

I-DG: A Secure Protocol for Disseminating Data to Subscribers via IP Multicast*

Aslihan Celik, JoAnne Holliday, Bindumadhavi Ramavarjula
Santa Clara University, Santa Clara, California, USA

Abstract

This paper proposes the IP Multicast-enabled Drop Groups (I-DG) protocol as a solution to the problem of efficiently and securely disseminating information to a large number of subscribers in the Internet. The I-DG protocol uses an appropriate IP multicasting protocol as the base and addresses the problems introduced by large scale data dissemination. I-DG provides dynamic subscription to multicast data, efficient secure subscription in IP multicasting, and proposes a solution to content organization in broadcasts.

Keywords: Content distribution, security, multicast, Internet delivery, simulation

1. Introduction

A popular information delivery mechanism is *push-based* data dissemination. A push-based system delivers the data to consumers without an explicit request from the consumer. This is particularly important for the case where the information is dynamic and the clients would like to be apprised of the most recent values. Instead of querying the information provider continuously, the provider “pushes” the information to the client. Furthermore, if a network channel is shared by the clients (such as a wireless network where the clients listen to the same frequency), then the information provider can simply send all the data items to the network in the form of a broadcast message [13]. This broadcast is nothing but a collection of data items. All the clients listen to the same broadcast and download the data items that they are interested in. Once a broadcast is over, the provider prepares a new one (possibly with updated values) and sends it.

Push-based data broadcasting has been studied extensively in the context of wireless networks [13, 7]). We argue, however, that data broadcasting techniques can also benefit data dissemination over the Internet. In particular,

the multicasting technology for the Internet provides the equivalent of a common frequency of a wireless network.

The DG protocol that we developed for data broadcasting [7], works efficiently in environments where clients share a channel. It provides a subscription-based solution for accessing data disseminated by an *information server*. One promising area of DG’s application is the Internet.

In the Internet, a specific push-based information dissemination paradigm currently in vogue is IP multicasting. IP multicasting aggregates the clients (i.e., hosts) that are interested in the same data into a group, then assigns a unique multicast IP address for that group. The server sends the information to that address and clients listen to the address to download it. The variability in delivery time under IP multicast is usually low and is dependent on the network. Low variability makes multicast especially desirable for time-critical data such as financial information. IP multicasting has mainly served the broadcasting of video and audio, video conferencing, dissemination of financial information such as stock quotes and reports, news feeds, software updates and database replication. For example, Yahoo’s product *financevision* (see <http://financevision.yahoo.com>) uses streaming webcast to provide users with live financial information. Another example is *isyndicate.com* that multicasts news and reports to 170,000 web sites.

A closer look at multicasting applications reveals that this field is far from mature, especially in the subscription based secure access area. In existing multicasting applications, information providers allocate a number of “channels” for data items, and the clients subscribe to these channels. Usually, the client is interested in just a subset of the content served in a channel. Yet, it has to pay for the entire channel. One solution is to decrease the content included in each channel, but this will increase the number of channels, and may require a client to subscribe to multiple channels. One extreme is to allocate an individual channel for each data item. However, this may complicate client access, since a client may have to listen to multiple channels simultaneously. A better solution would be to limit the number of channels, and allow clients access on a per item basis. This scheme necessitates a security layer to allow a client access to the items that it subscribed, and nothing more. Such

*This research is partially supported by an IBM Faculty Research Grant of Santa Clara University.

a solution is provided by the Drop Groups (DG) approach proposed in the data broadcasting context.

In this paper, we show that the DG protocol is applicable in the IP multicasting framework, and develop the IP Multicast-enabled DG (I-DG) protocol. In particular, we establish that the I-DG protocol can be implemented on top of an existing IP multicasting protocol as long as that protocol fulfills the requirements of the I-DG method. We also show the preliminary results of our simulations.

The rest of the paper is organized as follows: first, we review the literature in Section 2, then we outline the DG protocol in Section 3. In Section 4, we develop the I-DG protocol and propose further improvements. We present our simulation model and results in Section 5. Finally, we conclude in Section 6.

2. Related Work and Contribution

The first theme of this paper is multicasting on the Internet, namely, the IP multicasting. The second theme is data broadcasting. Finally, the third theme is secure access from broadcasts with a particular emphasis on client subscription.

The Internet Security Association and Key Management Protocol (ISAKMP) [15] defines the procedures for authenticating a communicating peer, creation and management of Security Associations, key generation techniques, and threat mitigation (e.g. denial of service and replay attacks). ISAKMP is a flexible method but its implementation is deemed complex. There is ongoing research on multicast security following the publication of [11]. In that paper, the three building blocks of secure IP multicast are identified as follows. (1) *Secure multicast data handling* covers problems concerning the security-related treatments of multicast data by the sender and the receiver. Multicast group data needs to be encrypted using a group key, and the sender must be authenticated. (2) *Group key management* involves generating, assigning and distributing keys to the members of a multicast group. [12] defines Group Security Associations (GSA) for the support of group key management in IP multicast security. The final building block of secure IP multicast is (3) *Multicast security policies*. These policies must be designed considering the fact that policies may be expressed in different ways, as they may exist at different levels in a given multicast security architecture and that they may be interpreted differently according to the context in which they are specified and implemented.

There is additional research on multicast groups and secure multicasting (e.g., [3]), however these mainly consider multicast routing issues.

The other theme of work related to this paper is data management in broadcast context which is of significant recent interest. Most of the work performed in this area has dealt with the following questions: (a) how does a server

determine *what* to broadcast? and (b) how do clients efficiently retrieve data from a broadcast? A number of papers have appeared on this subject, including [13, 8, 9, 1]. We assume a basic broadcast structure that is in agreement with some of this earlier work.

The third theme of this paper, subscription-based access is researched in the popular publish-subscribe context. In the publish-subscribe paradigm, information providers publish data items (events), and clients subscribe to categories of events. There are two major systems: (a) subject-based, and (b) content-based. In subject-based subscription an event is classified and labeled by the publisher as belonging to one of a fixed set of groups (or channels) [2, 17]. Here, each item is referred to by only one attribute (an id or a name). The subscribers are grouped based on subscribed items. Each group that has subscribers for a particular item receive the published events. The content-based systems, however, support an event schema that defines the type of information contained in each event [19, 16]. With this schema, subscribers can filter out unwanted information by applying predicates against the schema. Therefore, only the relevant information will be sent to each group of subscribers. However, practical applications must limit the number of groups since potentially each client has a unique subscription predicate set.

Contributions of this paper are three-fold:

1. **Dynamic subscription to multicast data:** Current multicast applications do not accommodate dynamic subscription as described above, i.e., clients tune in to multicast in an all-or nothing access mode. In general, once a client subscribes to the service, it is considered subscribed for the duration of the multicast. Clients can leave and join the multicast freely, but they are billed for the entire multicast (if there is a charge). Therefore, it is usually not possible to charge a client in an access-based manner. We propose to provide dynamic subscription that allows access-based charging using an encryption mechanism.

2. **Efficient secure subscription in IP multicasting:** The I-DG protocol developed in this paper encrypts the data using group keys. Although group keys introduce an information burden, the grouping criterion in I-DG minimizes the overhead in the multicast. Therefore, I-DG conserves network resources, addressing an important limitation in the widespread adoption of IP multicasting.

3. **Content organization in IP multicasting:** The protocols described in this paper organize the data items in the broadcast by introducing an index structure that has tuning information to the data items. The tuning information help clients find the right data items without having to listen to the multicast continuously. The index can further help the delivery of the multicast by limiting the distribution of a



Figure 1. A high-level view of a broadcast

certain data item to its subscribers only.

3. The Subscription-based data broadcasting problem

We are considering an information commerce environment in the Internet, where clients are interested in most recent values of certain items, and subscribe to these items for a period of time. This subscription entitles the client to access a data object for a specific period of time. Once the contracted period for a subscription is over, the subscription is considered to have *expired* and the client cannot access the data item any longer without resubscribing. Therefore, an encryption mechanism should be used, and only the rightful subscribers should be given the means to decrypt the communication.

3.1. Current Solutions

In this environment, there are two separate servers, the Subscription Server and the Broadcast Server. When a client wishes to access the service, it first contacts the Subscription Server (using regular http or shttp) that handles the subscription requests. The Subscription Server communicates all the requests to the Broadcast Server (over the Internet or a private network), and the Broadcast Server sends off a “broadcast” that includes the results of the queries. A broadcast consists of a sequence of data items preceded (or interleaved) by an index that acts as a table of contents for the items included in the broadcast. A simple broadcast structure is shown in Figure 1.

After exchanging information with the Subscription Server, the client listens to the broadcast and keeps downloads its item of interest until its subscription expires.

The two servers need to communicate with each other mainly for exchanging information specific to data items. The client needs to contact the Subscription Server, but has no interaction with the Broadcast Server except for listening to the broadcast.

The commonly used techniques that implement encryption for handling the subscription in this environment ([14, 5, 4]), such as the Unicast, the Group Multicast, and the Session Multicast protocols have various limitations.

- **Unicast:** In the Unicast model in IP is analogous to the Point-to-point (PTP) model; the information source contacts a host every time it sends data. In Unicast, the

aforementioned Broadcast server acts as an Information server (IS) only. The IS sends a copy of the data item for each subscriber individually. This approach is clearly inefficient since the same information is repeated for each client. For example, if the server provides a live video feed of a news event to a set of subscribers, it must send the video feed sequentially for each subscriber. A typical data rate of 4 megabits per second requires a bandwidth of 4 terabits per second if there are 1 million subscribers. Therefore, applications that require servicing a very large number of hosts may suffer from scalability problems.

- **Group Multicast:** IP multicast proposes to aggregate the hosts that are interested in the same data into a group, then assigns a unique multicast IP address for that group. The server sends the information only once to that IP address. A host listens to the multicast address for the group that it belongs to, and filters out unwanted multicasts at the data link layer. In IP Multicasting, the Group multicast method forms a group of clients for a set of data items and assigns an encryption key for the group [16]. It then sends one copy of the data item to all the members of the group. The network is in charge of multiplying the message such that all group members can receive it. When the group changes, the key must also be changed and the new key must be given to the remaining subscribers.
- **Session Multicast:** The Session multicast approach is modeled after the Session Key approach in the context of data broadcasting [7]. This approach considers each broadcast as a session and changes the data item keys at each broadcast period. Therefore, at each broadcast, the new key must be given to all of the subscribers of each data item.

Although key generation is rather fast and cheap, it is costly to distribute new keys to the clients. Each group member must be individually contacted and be given the new key. The key information may also be included in the broadcast as overhead, but this will increase the size of the broadcast. Furthermore, the size of this overhead is dependent of the number of subscribers or groups. Therefore, both the Group multicast and the Session multicast techniques suffer considerable scalability problems as demonstrated in [7].

The Drop Groups (DG) protocol, presented in [7], addresses this scalability problem and proposes a new grouping criterion to minimize the overhead associated with distributing the new keys. In the following, we outline the DG protocol and present an extension to it for deployment in the Internet.

3.2. The Drop Groups (DG) Protocol

In this section, we describe the Drop Groups (DG) protocol. For a more expanded treatment see [7].

The DG protocol assigns each client to predetermined DG groups, and assigns each DG group a group key that remains valid until the group changes. Note that the use of the term “group” is different from the groups under IP Multicasting. In order to avoid confusion, we use the term “DG group” to denote logical groupings of clients under DG, and the term “multicast group” to denote a group that listens to the same IP address. This is similar to the Group Key approach in that keys are assigned based on groupings of the clients. The difference, however, is in the way the groups are formed. In the Group Key protocol, subscribers of an item usually form a multicast group, and are given a group key valid until there are *drops* (i.e., subscription expiration) from the group. When a drop occurs, a new group key must be generated and distributed to remaining subscribers so that the dropped subscribers don’t have access to new values of the data item. To avoid having to generate a new group key, DG further divides the subscribers of each data item into sub-groups using an additional criterion. That criterion is the *time to drop*, which is simply the amount of time until a client’s subscription for a data item expires. Therefore, two subscribers, A and B of data item j are in the same DG group if and only if their subscription for j expires at the same time. In order to achieve this sort of grouping, subscription expirations are bunched together at discrete epochs. Since all subscribers in the same DG group will be dropped at the same time, it will never be necessary to issue a new group key and distribute it to the clients. It will be sufficient to generate a new key for the data item, and distribute it to the remaining DG groups.

DG divides the time continuum into subscription epochs such that subscription expirations are only scheduled to happen at the end of an epoch. For example, if the epoch length is 1 hour, a client must subscribe for a discrete number of epochs. There is a maximum on the number of epochs that a client can subscribe, and is called the *horizon*, and is denoted by H . Let d be the number of data items. At the end of each subscription epoch, d groups will be dropped, and d groups will be added, one group per each data item. Essentially, this bounds the number of groups and the number of keys to distribute to the clients.

The DG protocol was developed as a generic method intended for use in a shared network, such as a LAN, a mobile network, or the Internet. However, it needs to be fine-tuned to accommodate the requirements of each individual environment. In the following, we will augment DG to work efficiently in the Internet under the framework of IP multicasting.

4. I-DG: IP Multicast-enabled Drop Groups

In this section, we develop the I-DG protocol by augmenting the DG protocol for use in the IP Multicast environment. The organization of this section is as follows: (a) In Section 4.1, we propose an appropriate broadcast organization for I-DG, and (b) in Section 4.2, we discuss the base IP multicasting protocol upon which I-DG will be overlaid.

I-DG, as well as DG, is designed for use in a shared network. In the I-DG framework, upon subscribing to its items of interest, a client listens to the same broadcast as other clients¹.

As in DG, clients are responsible for tuning in to the information they requested and do not interact with the broadcast server. Recall that the subscription server keeps track of subscription expirations and notifies the broadcast server when there are no groups for a particular epoch for a particular item. If that is the case, the broadcast server does not include the key information for that epoch of that item.

4.1. Broadcast organization in I-DG

The broadcast organization in I-DG is similar to DG except for refinements introduced by the multicasting application. Since the Internet is designed for transmitting packets of data, the broadcast prepared at the Broadcast Server must be packetized. Basically, each broadcast should be cut up into smaller pieces and put into packets. Each packet should start with a multicast packet header that identifies multicast information including the sender and the recipient multicast group. When clients receive the packets, they will strip off the packet header, order the packets, and reconstruct the broadcast. Figure 2 shows the organization of the broadcast under IP multicast.

There are two options for packetizing the broadcast: using (1) fixed-size packets, or (2) variable-size packets. For the network, handling fixed-size packets is slightly easier than variable size packets. However, fixing the size of the packets complicates the multicasting application.

4.2. Selecting an IP multicast protocol for DG

The I-DG approach is more suitable to use with IP multicast protocols that employ dynamic registration such as IGMP [6]. Under such a protocol, the client has to join the specific multicast group to which the I-DG broadcast is sent. The authentication and subscription of the client can be performed before or after joining the multicast group, but note that without subscription, the client will not be able to decrypt the broadcast. Effectively, everybody who joins the

¹In the IP literature, the term “broadcast” refers to flooding the network. We continue using the term “broadcast” to refer to the “structure” of the information that is being multicast.

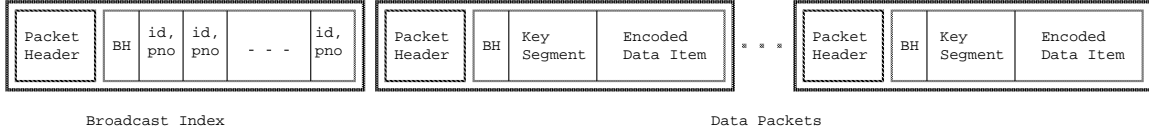


Figure 2. Detailed view of broadcast structure in I-DG

multicast group will receive the broadcast, but only the subscribers will get the decryption keys. Upon the expiration of the subscription, a client may remain in the multicast group and receive the broadcast. But, as the keys will be changed, the client will no longer be able to decrypt the items. In order to leave the multicast group, the client may have to send an explicit “leave” message depending on the multicast protocol used. This message is processed at the router closest to the client and will not be forwarded to the subscription server.

Depending on the network where I-DG is employed, both sparse and dense mode algorithms can be used. If the application multicasts data to a LAN, and almost all hosts are in the multicast group, a dense mode algorithm that is designed to reach a high number of clients attached to the same node is preferable. Otherwise, a sparse mode algorithm that would yield a high data rate and low latency such as PIM-SM [10] should be used. In general, the data broadcasting application on the Internet may be treated as a sparse network.

A fundamental issue is the reliability of the service. I-DG is designed not to receive any feedback (negative or positive acknowledgements) from the clients. After the subscription process, a client listens to the broadcast server, and does not contact the server for missed information. Should it miss an item of interest in one broadcast, the client simply waits for the next broadcast to download that item. Therefore, a multicast protocol such as RMDP [18] that has relatively low reliability and thus low overhead would be adequate. It follows that, a best effort delivery mechanism is more suitable for DG.

The choice of the data rate under I-DG should consider the average user’s equipment. For example, if the target customer is mobile, then the data rate should conform to wireless Internet access capability. We propose to divide the service into two separate multicast groups to accommodate wireless and wired access.

We have shown that there is a range of IP multicast protocols that can employ I-DG. However, extensive testing must be performed to see which protocol performs better under what condition. We will test I-DG with different multicast protocols in Section 5.

5. Simulation

To study the efficacy of our protocol, we ran detailed simulations of I-DG. In this section, we explain the simulation model and the performance parameters. We then present and discuss the results of the simulation.

5.1. The Simulation Model

Our simulation was run on ns2 - a C++ -based network simulator that uses Tcl scripts as a front-end.

We designed the computer model to simulate I-DG in a wired environment using several multicast protocols and the point-to-point (PTP) protocol. The ultimate objective of the simulation runs was to determine which protocol gave us the best performance with the PTP being the benchmark. For the PTP simulation, we used a static routing configuration and Dijkstra’s algorithm for route computations. In the multicasting scenario, we ran simulations using multicasting mechanisms that ns2 provided which closely resembled the PIM-SM and PIM-DM protocols. In particular, we used PIM-SM, PIM-DM and DVMRP (Distance-vector multicast routing protocol).

Our simulation model consists of four major parts: a) a database component, b) a client component, c) a server component, and d) a network component. In the following, we explain these four components.

5.1.1 The Database Component

The database consists of the item table and the group table. The *item table* is organized as a series of records. Each record represents information regarding a data item. Every record has three entries: id, encryption key of the data item and a list of keys of each group interested in receiving the item.

Similarly, the *group table* is organized as a series of records where each record holds information regarding the group. Each record has four entries: id, encryption key of the group, the expiry time of the group and a list of clients that belong to this group.

5.1.2 The Client Component

In our model, the client is interested in a specific set of data items. *MaxItemsClient* denotes the maximum number of

items a client can subscribe to with a minimum being 1.

The client requests to obtain its items of interest from the Subscription Server for a certain time duration, the maximum duration being *NumEpochs* - the number of epochs for which the simulation runs. Modeling of both the items requested and the duration is random.

Once the Subscription Server allots the client to a specific group, the client can begin downloading the items sent to it by the Broadcast Server. It is important to note that the client will be interested only in some of the items being broadcasted. Whether or not a client downloads an item is dependent on a) whether a client is authorized to do so (whether the client has the key to obtaining information), and b) whether a client wants the information.

Finally, when the client's subscription expires, the client re-subscribes after waiting a random amount of time denoted by *MeanWaitPeriod*. The waiting period between subscriptions is a random variable generated from an exponential distribution.

5.1.3 The Server Component

As mentioned earlier we have modeled both a Subscription Server and a Broadcast Server. The Subscription Server handles the registry of a new client and the maintenance of the database. The Broadcast Server actually broadcasts packets to a common multicast group address shared by all groups. However, in the PTP case it sends each packet of the broadcast individually to each interested client.

5.1.4 The Network Component

In order to get a more realistic scenario; we ran our protocol on a simulated network of 255 nodes whose topology is based on the Mbone structure. Future research will involve simulation of larger and more complex topologies. We are exploring the possibilities of simulating the I-DG using topology generators like *gt-itm* and *inet*.

5.2. Performance Metrics

The following are the performance metrics we use in our evaluation:

- **Average Broadcast Length (ABL):** This parameter, measured in KiloBytes, defines how long a broadcast is on an average. The access time of a client - the amount of time a client will have to wait to receive a broadcast - is a function of ABL. Since we desire to study the efficacy of I-DG in providing real-time access to varied pieces of information, this metric is critical.
- **Load on the network:** Multicast protocols are notorious for the network bandwidth they consume. Running any reliable application above this would further

increase the chances of congestion of the intermediate network. Hence, it is of very great concern to us that the I-DG per se is a "light" protocol to not add to this problem. Therefore, we keep track of the total number of packets generated by the multicast protocol during the simulation. Note that some of these packets are *Overhead packets* generated to create and maintain the multicast tree structure. When a node joins the multicast network, a tree branch is created such that the packet traffic can reach the new node. When the node wishes to unsubscribe, the multicast tree can be pruned. Thus, the multicast protocol sends periodic messages to see if the nodes are still in the multicast.

5.3. Results of the Simulation

We have compared unicasting, DVMRP, PIM-DM and PIM-SM. Figure 3 shows the results of our experiments. The results are obtained using a database of 100 data items and an average of 10 data items per client. Each data item is 1KB. The epoch length is 10 minutes. Each client is subscribed for at least 1 epoch and is re-subscribed automatically after an exponentially random time interval. The simulation is run for 30 minutes. Data keys are 128 bits long. The bandwidth of each link is 2Mbps and the link delay is 2ms. The results are shown in the Figures 3 and 4.

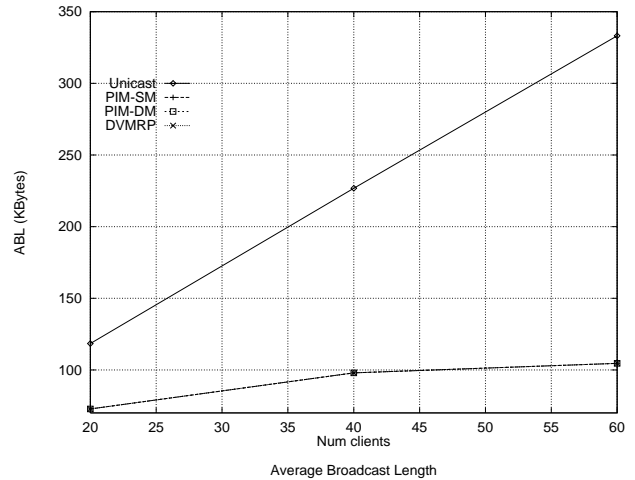


Figure 3. Baseline Simulation Results

Firstly, note that the ABL for all protocols except the Unicast is the same for all nodes. This is expected since the broadcast is prepared the same way using the I-DG technique. Recall that under unicasting, we modeled the point-to-point data delivery approach where a data item is sent as many times as there are subscribers. Each data item is sent to its subscriber only. Therefore, there is no packet duplication in the Unicast network.

The baseline results agree with earlier simulations of the DG protocol under a wireless network. Basically, the size

of the broadcast increases with increasing client load in the system, but the rate of increase slows down. This is due to the fact that as the number of clients is increased, most of the data items are already included in the broadcast. Thus, additional clients when the number of clients is high, do not increase the size of the broadcast as much as when the number of clients is low.

The load (number of packets) generated by different multicast protocols is shown in Figure 4. Here, we compare unicasting, DVMRP, PIM-DM and PIM-SM.

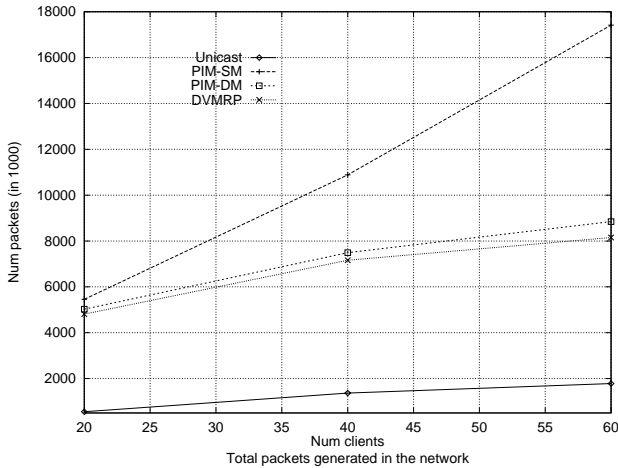


Figure 4. Load on the network

Figure 4 shows how the load on the network varies with the number of nodes participating in the multicast. Each link had a separate packet counter in each direction. The load is simply the summation of the packet counts over all the links in both directions.

Here surprisingly we see that the Unicast protocol performs better than all the multicast protocols. We theorized that this was probably because of the multicast overhead. Further study revealed that the multicast overhead accounted for 10 percent of the total load. However the multicast protocols were generating packets an order of a magnitude greater than the Unicast! When we looked into the implementation, we found that the polling for the multicast tree maintenance happened every 0.5 secs. This combined with the fact that a prune packet is generated in response to a multicast packet helped convince us that unicasting was indeed a better alternative in scenarios where the participating multicasting nodes do not vary very often and are small in number.

Note that with fewer nodes, PIM-SM generated more packets than PIM-DM partly because we ran the simulations with a rather dense network. PIM-SM is expected to perform better with a sparse network (i.e., with fewer clients at the leaf nodes). Although at first glance, a network with only 20 nodes in the multicast seems sparse, these nodes make up about 10 percent of the entire network. We believe

that the PIM-SM protocol would work better for even lower network densities. Extending the graph in the negative direction would suggest the same.

6. Conclusions and Future Work

In this paper, we proposed I-DG, a secure data dissemination protocol that can be implemented in the Internet environment, within the framework of an IP multicasting protocol. The I-DG protocol allows dynamic subscription to multicast data and utilizes an efficient security mechanism for managing access to data.

We have discussed candidate multicasting protocols, and simulated the I-DG technique with three such protocols along with point-to-point data dissemination. Our results showed that, I-DG requires least bandwidth in between the broadcast originator and the entry point to the network, but due to the underlying multicast protocols, the network ends up being swarmed with multicast packets. This is precisely why network providers are reluctant to support multicast traffic.

The results of our simulations suggest that IP Multicasting is indeed costly from the point of view of the overall network. Clearly, multicasting is more advantageous than unicasting for the data dissemination application. However, due to network congestion, the rate at which the broadcast is sent must be kept much lower than the maximum available bandwidth. Otherwise, the network will start dropping packets.

This paper provides motivation for future work in various areas. First, we plan to perform extensive performance tests for I-DG under different IP multicasting protocols. These tests are needed to choose the right multicasting protocol with various data, customer and network parameters. Particularly, it is interesting to see the performance in sparse and dense networks.

A second area for future work is in the context of protocol design. We believe that data dissemination applications could benefit from a new multicasting protocol. Currently, the entire multicast message is sent to all the nodes. It could be desirable to break the multicast into smaller subsets and send these subsets to corresponding subscribers. This necessitates cooperation of the information server with the multicasting application. The multicast groups can be defined using I-DG's grouping criterion, and group keys can be assigned accordingly. The packets can then be routed according to client groupings. We plan to investigate this approach further.

References

- [1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: Data management for asymmetric

- communication environments. In *Proc. ACM SIGMOD*, San Jose, California, May 1995.
- [2] K. P. Birman. The process group approach to reliable distributed computing. *Communications of the ACM*, 36(12), 1993.
- [3] L. Blazevic and J. Le Boudec. Distributed core multicast (dcm): a multicast routing protocol for any groups with few receivers. In *Proc. of the ACM SIGCOMM'99*, volume 29, 1999.
- [4] G. Blohm, R. Parikh, E. Davis, A. R. Bhatt, and A. Ware. Information dissemination via global broadcast service (gbs). In *Proc. MILCOM 96*, McLean, VA, October 1996.
- [5] C. Blundo, A. Cresti, and A. DeSantis. Space requirements for broadcast encryption. In *Proc. Advances in Cryptology - EUROCRYPT'94*, Perugia, Italy, May 1994.
- [6] B. Cain, S. Deering, and A. Thyagarajan. Internet group management protocol, version 3, February 1999. Internet Engineering Task Force (IETF), Internet draft (work in progress).
- [7] A. Celik and A. Datta. A scalable approach for subscription-based information commerce. In *Proc. of Workshop on E-Commerce and Web-based Information Systems, Milpitas, California*, June 2000.
- [8] A. Datta, A. Celik, J.K. Kim, D. VanderMeer, and V. Kumar. Adaptive broadcast protocols to support power conservant retrieval by mobile users. In *Proceedings of the Thirteenth International Conference on Data Engineering (ICDE 1997)*, pages 124–133, Birmingham, UK, April 1997.
- [9] A. Datta, D. VanderMeer, A. Celik, and V. Kumar. Adaptive broadcast protocols to support efficient and energy conserving retrieval from databases in mobile computing environments. *ACM Transactions on Database Systems (TODS)*, 24(1):1–79, March 1999.
- [10] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol independent multicast sparse-mode (pim-sm): Protocol specification, June 1998. Internet Engineering Task Force (IETF), RFC 2362.
- [11] T. Hardjono, R. Canetti, M. Baugher, and P. Disnmore. Secure ip multicast: Problem areas, framework, and building blocks, 1999. Internet Research Task Force, Internet draft (work in progress).
- [12] H. Harney, M. Baugher, and T. Hardjono. Gkm building block: Group security association (gsa) definition, 2000. Internet Research Task Force (IETF), Internet draft (work in progress).
- [13] T. Imielinski, S. Vishwanath, and B. R. Badrinath. Data on air: Organization and access. *IEEE Transactions on Knowledge and Data Engineering*, 9(3):353–372, May/June 1997.
- [14] W. A. Jackson, K. M. Martin, C. M. O'Keefe, J. Pieprzyk, and R. Safavi-Naini. On sharing many secrets. In *Proc. Advances in Cryptology - ASIACRYPT'94*, Wollongong, Australia, Nov 1994.
- [15] D. Maughan, M. Shertler, M. Schneider, and J. Turner. Internet security association and key management protocol, November 1998. Internet Engineering Task Force (IETF), RFC 2408.
- [16] L. Opyrchal, M. Astley, J. Auerbach, G. Banavar, R. Strom, and D. Sturman. Exploiting ip multicast in content-based publish-subscribe systems. In *Proc. of Middleware 2000*, 2000.
- [17] D. Powell. Group communication. *Communications of the ACM*, 39(4), 1996.
- [18] L. Rizzo and L. Vicisano. A reliable multicast data distribution protocol based on software fec techniques (rmdp). In *Proc. of the Fourth IEEE HPCs'97 Workshop, Chalkidiki, Greece*, June 1997.
- [19] B. Segall and D. Arnold. Elvin has left the building: A publish/subscribe notification service with quenching. In *Proc. of AUUG97, Brisbane, Australia*, September 1997.