. 5 shows the results obtained from this analysis. The case of 1 active node per stub domain denotes the existence of a HSD alone for each community of interest. In the MENU

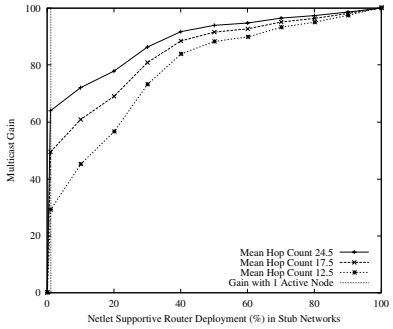

Fig. 5
MULTICAST GAIN WITH ACTIVE STUB NETWORKS

model this will result in individual unicast connections being setup from source to each HSD and from each HSD to all its end users. With only the HSD being present, the gain varied from 30% to 73% for mean path lengths from 12.5 to 23.5 respectively. This is due to fact that all receivers from a community were served from the local HSD, which in turn communicated with the traffic source. The reason for the increase in gain with path length is that the HSD was closer to the receivers than to the source. Note that even with a deployment ratio of only 40% of active nodes per stub domain, the average multicast gain was close to 75%. Overall, it can be concluded that ISPs wishing to provide MENU based multicast services can deploy active nodes and attain significant gain over unicast without considering service availability at other parts in the network.

Link Stress: Link stress [18] indicates the number of times that a semantically identical multicast packet traverses a given link. Examining stress will give an estimate on the level of packet redundancy experienced by network links. Furthermore, using this we will be able to assess the optimum level of active node support required to achieve minimum packet redundancy. Note that, with traditional IP Multicast the stress value never exceeds 1 when all nodes in the network support multicast, i.e. the ideal case.

We quantified the packet redundancy on a per stub domain basis i.e. corresponding to a single ISP. This will allow ISPs to compare the multicast gain against packet redundancy for various percentages of active node deployment. In this analysis, we used a 25 node network which connects 50 receivers to the traffic source. The traffic delivery tree is constructed using the approach described in section V. For various levels of active node deployment, the number of duplicate packets traversing each link in the network was recorded. The experiment was repeated for networks with different connectivity levels. Fig. 6 shows the change in link stress with increasing number of active nodes in a network. On average, with close to 40% of active nodes, the stress factor reduced by 50% i.e. from 3.5 to 1.72. This is because, as the network connectivity improved there were many optimal shortest path routes available from every node in the network to the HSD which was the virtual source for that network.

Forwarding State Saving: In MENU, forwarding states are only stored at branch points of the traffic delivery tree. The state saving achieved by not storing at non-branch nodes of the tree, reduces packet processing delays and memory requirements at intermediate network nodes. In this experiment we

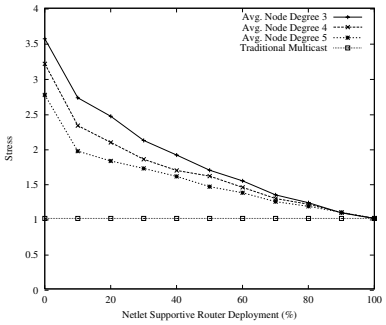

Fig. 6
LINK STRESS

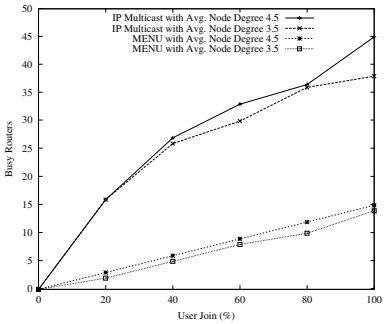

Fig. 7
REDUCTION IN FORWARDING STATE

evaluated the forwarding state saving achieved per established session. The traffic delivery tree is constructed as described in section V. For different levels of user joins to the session, I record the number of routers that had to store packet forwarding state information.

Fig. 7 shows the state saving achieved by MENU when compared to traditional IP multicast protocols. It is evident that only around 30% of routers present in the delivery tree are required to store session details. This allows a reduction in state storage which inherently reduces packet processing delays, avoids complex packet handling software modules and minimises memory requirements. This reduction improves the scalability of the protocol. Note that, this result is in agreement with the results presented in [6, 11], which reports that close to 70% of the nodes in a tree are non-branch nodes and only function as traffic relay nodes.

VIII. KEY BENEFITS OF MENU

- ◇ **Minimises processing requirements and the amount of state information in networks:** The MENU model does not store forwarding states in the core of the network. Furthermore, in stub domains, forwarding state entries are only maintained at branch points of the traffic delivery tree.
- ◇ **Achieves scalability through route aggregation:** By using the unicast addresses, the MENU model supports address aggregation with hierarchical address allocation as in the unicast communication model.
- ◇ **Avoids the need for new Inter-Domain multicast routing protocols:** MENU uses the inter-domain unicast routing protocols for session establishment purposes.
- ◇ **Supports incremental deployment:** MENU allows gradual and transparent deployment of the protocol in the network without penalising or disrupting existing network services.
- ◇ **Minimal Error Recovery Delay:** Error recovery processes that are triggered due to packet losses within the network can be minimised by providing data caching support at network nodes present in the stub domain.

IX. CONCLUSIONS

In this paper we presented a new multicast protocol referred to as MENU. MENU builds a scalable multicast protocol model by pushing the tree building complexity to the edge network, thereby eliminating processing and state storage in the core of the network. By employing existing unicast protocols, MENU achieves is cost-effective to deploy in large scale network environments. Furthermore, MENU also provides reliable multicast communication services by supporting data caching within the network. Simulation results show that with only around 40% of routers being active per stub domain and for a mean path length of 23.5, MENU achieves a multicast gain close to 70% and packet redundancy of only 1.72. Furthermore, MENU operates with a 70% state saving, thus overcoming the scalability problems faced by conventional IP multicast protocols. Overall, MENU provides an efficient means to incrementally build a source customisable secured multicast protocol which is both scalable and reliable.

REFERENCES

[1] Deering S and et al. Multicast Routing in Datagram Internetworks and Extended LANS. *ACM Transactions on Computer Systems*, May 1990.
[2] Christophe Diot and et al. Deployment issues for the IP multicast service and architecture. *IEEE Network*, Vol. 14, 2000.
[3] Cheriton D.R Holbrook H.W. IP Multicast channels: EXPRESS support for large-scale single source applications. *SIGCOMM*, 1999.
[4] K. Yano and S. McCanne. The Breadcrumb Forwarding Service. *ACM SIGCOMM Computer Communication Review*, April, 2000.
[5] Bhaskar N. Source-specific protocol independent multicast. *Internet-draft*, pages draft–bhaskar–pim–ss–00.txt, 2000.
[6] Ion Stoica and T. S. Eugene Ng and Hui Zhang. REUNITE: A recursive unicast approach to multicast. *INFOCOM*, 2000.
[7] Su Wen and et al. Building multicast services from unicast forwarding and ephemeral state. *Elsevier Journal of Computer Networks*, 38, 2002.
[8] Kalaiarul Dharmalingam. Network Support for Multimedia Applications using the Netlets Architecture. *Ph.D Thesis, Dublin City University*, February, 2004.
[9] Martin Collier. Netlets: The Future of Networking? *IEEE OPENARCH*, April 1998.
[10] Kalaiarul Dharmalingam and Martin Collier. RSVP Reservation Gaps: Problems and Solutions. *IEEE ICC*, May 2003.
[11] J. Pansiot and D. Grad. On routes and multicast trees in the Internet. *ACM Computer Communication Review, vol. 28, no. 1*, June 1998.
[12] Kalaiarul Dharmalingam and Martin Collier. Scalable Client-side Server Selection using Netlets. *IEEE OPENARCH*, April 2003.
[13] Dina Katabi and John Wroclawski. A framework for scalable global IP-anycast (GIA). *SIGCOMM*, 2000.
[14] K. Almeroth and et al. Multicast group behavior in the internet's multicast backbone (mbone). *IEEE Communications Magazine*, June 1997.
[15] B. Krishnamurthy and J. Wang. On Network-aware Clustering of Web Clients. *In Proceedings of ACM SIGCOMM*, 2000.
[16] Lee Breslau and et al. Web Caching and Zipf-like Distributions: Evidence and Implications. *INFOCOM*, 1999.
[17] R. Engel et al. Using IP Anycast for Load Distribution and Server Location. *In Proc. Third Global Internet Conference*, 1998.
[18] Y.-H. Chu, S. G. Rao, and H. Zhang. A Case for End System Multicast. *ACM SIGMETRICS*, June 2000.
[19] Ellen W. Zegura et. al. How to model an internetwork. *IEEE Infocom*, March 1996.
[20] Chalmers, R. and Almeroth, K. Developing a Multicast Metric. *Proceedings of IEEE Global Internet Conference, IEEE GLOBECOM*, 2000.