Secure Group Communications for Delay-Tolerant Networks

Paul T. Edelman
Baylor University
paul@pauledelman.net

Michael J. Donahoo
Baylor University
jeff_donahoo@baylor.edu

David B. Sturgill
Baylor University
sturgill@cs.baylor.edu

Abstract

The increased demand for mobile communication and use of mobile devices in high-latency, resource impoverished environments has spurred the development and growth of Delay-Tolerant Networks (DTN). DTNs aim to provide interoperability between a range of heterogeneous networks, operating under resource-constrained circumstances and traditional infrastructure networks such as the Internet. Because of the circumstances, DTNs possess some interesting characteristics that make a traditional end-to-end security paradigm unsuitable and increase the value of the overlay's resources. Controlling access to overlay resources and providing for secure group communications over unknown intermediate networks is essential. We propose a novel solution based on previous work in secure group communications using key-graphs and in extension to work on scalable access authorization in self-organizing overlays to provide a scalable mechanism for access control and secure group communications in DTNs. Since resources are especially limited, our implementation focuses on minimizing the traffic on the overlay associated with the maintenance of our solution.

1. Introduction

Delay Tolerant Networking has grown out of the classification of challenged networks, that violate one or more of the assumptions made by the existing TCP/IP Internet. [4] provides an excellent introduction to DTNs, which we summarize here. Challenged networks are, in general, classified by certain characteristics divided into two categories, path and link, and end system. The path and link category includes high latency and low data rate, disconnection, long queuing times, interoperability considerations, and security. End system characteristics include limited longevity, low duty cycle operation, and limited resources. The very basis of Delay-Tolerant Networking is that these types of challenged networks are increasingly becoming important. In his seminal work on the topic, Kevin Fall introduces four examples of challenged networks, terrestrial mobile networks, exotic media networks, military ad-hoc networks, and sensor and sensor/actuator networks. We are most interested in classes of ad-hoc networks.

Challenged networks often arise from various forms of host and router mobility, as well as disconnection due to power management, interference, or unavailability. As malicious Internet traffic increases, interference may be among the most common causes. Fall demonstrates that, despite our desire to apply the well understood TCP/IP solution to challenged networks, the end-to-end nature of TCP/IP and its base assumptions prevent it from being an adequate solution.

Delay-Tolerant Networking provides a solution for resource impoverished networks through an optionally-reliable asynchronous message forwarding protocol that has limited expectations of end-to-end connectivity. This protocol is called the Bundle Protocol [4]. For our purposes, a knowledge of security concerns related to the Bundle Protocol is sufficient. It is important to note that we will address security requirements and solutions at the overlay level and try to avoid addressing issues with a particular implementation of DTN. After an evaluation of the security issues enumerated for DTNs [5] we focus on the following concerns:

1. Control access to overlay resources
2. Allow for explicit revocation of credentials
3. Provide for secure group communications
4. Handle a dynamic group

We do no attempt to address an end-to-end security solution; thus we define a group-oriented security protocol. Simply reviewing the characteristics of challenged networks points out the fact that we will have to be aware of the limited resources we have at our disposal. A security solution to any of the considerations listed above must scale. Most importantly we must minimize the amount of traffic overhead required to maintain security for the group and between individual group members. As Fall notes, we cannot create a chatty proto-

col that involves numerous round trips in order to establish or maintain security. We must also be able to handle a changing group while maintaining security that is achieved without great effort.

Our problem scope is defined in terms of two domains: a subset of security requirements and a class of Delay-Tolerant Networks exhibiting certain characteristics. These characteristics are (1) that the rate of churn (join/leave requests) is lower than the rate of peering decisions, (2) end-to-end disconnection of a node for some time does not mean that the node has left the group, and (3) that the overlay has the connectivity and capacity required to support a certain level of churn in the group.

In our solution, we introduce an approach that provides both scalable access control based on [7] and secure group communications based on [10]. Our solution provides security information for access control and secure group communications with a minimal impact on the DTN overlay in the face of a dynamic group. In addition to providing this security information, our solution may be used effectively by overlay members to implement and enforce a variety of security policies that will protect the DTN overlay under different categories of attacks. We address some security policies in [8].

2. Related Work

Much work on DTN security is directed at end-to-end solutions. Since our focus is on a group-oriented solution, we have found few competing techniques. In [3] the authors argue that attacks on Disruption-Tolerant Networks can be effectively survived without the use of authentication; however, this is based on a given DTN deployment. [3] also notes that any access control may exclude willing participants that are not malicious, thereby reducing the number of routes. We view access control as a mechanism for excluding known compromised or misbehaving nodes, and secure group communications for providing security over untrusted routes. We have demonstrated attacks in our simulations that successfully reduce application traffic throughput by more than what was shown using DieselNet; therefore, having access to some reasonable security solution is worthwhile.

Recently Identity-Based Cryptography has been explored as a mechanism for certain security concerns within Delay and Disruption-Tolerant Networks. [2] shows that, "for authentication and integrity IBC has no significant advantage over traditional cryptography." In [1] the author's third concern, "protection from users and infrastructure nodes whose credentials have been revoked or compromised" overlaps with our concerns. Their solution is the use of time-based keys, which rely on the highly synchronized clocks between all entities, which as we have seen in [6] can be a problem of practicality with respect to actual deployment. We understand the argument that time-based keys provide a system robust to failure, but we believe that our failure mitigation techniques and the replicating nature of Delay-Tolerant Network routing help to overcome the problem of keeping nodes credentials up-to-date. [1] is also based on a form of Identity-Based Cryptography.

To date we have not found any work that is related to providing a solution for secure group communications over DTNs. We believe that for many types of DTN deployments secure group communications will be a highly desirable security feature, particularly for military and law enforcement

3. Solution

Our approach is an amalgamation of two existing solutions to related problems. The first addresses scalable access control in overlay networks [7], while the second addresses secure group communications [10]. We describe our solution in terms of the two areas of access control and secure group communications; however, we only take secure group communications as far as solving the key-distribution problem. We don't require our system to use the group keys we distribute; rather, we assume that these keys will be used at the application layer, or may be integrated into a larger system encrypting all traffic. The security policy of the group is determined by the owner of the overlay and the Authorization Service. As a result, we assume a default policy as we discuss our solution; additional security policies can be developed and added if needed. For details on why we do not use group-keys alone to perform group authentication please see [8].

We have bifurcated our solution into two domains. One domain administers the core accounting of credentials and group state, called the Authorization Service. The second domain is with regard to the individual nodes in the overlay, called the Local Service. Another way to think about the two parts of our solution is that the Authorization Service (AS) defines the group at a given moment in group time, and the Local Service (LS) enforces the integrity of the group.

We address the remainder of the solution in the following order. First, how we define the group. Second, we cover operations that may be performed on the group. This includes events that result in a change in group state. Third, we discuss how we maintain the integrity of the group, and finally we will conclude the solution portion by addressing synchronization, privacy, security and overhead. For a more detailed specification

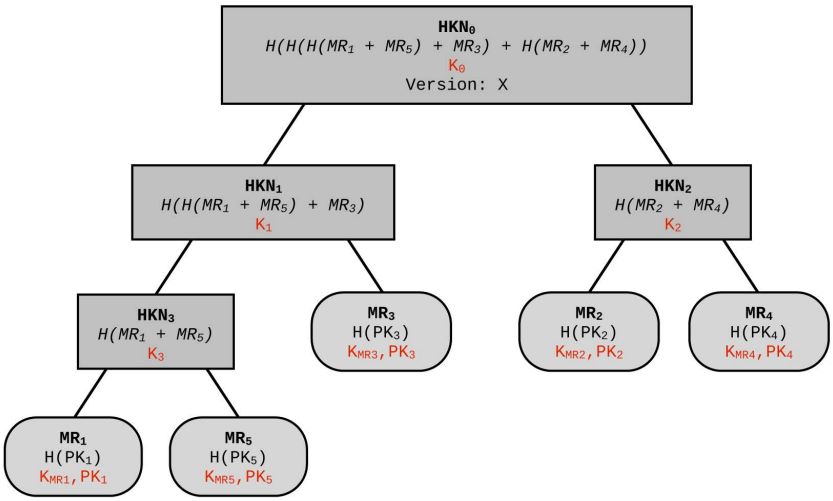


Figure 1. Group Membership Tree

refer to [8].

3.1. Defining the Group

The security core is made up of three parts: the identities of individual clients or members, the group membership tree, which is the authority on who is in the group at a given point in group time, and the credentials and keys produced by the group membership tree and distributed to the group's membership.

When a new member joins the group, its identity information is added to the group membership tree. This in turn affects the state of the tree and the credentials used to prove group membership, which requires that new credentials be distributed. New credentials and keys are distributed via a group membership update message. Likewise when a member is removed from the group, the converse happens with the same effects.

3.1.1. Member Identity

We represent the identity of a member with a membership record. Note well that any client may create a membership record, so holding a membership record does not guarantee membership in the group. To create a membership record, a client simply generates an asymmetric key pair and places their public key in the membership record, the membership record then contains only the client's public key.

3.1.2. Group Membership Tree

Recall that the group membership tree (GMT) keeps up with the state of our group and thus is the authority on group membership at any one time. When a user joins or leaves the group, the GMT is recomputed to reflect the change. This means that value, and in some cases structural changes will occur to the tree. In our architecture, the AS is responsible for the GMT. Specifically, it processes the requests to join, leave or remove and updates the tree accordingly. Different approaches for managing, updating, and distributing updates could be taken. Figure 1 illustrates the structure of the GMT.

The nodes labeled with MR represent nodes with membership records for members in the group; hence, the members are all leaf nodes. The internal nodes, labeled HKN, form the internal structure of the tree including hashes, sub-group, and ultimately the group key. For example, the primary path of MR3 is MR3, HKN1, HKN0, and its sibling path is HKN3, HKN2. When talking about operations, the incident path is analogous to primary path. We see that the hash for MR3 is based on the public key for that member, for which it must possess the private key. All of these items will come into play when validating credentials and identities within the execution of the overlay.

Once the group membership tree has been updated to reflect the current group state, we must generate an update to inform the group membership of the changes. We aptly call this update a group membership update (GMU).

3.2. Operations on the Group

A number of operations may be performed on the group that change the group's membership. Members may join and leave the group, and even be forcibly removed, which is covered in [8] along with secondary use

cases for the various operations.

3.2.1. Joining the Group

When a Requesting Client (RC) wishes to join the group, it must first ask permission and be admitted by the Authorization Service (AS). After admission, the join protocol follows several steps:

1. The RC sends a request to the AS to join the group. This request includes the RC's self generated membership record, which contains its public key.
2. Once the AS has received the request, it determines if the user should be added to the group. We assume that this user is permitted access.
3. The AS creates a Membership Record (MR) for the RC and add it to the Group Membership Tree (GMT).
4. The AS then recomputes the GMT. See the section titled, "Updating the Group Membership Tree."
5. The AS sends a response to the RC that includes their new group credentials.
6. After the GMT has been recomputed, the AS creates a new group membership update (GMU) that contains the new group credentials needed by all members of the group. See the section titled, "Group Membership Update."
7. The AS sends the GMU to all group members.
8. The RC is considered a member of the group.

If the requesting client is not permitted access to the group, then the AS sends a response to the client informing that it was not permitted access. When a client successfully joins the group, we must recompute the GMT to reflect the changes. This means updating the hashes and keys in the GMT along the incident path. For a join the old keys may be used to encrypt the new keys. If a structural change occurs, we may also need to inform some existing members of changes to their sibling paths.

3.2.2. Leaving the Group

A group member may leave the group for any number of reasons, including being involuntarily removed by some authority. Leaving the group has a greater impact on our protocol and the group as a whole. This is because the user that is leaving has the set of past credentials. The following is the procedure for leaving the group:

1. The RC sends a request to the AS to leave the group. This request includes the member's self-signed membership record, which the AS can verify as authentic since it possesses all member's public keys.
2. Once the AS has received the request, it confirms the request is not a duplicate request, as well as verifies the digital signature.
3. The AS removes the membership record (MR) for the RC.
4. The AS then recomputes the GMT. See the section titled, "Updating the Group Membership Tree."
5. After the GMT has been recomputed, the AS creates a new group membership update (GMU) that contains the new group credentials needed by all members of the group. See the section titled, "Group Membership Update."
6. The AS sends the GMU to all group members.
7. At this point, the RC is no longer considered a member of the group.

3.2.3. Updating the Group Membership Tree

Whenever any operation is performed on the group by the Authorization Service, the group membership tree (GMT) undergoes an initial change (adding or removing a member) that results in the changing of credentials used in the tree. The following outlines this process:

1. Beginning with the incident node traverse the path up to the root node of the tree.
2. At each node:
   (a) Cache the current hash value and key.
   (b) Generate a new hash value based on the rules for generating hash values at nodes.
   (c) Create a new random key.
   (d) Add the node to the primary path (a list of nodes we use for computation).
   (e) Locate the sibling node for the primary node and add it to the sibling path (similar to the primary path).

Figure 2 shows MR3 being removed from the GMT and the result of that operation. In this case, we free the node that previously contained the membership record MR3, making it available for the insertion of a future membership record. Black nodes represent the primary path, and the red nodes represent the sibling path. Along the primary path, all hashes and keys are recomputed, resulting in a new root hash and new group keys. Once the GMT has been updated, we create and send out a group membership update (GMU).

3.3. Maintaining Group Integrity

Maintaining group integrity across a number of separate nodes with limited connection capabilities and a

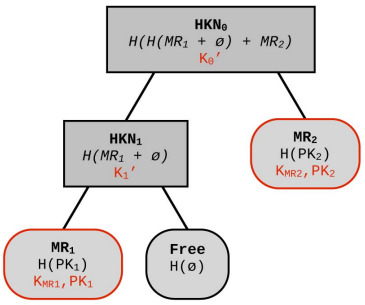


Figure 2. Removal of MR_3

central authority on the state of the group requires that updates to group credentials and state be distributed to group members. We are able to maintain the integrity and security of the group as well as we are able to keep individual group members appraised of the state of the group.

3.3.1. Group Membership Update

In order to support a dynamic group with dynamic credentials, we must send out some information updating those credentials every time a change in group membership occurs. The process of updating group member's credentials includes updating both access control credentials and distributing new group keys securely.

We have currently limited the required update information to a root certificate and digital signature, a hash list, key list, membership record list, any member specific blocks if required, and a digital signature for the entire group membership update. Figure 4 gives the structure.

Group Membership Update
Root Certificate
Root Certificate Digital Signature
Hash List
Key List
Membership Record List
Member Specific Blocks
Group Membership Update Digital Signature

Figure 3. Group Membership Update

The root certificate serves as the authentication credential for group members in the overlay. We sign it separately so that we can exchange only the root certificate during node interconnection negotiation. Next, the hash list contains a formatted list of hash/key node identifiers and values. This allows us to update group members as to changes in values along their sibling path that they might need to know. The key list is a list of hash/key identifiers and new key values encrypted with the appropriate previous key so as to maintain secrecy. The membership record list is a list of membership records and flags to indicate members that have joined or left. Nodes will use this information to update relationships with neighbors that just left the group. In order to send out specific structural changes to primary and sibling paths, we provide a member specific block section that stores formatted member specific blocks with this information. Finally a digital signature for the entire group membership update ensures the authenticity and integrity of the update during transport. The following steps detail the procedure for constructing a group membership update after a change in the group membership tree. See [8] for specifics of how the GMU is created.

3.3.2. Node Interconnection Negotiation

Node interconnection negotiation is for the purpose of access control; it answers the question, "Are you in the group?" When a member in the group meets a new neighbor, that member must be able to determine the status of the neighbor in order to execute the access control policy assigned to the group. How the member node acts based on the neighbor's status depends on the access control policy. We answer the question, "Are you in the group?" by computing the reverse primary path hash. When two clients meet, they exchange group membership credentials with one another. These credentials include the member's membership record, sibling path with values, and the member's most recent group version. Each client verifies the member's credentials by recomputing hashes starting with the membership record of the client up to the root hash. If a client is a member of the group, the hash equals the latest version of the root hash. If two clients have the same group version number and credentials for that version, they both consider one another as being in the group at that time. Figure 5 illustrates reverse primary path hash computation.

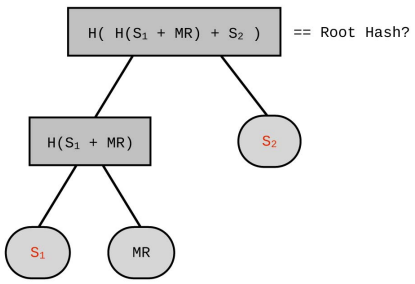


Figure 4. Reverse hash computation

Table 1. Protocol overhead

Operation	Cost
Change in Group Membership (AS)	*O(log n)*
Size of Group Membership Update	*O(log n)*
Overlay Impact	*O(m log n)*

3.4. Synchronization, Privacy and Security

As with any networked system, failure to communicate properly can easily arise in our context, which results in stale credentials. To deal with stale credentials, local nodes keep a bounded history of group state. Refer to [8] for specifics.

For applications requiring message privacy within the group at a given version of group membership, we provide group keys based on the key graph solution for distributing group keys, which maintains both past and future secrecy [10]. Because of the guarantee of both past and future secrecy when updating group keys, applications do not have to worry about malicious members or clients forging group version numbers on messages or intercepting message contents. As a result, group key encrypted messages can be forwarded to any willing participant, and privacy is upheld without an impact on routing.

Our solution requires that only certain key information remain secret, namely individual private and symmetric keys, and group and sub-group keys within their respective groups. For access control only individual private keys must remain secret, all other hash and group information is known. Private key information is never transmitted in any form.

3.5. Overhead

In general the overhead of our solution is tied to the tree structure we use for maintaining the group at the Authorization Service (AS). The actual impact of our updates on the overlay depends largely on the routing algorithm used, as well as other factors related to the organization of the overlay. As a result of this we generalize the impact on the overlay as some factor times the size of the group membership update. See table 1 for details.

4. Implementation

In addition to designing the protocol we also developed a prototype implementation based on the DTNRG reference implementation. Certain limitations within the reference implementation prompted us to start porting our implementation to libraries for our solution in C++ and Java that can be used independently from any specific DTN implementation. At the time of this writing these libraries are under development. Our work with the DTN reference implementation gave us great insight into how to design and implement our protocol. See the full technical report for more details [8].

5. Evaluation

We adopted the ONE simulator and modified it to incorporate a prototype of our protocol, three different types of security policies, and two types of attack generators in order to evaluate the performance of our solution. All of our simulations were run on the Baylor ARCS Kodiak cluster.

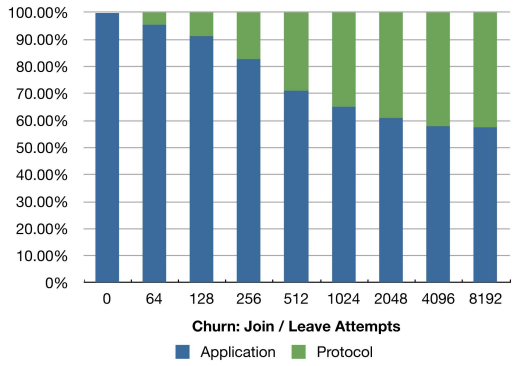
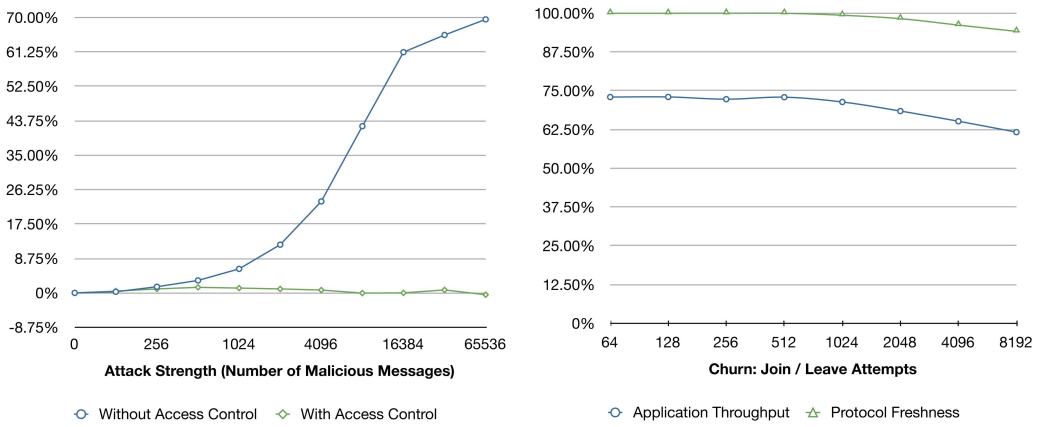


Figure 5. Percent traffic load

Our objective is to demonstrate that our solution is an effective security enhancement for a sub-set of DTNs as defined by our problem scope, that does not overwhelm the overlay, and provides useful security in the face of a certain class of attacks. Because we already know that the key graph solution for group key distribution maintains group communication security [10], we are only interested in evaluating our solution with respect to three areas. First, what is the cost of maintaining group integrity, and how effective are we at doing that? Second, can access control information be used to form security policies that mitigate various types of denial of service attacks on a DTN overlay? Finally, how does our solution scale?

We evaluate our experiments in terms of overlay performance, overlay stress, and group integrity. When conducting our experiments, we introduce three categories of traffic into the overlay: application, protocol (SGS/SGCP), and malicious traffic. Overlay performance is a measure of application traffic throughput and loss. Higher throughput and lower loss provide better performance. Overlay stress is a measure of all traffic, although we may view them separately, with respect to load or how many messages are routed for a given num-



(a) Percent Load as Attack Increases

(b) Percent Success as Churn Increases

Figure 6. Protocol Performance

ber of messages created. Group integrity measures how successful we are at maintaining up-to-date group state and credentials.

To answer our first question we conducted experiments to see (1) how group churn impacted the overlay and (2) how successful we were at keeping group members up-to-date. In our simulations we see that even when a large number of changes in group membership occur we maintain reasonable application throughput and protocol freshness. In our worst case scenario we reduce application throughput by 15% from the baseline performance with an average of 23.5 group changes per node over a 24 hour period. Table 2 shows the impact on the overlay as churn increases. Recall that our overlay contains only 32 nodes. We believe that in application, rates of churn this high for a group this small are unlikely if even necessary. Figure 6 shows what we would expect, as we increase churn we introduce more and more protocol traffic, while application traffic is fixed such that our protocol begins to place a higher percentage load on the overlay.

The next evaluation that we perform aims to answer our second question. To accomplish this we establish a fixed group with no churn, and execute a number of denial of service attacks to see how the overlay handles extreme increases in malicious traffic. We then also perform the same test with a strict security policy enabled [8]. Figure 7 shows that if even a small number of malicious nodes are present, which are capable of introducing large quantities of traffic relative to overlay capacity, that application throughput drops off sharply. Figure 7 also shows that given the information to determine group membership we are able to successfully keep the unauthorized malicious nodes from flooding the overlay with malicious traffic. In the event that circumstances exist allowing for such an attack, having the ability to control access to the overlay is an effective defense. This of course does not address the issue of compromised member nodes. We assume for our purposes that member nodes are trusted to not become compromised.

To answer our final question we conduct a number of experiments where we increase group size up to 1024 nodes. These experiments indicate that the length of the history for dealing with stale credentials is relative to the size of the group given a projected rate of churn. A large group with a higher rate of churn needs to keep more history in order to avoid increased overlay impact by our protocol. We also conduct simulations to show the computation overhead in terms of encryption and decryption operations. Figure 8 shows the results of our first experiment for a group of 512 nodes, as opposed to 32 nodes. Due to space limitations please refer to [8] for the full set of experiments and data. Overall our data suggests that our solution does scale as group size increases for the class of DTNs considered.

6. Conclusion

We have taken two existing solutions to related and important security concerns for Delay-Tolerant Networks and expanded and integrated them to create a group-oriented security solution for DTN that provides access control and secure group communications. In addition to drafting and evaluating this solution for feasibility, we have also implemented a prototype addition to the DTNRG reference implementation and begun development of C++ and Java libraries. We have shown that we are capable of providing security information to the membership of the overlay with minimal consumption of overlay resources in the face of a dynamic group. The security information we provide allows for the determi-

Table 2. Effect of churn on the group and overlay

Request Attempts	Group Versions	Global Messages Sent	Freshness	Application Throughput	Throughput Delta	Changes / Node
0	0	0	N/A	93.06%	0.00%	N/A
64	100	259	100%	91.78%	1.28%	3.13
128	196	504	99.96%	91.03%	2.03%	6.13
256	341	999	98.95%	88.19%	4.87%	10.66
512	483	1,842	95.21%	83.12%	9.94%	15.09
1024	600	2,327	93.66%	82.81%	10.25%	18.75
2048	692	2,801	91.89%	78.97%	14.09%	21.63
4096	714	3,064	90.35%	79.72%	13.34%	22.31
8192	752	3,207	89.39%	77.97%	15.09%	23.50

nation of group membership without global queries and versioned message privacy in the face of unsecured hops without impacting DTN routing. We have also shown that the security information provided by our solution may be used to implement security policies that are effective at defending against a class of denial-of-service based attacks on the overlay.

7. Future Work

We anticipate several areas of continued work. First, we hope to improve the ONE simulator, exploiting parallelism to improve performance where possible.

As we have seen from our experiments, it is difficult to be protected against join request flood attacks if our security policy allows for joins to be processed using the resources of the overlay. We also would like to conduct future experiments related to the case of a compromised member node that turns malicious. Additional work should be done to investigate barriers to entry such as the type of computational puzzle used by [9]. Such computational puzzles could either be used for the generation of join requests or even identities themselves.

Additional security policies should be considered and evaluated under different overlay scenarios. More options for failure mitigation should also be investigated, including different strategies for storing history and bringing stale group members back up-to-date. Such an evaluation may also reveal new strategies to help reduce global traffic (*e.g.*, batch updates during periods of higher rates of churn). Future work should also consider different methods for the structure and maintenance of the group membership tree.

References

[1] S. K. A. Seth. Practical security for disconnected nodes, February 2008.

[2] N. Asokan, K. Kostiainen, P. Ginzboorg, J. Ott, and C. Luo. Applicability of identity-based cryptography for disruption-tolerant networking. In *MobiOpp '07: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 52–56, New York, NY, USA, 2007. ACM.

[3] J. Burgess, G. D. Bissias, M. D. Corner, and B. N. Levine. Surviving attacks on disruption-tolerant networks without authentication. In *MobiHoc '07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 61–70, New York, NY, USA, 2007. ACM.

[4] K. Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34, New York, NY, USA, 2003. ACM.

[5] S. Farrell and V. Cahill. Security considerations in space and delay tolerant networks. In *SMC-IT '06: Proceedings of the 2nd IEEE International Conference on Space Mission Challenges for Information Technology*, pages 29–38, Washington, DC, USA, 2006. IEEE Computer Society.

[6] P. H. L. Wood, W. Eddy. A bundle of problems. In *IEEE Aerospace conference*, pages 1–17, Washington, DC, USA, 2009. IEEE Computer Society.

[7] S. S. M. J. Donahoo, D. B. Sturgill. Scalable access authorization in self-organizing overlays. Technical report, Baylor University, Waco, TX, 2002.

[8] D. S. P.T. Edelman, M. J. Donahoo. Secure group communications for delay-tolerant networks. Master's thesis, Baylor University, Waco, TX, 2009.

[9] E. G. S. V. Vishnumurthy, S. Chandrakumar. Karma: A secure economic framework for peer-to-peer resource sharing, 2003.

[10] C. K. Wong, M. Gouda, and S. S. Lam. Secure group communications using key graphs. In *SIGCOMM '98: Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 68–79, New York, NY, USA, 1998. ACM.