Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Electronic Notes in Theoretical Computer Science 171 (2007) 57–69
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

A Mechanism for Communication-Efficient Broadcast Encryption over Wireless Ad Hoc Networks

Reza Curtmola [1] Seny Kamara [2]

Department of Computer Science
The Johns Hopkins University
Baltimore, USA

Abstract

Due to its low communication cost, stateful broadcast encryption is an appealing solution for secure content distribution in mobile ad hoc wireless networks (MANETs). Unfortunately, the inherent limitations of MANETs prevent a standard application of such schemes since they require receivers to be online. In this paper, we present a reliable message delivery mechanism for MANETs that is based on erasure codes and that leverages node mobility in order to achieve non-interactive recovery of missed messages. We then show how our mechanism can be used to reliably deliver the key updates of a stateful broadcast encryption scheme. Our solution has several useful properties: it allows trade-offs between the amount of storage required at each node and the speed of message recovery; and it has the ability to leverage the resources of unauthorized nodes. We evaluate the performance of our approach through simulation, and show that it achieves good performance for networks with high node density.

Keywords: reliable message delivery, broadcast encryption, key updates, storage, mobility, wireless ad hoc wireless networks.

1 Introduction

We consider the problem of secure content distribution in mobile ad hoc networks (MANETs), where a single source disseminates data to a dynamically changing group of authorized receivers. In this context, the source and the receivers are mobile nodes part of a multihop ad hoc wireless network. A simple way of ensuring the secrecy of the data (i.e., that only an authorized subset of users be able to recover it), is for the source to use (at the application layer) a broadcast encryption scheme to encrypt all messages before transmission. This guarantees that even if the encrypted data is received by all users, only the authorized subset will be able to recover the content.

[1] Email:crix@cs.jhu.edu
[2] Email:seny@cs.jhu.edu

doi:10.1016/j.entcs.2006.11.009

Broadcast encryption was first introduced by Fiat and Naor [8] and has since been studied under many variants [19,11,10,27,26]. Typically, broadcast encryption schemes are classified as either *stateful* or *stateless*. Stateless schemes provide users with long-term keys that are never changed throughout the lifetime of the system, while stateful schemes provide keys that may be updated after join or revocation events. While the former require receivers to be online in order to receive key update messages, a simple argument shows that after a logarithmic (in the number of users) number of revocations occur, stateful schemes typically achieve lower communication cost than stateless schemes (see Appendix A).

The characteristic properties of MANETs, including low bandwidth, lossy links, and mobility, make deploying broadcast encryption challenging for the following reasons. The limited bandwidth available in such networks stresses the importance of minimal communication costs and favors the use of stateful schemes. However, a standard application of stateful broadcast encryption is not possible since it requires receivers to be online, which cannot be guaranteed in MANETs since mobility may cause nodes to become partitioned from the network.

In this work, we propose a reliable message delivery mechanism for MANETs which guarantees that partitioned nodes can recover lost messages within a reasonable amount of time. Intuitively, we require each node in the network to store a "piece" of a message, and we leverage node mobility to allow partitioned nodes to recover missed messages after encountering a specified number of nodes. Our mechanism is based on erasure codes and has the advantage of allowing a trade-off between message recovery time and node storage.

We then show how to achieve communication efficient secure content distribution in MANETs using a stateful broadcast encryption scheme provisioned with our reliable message delivery mechanism. In particular, we use our mechanism to deliver the scheme's key updates in a reliable way to each node in the network. We are thus able to preserve the advantage, in terms of reduced communication cost at the source, that stateful broadcast encryption schemes offer over stateless schemes. Besides communication efficiency, we note that since our reliability mechanism is independent of our secure content distribution mechanism (i.e., of the underlying broadcast encryption scheme), our solution has the ability to leverage the resources of unauthorized nodes. In other words, the reliability of our content distribution scheme improves not only with the number of authorized users, but also with the number of unauthorized users.

2 Related Work

Reliable message delivery can be achieved using *interactive* or *non-interactive* means, depending on whether clients contact the source to retrieve missed messages. In the ad hoc wireless setting, characterized by a relatively high packet loss and limited bandwidth, interactive solutions [20,29,31,23] are undesirable because the increased communication cost can cause packet implosion at the source. Moreover, these interactive schemes become problematic if nodes are partitioned and cannot contact the

center. In order to avoid these scalability and connectivity issues, in this work, we only consider non-interactive solutions. Some of these interactive solutions [28,29] use forward error-correcting codes to improve the reliability of multicast rekeying.

In gossip-based reliable multicast protocols [6,16] the members of the multicast group pro-actively exchange information about the "state" of the group with a set of other group members, which are selected at random. This allows probabilistic recovery of missed packets at the cost of increased traffic caused by repeated rounds of gossip message exchanges. Such protocols have not been considered within a security framework.

Self-healing key distribution, proposed in [25] and improved in [14], can be used to reliably (and non-interactively) deliver key updates to users. Self-healing schemes allow users that missed t consecutive updates to recover them if they receive one update preceding and one update following the t missed updates. Unfortunately, even for the most efficient self-healing scheme [14] the size of updates is $O(t \cdot v)$, where v is the size of the largest coalition (of revoked users) the scheme can handle. And since both t and v must be fixed during system initialization, one will typically have to choose large values in order to cover the largest period in which consecutive updates could be missed, and the largest possible coalition of revoked nodes that may form.

GKMPAN [32,33] provides scalable and efficient mechanisms for group rekeying in ad hoc wireless networks, based on probabilistic key pre-deployment [7,34]. During system setup, nodes are pre-loaded with m symmetric keys chosen out of a pool of l keys. These keys are later used by neighboring nodes to establish secure channels. When an addition or revocation event occurs, the group manager generates an intermediate key and propagates it through the network. Nodes use this intermediate key to update both the group key and the keys they share with any revoked node. Since, with high probability, any two nodes will store at least one key in common, a partitioned node will likely be able to recover missed updates from its neighbors. This work comes closest to ours since nodes that miss update messages (e.g., due to lossy links or network partitions) are able to recover the new group key mostly through local interactions (with low probability a node might need to contact the key manager). This partial statelessness feature makes GKMPAN attractive for use in MANETs, especially since rekeying has a small per node transmission cost.

While GKMPAN has several desirable features, the approach we take in this work achieves new and useful properties. For instance, since our reliability mechanism is independent of our secure content distribution mechanism (i.e., of the underlying broadcast encryption scheme), we can leverage the resources of nodes that are *not* authorized group members. Our system also allows partitioned nodes to recover any arbitrary (but fixed) number of missed messages with certainty, while GKMPAN only provides a probabilistic guarantee. In addition, since GKMPAN requires all connected nodes to erase the intermediate key after they update the group key and their compromised keys, a partitioned node that rejoins the network after a missed rekey message will never be able to update its compromised keys. As a

consequence, the probability that it shares a key with its neighbors (and that it can establish a secure channel to recover the missed update) will decrease as the number of missed messages increases. Finally, we do not assume that node compromises are detected within a bounded period of time.

3 Preliminaries

Network and security model.

This work relies on several network and security assumptions. The receivers in a broadcast encryption scheme are nodes in a mobile ad hoc network. In addition to nodes in the subset of authorized receivers, the network may also contain non-authorized nodes, which may offer their services to improve the quality of the content distribution service. If an authorized receiver is compromised, the attacker will obtain its credentials, and will be able to decrypt the broadcast content until the compromise has been detected.

We assume the existence of a network layer mechanism (e.g., a multicast routing protocol [22,9]) that ensures efficient message delivery from the center to a specified group of nodes in the ad hoc network. We also assume that the communication and computational capabilities of a node are limited, but that it has relatively large storage capabilities (e.g., a PDA can store up to several MBytes).

Erasure codes.

An erasure code [21,15,3] is a pair of (possibly) probabilistic algorithms $\mathcal{C} = (\mathsf{Encode}, \mathsf{Decode})$. A sender uses the Encode algorithm to encode a message m, composed of ℓ symbols, into a codeword c, composed of λ symbols. We say that $\mathcal{C}$ has minimum distance d if it can tolerate up to $d-1$ erasures, or in other words, if the receiver can recover m from any $\lambda - d + 1$ symbols of c. The *rate* of a code is the value $\rho = \frac{\ell}{\lambda}$, and we say that $\mathcal{C}$ is a (λ, ℓ, d)-erasure code if it encodes a message of ℓ symbols into a codeword of λ symbols, and has minimum distance d. Note that symbols can be any "unit" of data (e.g., packets or bits), but when applying our construction to broadcast encryption, we will assume symbols are blocks of a specified bit length (see Section 5 for details).

For several examples of erasure codes, encoding and decoding can be done efficiently. In particular, Reed-Solomon [21] codes can be encoded and decoded in time $O(\lambda^2)$, and Tornado codes [15] in time $O(\lambda)$ (though Tornado codes only guarantee that messages are recovered with high probability).

Finally, we note that erasure codes are vulnerable to pollution attacks, which are denial of service attacks where an adversary introduces invalid symbols into the decoding process. Such attacks, however, can be mitigated by using distillation codes [13] at the cost of increased computational and communication overhead.

Broadcast encryption.

We use interchangeably the terms *source* and *center* to refer to the source that disseminates the content. Similarly, we interchangeably denote the users that receive

content as *users* or *receivers*. Users are either *authorized* (i.e., are *non-revoked* and are allowed to access the content) or *revoked*. We use revoked users as a generic term to denote users that are not allowed to access the content. We use N to denote the set of users in the system, $\mathsf{G} \subseteq \mathsf{N}$ for the set of authorized users and $\mathsf{R} = \mathsf{N} \backslash \mathsf{G}$ for the set of revoked users. Let $n = |\mathsf{N}|$ and $r = n - |\mathsf{G}|$.

Let $\mathsf{BE} = (\mathcal{G}^{\mathsf{BE}}, \mathcal{E}^{\mathsf{BE}}_{\mathsf{G}}, \mathcal{D}^{\mathsf{BE}}_{\mathsf{G}})$ be a stateful broadcast encryption scheme and $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme. Users share *long-term keys* with the center. A broadcast encryption scheme encrypts each message with a *session key* and then encrypts the session key such that only non-revoked users are able to decrypt it based on their long-term keys. Specifically, for a message m the center broadcasts $[\mathcal{E}^{\mathsf{BE}}_{\mathsf{G}}(K), \mathcal{E}_K(m)]$, where K is the session key and $\mathcal{E}^{\mathsf{BE}}_{\mathsf{G}}(K)$ is the *header* that encapsulates the session key. When comparing the communication cost of a broadcast encryption scheme, we usually refer to the size of this header.

4 Reliable Message Delivery

In mobile ad hoc networks nodes often miss broadcast messages due to network partitions or lossy links. A reliable message delivery mechanism enables a source to deliver messages to all the nodes in a MANET in a reliable way. We will mainly be interested in three aspects of such a mechanism: scalability, storage per node, and recovery time. We loosely define scalability as the mechanism's ability to handle large network sizes, and recovery time as the number of nodes a rejoining node needs to encounter in order to recover a missed message. We begin by reviewing two basic approaches to reliable message delivery and point out their limitations in terms of scalability and storage requirements. We then present our preferred mechanism which is highly scalable and allows a trade-off between storage and recovery time (i.e., the number of encounters needed).

An interactive approach.

A trivial reliable delivery mechanism can be constructed as follows. The content distribution source stores each message it sends. When a node that missed a message (either because it was partitioned or because it was off-line) rejoins the network, it simply asks the source for the messages it missed. This interactive solution is clearly not scalable as the source might be overwhelmed with message requests when the network size is large. It is also problematic because the node might not be able to directly communicate with the source (e.g., it might be out of the source's transmission range) and re-transmission might have to be relayed through other nodes, which would contribute to an increase in overall network traffic.

A (naive) non-interactive approach.

A naive non-interactive solution would be to require each node that receives a message from the source to store it. When a node that missed a message rejoins the network, it can ask any node that was connected during the broadcast for the missed message. This approach achieves optimal recovery time since disconnected

nodes need only encounter a single node in order to recover a message. On the other hand, it requires a large amount of storage per node since each node will have to store $r \cdot q$ bits, where r is the number of messages broadcast by the source, and q is the size of each message.

4.1 Our Approach

We propose a non-interactive solution, but provide a method that requires nodes to store less than the naive non-interactive scheme. The cost paid is that disconnected nodes need to encounter more than one node in order to recover the messages they missed. Since nodes are mobile, the rejoining nodes will encounter other nodes, so it seems reasonable to trade storage size for recovery time.

Informally, our approach is to use an erasure code to encode each message m broadcast by the source. Upon receiving the codeword each node will only store a "piece" of the encoded message so that when a node rejoins the network, it will be able to recover the messages it lost after it encounters a specified number of nodes. We note that node mobility is *crucial* and *beneficial* in this context since the more nodes are encountered, the more codeword symbols can be recovered and the faster a rejoining node will be able to reconstruct the message. Since, clearly, nodes cannot store every message ever broadcast by the source, we only require them to store "pieces" for a finite number of messages.

Preliminaries.

Before describing our reliable message delivery mechanism, we first introduce some basic definitions. Recall that N is the set of all nodes in the network. At each time step $t \geq 1$, the source S broadcasts a message m_t. Let $\mathsf{ON}(t) \subseteq \mathsf{N}$ and $\mathsf{OFF}(t) \subseteq \mathsf{N}$ be the subsets of nodes that were online (i.e., connected) and offline (i.e., disconnected) at the time when message m_t was broadcast, respectively. We consider the following scenario. A node $\mathsf{v} \in \mathsf{OFF}(t)$ is offline at time t and therefore misses m_t. If at time $t' > t$, v rejoins the network and wishes to recover m_t, then we assume it will begin to encounter other nodes in the network, and in particular nodes that store symbols of c_t.

In addition, let μ be a finite value that determines for how many time steps each node in $\mathsf{ON}(t)$ will store a "piece". Intuitively, one can think of μ as determining the memory of the mechanism, or in other words, the number of time steps over which the system will guarantee that a message is recoverable. So if $\mathsf{v} \in \mathsf{OFF}(t)$ rejoins the network at some time $t' > t$, then it will only be able to recover m_t if $t' < t + \mu$.

Let $\mathcal{C} = (\mathsf{Encode}, \mathsf{Decode})$ be a (λ, ℓ, d)-erasure code. We assume that for any broadcast there are always at least λ nodes online (i.e., for $t \geq 1$, $|\mathsf{ON}(t)| \geq \lambda$). For ease of exposition, we also assume all messages have the same length, but note that our analysis can be easily adapted to the case where messages have different lengths. Our reliable message delivery mechanism works as follows. For all $t \geq 1$:

- S broadcasts $c_t = \mathsf{Encode}(m_t) = (s_{t,1}, \ldots, s_{t,\lambda})$.
- each node in $\mathsf{ON}(t)$ does the following:
  (i) if $t > \mu$ then erase $s_{t-\mu}$
  (ii) store $s_{t,y}$, where $y \xleftarrow{R} [1, \lambda]$.
- if at time $t' > t$, node $\mathsf{v} \in \mathsf{OFF}(t)$ wishes to recover m_t, it requests symbols of c_t from its neighbors until it has enough unique symbols to decode and recover m_t.
- after recovering m_t, v re-encodes it and stores symbol $s_{t,y}$, where $y \xleftarrow{R} [1, \lambda]$.

Since for each message m_t a node is required to store at most one symbol of the codeword c_t, the total storage per node is at most μ symbols.

Recovering a single message.

We now establish the expected number of encounters needed to recover a single missed message. In order to fully understand the trade-off (between storage and recovery time) that our mechanism provides, we derive this expectation as a function of the underlying erasure code's parameters. However, in Section 5 we fix a specific code and study the trade-off more precisely in the context of broadcast encryption. We note that the underlying model used in our analysis only approximates a MANET.

Let $\mathsf{Store}(m_t, t') \subseteq \mathsf{N}$ be the subset of nodes that store a symbol of $c_t = \mathsf{Encode}(m_t)$ at time $t' > t$, and let $\mathsf{Rec}(m_t, t') = \mathsf{OFF}(t) \cap \mathsf{Store}(m_t, t')$ be the subset of nodes that missed the broadcast of m_t (i.e., that were offline at time t), but that recovered m_t by time $t' > t$. So $\mathsf{Rec}(m_t, t')$ includes, for example, any nodes that missed m_t's original broadcast but that rejoined the network (at some time $\theta < t'$) and have already recovered m_t by time t'.

Given a node $\mathsf{v} \in \mathsf{OFF}(t)$ that rejoins the network at time $t' > t$ and wishes to recover m_t, we want to compute an upper bound on the expected number of nodes in $\mathsf{Store}(m_t, t')$ that v must encounter in order to recover m_t. By definition,

$$|\mathsf{Store}(m_t, t')| = |\mathsf{ON}(t)| + |\mathsf{Rec}(m_t, t')| \geq |\mathsf{ON}(t)|.$$

Since each node in $\mathsf{ON}(t)$ selects a symbol of $c_t = (s_{t,1}, \ldots, s_{t,\lambda})$ uniformly at random, then, on average, at time t each symbol will be stored by $\frac{|\mathsf{ON}(t)|}{\lambda}$ nodes. Furthermore, since each node in $\mathsf{ON}(t)$ only stores symbols for messages broadcast in the last μ time steps, at any time $t < t' < t + \mu$ we know that each symbol of c_t will be stored by at least $\frac{|\mathsf{ON}(t)|}{\lambda}$ nodes. Or in other words, each symbol will be almost equally dispersed throughout the network. In addition, since every node in $\mathsf{ON}(t)$ chooses which symbol of c_t to store uniformly at random, when encountering a node in $\mathsf{ON}(t)$, v will recover a uniformly distributed symbol of c_t. It follows that at any time $t < t' < t + \mu$, the number of nodes in $\mathsf{Store}(m_t, t')$ that v must encounter in order to recover at least $\lambda - d + 1$ unique symbols of c_t is at most the number of independent and uniformly distributed samples needed to recover at least $\lambda - d + 1$ unique elements from a set of λ elements. This reduces to the coupon collector's problem [18,17].

Let X be a random variable defined to be the latter and let X_i, where $0 \leq i \leq \lambda - d$, be the random variable defined to be the number of samples needed to select an element of c_t not sampled in any of the previous experiments X_j, where $0 \leq j \leq i-1$. It follows that $X = \sum_{i=0}^{\lambda-d} X_i$, and by the linearity of expectation $E[X] = \sum_{i=0}^{\lambda-d} E[X_i]$. Since each X_i is geometrically distributed with parameter $p_i = \frac{\lambda-i}{\lambda}$, we have $E[X_i] = \frac{1}{p_i}$ and

$$E[X] = \sum_{i=0}^{\lambda-d} \frac{\lambda}{\lambda - i} = \lambda \sum_{i=1}^{\lambda-d+1} \frac{1}{\lambda - i + 1} = \lambda \sum_{i=d}^{\lambda} \frac{1}{i} = \lambda(H_\lambda - H_{d-1}),$$

where H_n denotes the n^{th} harmonic number. Since $H_n \leq \ln n + \gamma$, where γ is a constant, we have

$$E[X] \leq \lambda \left(\ln \lambda - \ln(d-1)\right) \leq \lambda \ln\left(\frac{\lambda}{d-1}\right) \leq \frac{\ell}{\rho} \ln\left(\frac{\ell}{\rho \cdot (d-1)}\right). \tag{1}$$

So at any time $t < t' < t + \mu$, on average, $\mathsf{v} \in \mathsf{OFF}(t)$ will need to encounter $O(\ell \ln \ell)$ nodes in $\mathsf{Store}(m_t, t')$ in order to recover m_t.

Recovering multiple messages.

We can now establish a (loose) upper bound on the number of encounters needed to recover multiple messages. Let $\Delta = (m_{t_1}, \ldots, m_{t_r})$ be the set of messages missed by v and let t' be the time at which $\mathsf{v} \in \bigcap_{i=1}^{r} \mathsf{OFF}(t_i)$ rejoins the network. Assuming $t' < t_1 + \mu$, then we know that all the missed messages are recoverable at time t'; and that for each message m_{t_i}, where $1 \leq i \leq r$, v needs to encounter at most $\frac{\ell}{\rho} \ln\left(\frac{\ell}{\rho \cdot (d-1)}\right)$ nodes in $\mathsf{Store}(m_{t_i}, t')$. Therefore, it follows that v will need to encounter a total of at most

$$r \cdot \frac{\ell}{\rho} \ln\left(\frac{\ell}{\rho \cdot (d-1)}\right) \tag{2}$$

nodes in $\bigcup_{i=1}^{r} \mathsf{Store}(m_{t_i}, t')$. We note that this bound is not tight and that in practice the number of encounters will be smaller due to the fact that some nodes will be online during multiple time steps and, therefore, will store symbols for multiple messages. Consequently, v may recover symbols for multiple messages from a single encounter. And this will decrease the total number of encounters needed to recover the messages in Δ.

5 Reliable Stateful Broadcast Encryption

We now address the problem of secure content distribution in MANETs by using stateful broadcast encryption provisioned with a reliable message delivery mechanism. Stateful broadcast encryption schemes, such as LKH [27,26] or its more efficient variants [4,5,24], are appealing because of their reduced communication cost. However, in the context of MANETs it is likely that a node will miss key

updates due to network partitions or lossy links. To address this, we deliver the key updates using our reliable message delivery mechanism. This allows us to take advantage of the reduced communication cost offered by stateful broadcast encryption schemes in the context of highly dynamic mobile ad hoc wireless networks.

We use our reliable message delivery mechanism instantiated with a $(2\ell, \ell, \ell+1)$-erasure code to reliably deliver the broadcast encryption scheme's key updates to all the nodes in the network. Here, we consider a broadcast encryption scheme with key updates of bit length $k \cdot \log n$, where k is the size of the session key, and n is the number of nodes in the system. Several erasure codes and broadcast encryption schemes achieve these parameters including, for example, [21,15] and [4,5,24], respectively.

Recovering a single key update.

From Equation (1) in Section 4, we know that if instantiated with a $(2\ell, \ell, \ell+1)$-erasure code our delivery mechanism will guarantee that at any time $t < t' < t + \mu$, on average, node $\mathsf{v} \in \mathsf{OFF}(t)$ will have to encounter at most $2\ell \cdot \ln(2)$ nodes in $\mathsf{Store}(m_t, t')$ in order to recover the t^{th} key update. If σ is the size of a symbol (measured in bits), then each node will store $\mu \cdot \sigma$ bits. Furthermore, if $k \cdot \log(n)$ is the size of the key update (also in bits), then $\ell = \frac{k \cdot \log(n)}{\sigma}$ and

$$E[X] \leq \frac{2 \cdot k \cdot \log(n) \cdot \ln(2)}{\sigma} \leq 1.3863 \cdot \frac{k \cdot \log(n)}{\sigma},$$

where X is the random variable defined to be the number of nodes $\mathsf{v} \in \mathsf{OFF}(t)$ needs to encounter in order to recover the t^{th} key update.

Recovering multiple key updates.

Consider the case where a node v is offline during r revocation events. v will then miss the following r key updates broadcast by the source: $\Delta = (m_{t_1}, \ldots, m_{t_r})$. Each node will store $\mu \cdot \sigma$ bits and if $t' < t_1 + \mu$ then, by Equation (2) in Section 4, v will need to encounter a total of at most $1.3863 \cdot \frac{r \cdot k \cdot \log(n)}{\sigma}$ nodes in $\bigcup_{i=1}^{r} \mathsf{Store}(m_{t_i}, t')$.

Remark.

The data stored by non-authorized nodes does not allow them to decrypt the broadcast content. Consequently, the security of the system holds even if such non-authorized nodes are compromised.

6 Experimental Results

In this section we evaluate experimentally the performance of our message delivery mechanism.

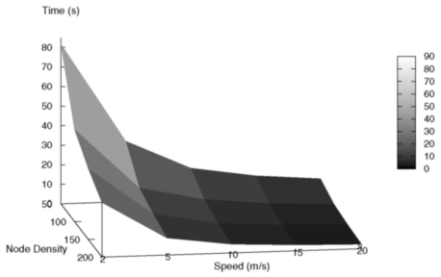

Fig. 1. Time required to encounter a single node after rejoining, as a function of node density and node speed.

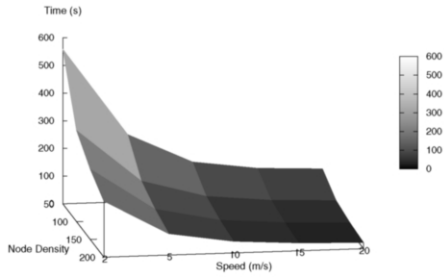

Fig. 2. Time required to encounter ten nodes after rejoining, as a function of node density and node speed.

Setup.

Nodes in the network were configured to use 802.11 radios with a nominal range of 250 meters (m). We randomly placed nodes within a 1500 by 1500 square meter area and varied the node density between 50 and 200 nodes per square kilometer. We also varied the nodes' maximum speed between 2 and 20 m/s (note a maximum speed of 10 m/s corresponds to an average speed of 5 m/s, since speeds are chosen uniformly between 0 and the maximum speed). Each data point in the figures of this section is the average result of 500 different random environments.

We used a random way-point mobility model, but incorporated changes to address concerns raised in [30] about the validity of the standard random way-point model. In particular, nodes select a speed uniformly between 10% and 90% of the given maximum speed to achieve a more steady mobility pattern and ensure that the average speed does not drop drastically over the course of the simulation. In addition, 300 seconds of mobility are generated before the start of the simulation so that nodes are already in motion. This allows the average speed and node distribution to stabilize before the simulation starts.

Since our mechanism allows one to trade storage per node for the number of encounters required, we measured the amount of time (in seconds) needed to encounter a specific number of nodes. We conducted two experiments, one to measure the amount of time needed for a node to encounter a single node, and the other to measure the amount of time needed to encounter ten nodes. In both experiments, we identify the nodes that are online at the time a node v gets disconnected. We then measure the time it takes v to encounter those nodes.

Figures 1 and 2 show the amount of time required for a rejoining node to encounter one and ten nodes, respectively. We observe that as the node density increases, the time required to meet a given number of nodes decreases. The same holds as the maximum speed of the nodes increases. We note that for high node densities the time values become very reasonable. In fact, for a network density of 200 nodes per square kilometer and a low maximum node speed of 5 m/s, the first node is encountered in 8 seconds and the tenth node is encountered after 77 seconds.

Remark.

During our experiments, we noticed that while a node v is disconnected, the number of connected nodes typically remains very high. Thus, as noted in Section 5, when v rejoins the network, it should be able to recover several symbols from each encounter. We therefore expect the recovery time for multiple messages to be relatively close to the recovery time for a single message.

An extension for low node densities.

We have seen that our mechanism achieves the best results when used in networks with high node densities. This is due to the fact that a rejoining node recovers missed messages only through direct encounters with other nodes. We propose the following modification in order to increase the speed of message recovery: when a rejoining node v encounters the first node, it not only recovers symbols from this node, but asks the node to retrieve other symbols from its own neighbors. The retrieval of symbols can thus continue recursively in an expanding ring fashion until node v recovers all the symbols it needs. This will not significantly affect the overall network communication, since the additional traffic will be localized in v's neighborhood. However, by leveraging the network, a rejoining node can greatly reduce its message recovery time, even for networks with relatively low node densities.

7 Acknowledgements

The authors would like to thank the anonymous reviewers for helpful comments. The second author was supported by a Bell Labs Graduate Research Fellowship.

References

[1] Attrapadung, N., K. Kobara and H. Imai, *Broadcast encryption with short keys and transmissions*, in: *DRM '03: Proceedings of the 3rd ACM workshop on Digital rights management*, 2003, pp. 55–66.

[2] Boneh, D., C. Gentry and B. Waters, *Collusion resistant broadcast encryption with short ciphertexts and private keys*, in: *Advances in Cryptology - Crypto '05*, 2005.

[3] Byers, J., M. Luby, M. Mitzenmacher and A. Rege, *A digital fountain approach to reliable distribution of bulk data*, in: *Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 1998.

[4] Canetti, R., J. A. Garay, G. Itkis, D. Micciancio, M. Naor and B. Pinkas, *Multicast security: A taxonomy and some efficient constructions*, in: *INFOCOM*, 1999, pp. 708–716.

[5] Canetti, R., T. Malkin and K. Nissim, *Efficient communication-storage tradeoffs for multicast encryption*, in: *EUROCRYPT*, 1999, pp. 459–474.

[6] Chandra, R., V. Ramasubramanian and K. Birman, *Anonymous gossip: Improving multicast reliability in mobile ad-hoc networks*, in: *ICDCS*, 2001, pp. 275–283.

[7] Eschenauer, L. and V. Gligor, *A key-management scheme for distributed sensor networks*, in: *Proceedings of the 9th ACM conference on Computer and Communications Security*, 2002, pp. 41–47.

[8] Fiat, A. and M. Naor, *Broadcast encryption*, in: D. R. Stinson, editor, *Advances in Cryptology - Crypto '94*, Lecture Notes in Computer Science **773** (1994), pp. 480–491.

[9] Gerla, M., S. Lee and W. Su, *On-demand multicast routing protocol (ODMRP) for ad hoc networks*, in: *Internet Draft: draft-ietf-odmrp-02.txt*, 2000.

[10] Goodrich, M. T., J. Z. Sun and R. Tamassia, *Efficient tree-based revocation in groups of low-state devices.*, in: *Proceedings of Crypto '04*, LNCS **3152** (2004), pp. 511–527.

[11] Halevy, D. and A. Shamir, *The LSD broadcast encryption scheme*, in: *Advances in Cryptology - Crypto '02*, Lecture Notes in Computer Science **2442** (2002), pp. 47–60.

[12] Jho, N.-S., J. Y. Hwang, J. H. Cheon, M.-H. Kim, D. H. Lee and E. S. Yoo, *One-way chain based broadcast encryption schemes.*, in: *EUROCRYPT 2005*, LNCS **3494**, 2005, pp. 559–574.

[13] Karlof, C., N. Sastry, Y.Li, A. Perrig and J. D. Tygar, *Distillation codes and applications to DoS resistant multicast authentication.*, in: *Proceedings of NDSS '04*, 2004.

[14] Liu, D., P. Ning and K. Sun, *Efficient self-healing group key distribution with revocation capability*, in: *Proc. of the 10th ACM conference on Computer and Communications Security*, 2003, pp. 231–240.

[15] Luby, M., M. Mitzenmacher, M. Shokrollahi and D. Spielman, *Efficient erasure correcting codes*, Transactions on Information Theory **47** (2001), pp. 569–584.

[16] Luo, J., P. Eugster and J.-P. Hubaux, *Route driven gossip: Probabilistic reliable multicast in ad hoc networks*, in: *Proceedings of INFOCOM 2003*, 2003.

[17] Mitzenmacher, M. and E. Upfal, "Probability and Computing : Randomized Algorithms and Probabilistic Analysis," Cambridge University Press, 2005 .

[18] Motwani, R. and P. Raghavan, "Randomized Algorithms," Cambridge University Press, 1995 .

[19] Naor, D., M. Naor and J. Lotspiech, *Revocation and tracing schemes for stateless receivers*, in: *Advances in Cryptology - Crypto '01*, LNCS **2139** (2001), pp. 41–62.

[20] Pinkas, B., *Efficient state updates for key management.*, Proceedings of the IEEE, Special Issue on Enabling Technologies for Digital Rights Management **92** (2004), pp. 910–917.

[21] Reed, I. and G. Solomon, *Polynomial codes over certain finite fields*, in: *Journal of the Society for Industrial and Applied Mathematics*, 1960.

[22] Royer, E. and C. Perkins, *Multicast operation of the ad-hoc on-demand distance vector routing protocol*, in: *Proceedings of MobiCom '99* (1999), pp. 207–218.

[23] Setia, S., S. Zhu and S. Jajodia, *A comparitive performance analysis of reliable group rekey transport protocols for secure multicast*, Performance Evaluation **49** (2002), pp. 21–41.

[24] Sherman, A. T. and D. A. McGrew, *Key establishment in large dynamic groups using one-way function trees*, IEEE Trans. Softw. Eng. **29** (2003), pp. 444–458.

[25] Staddon, J., S. K. Miner, M. Franklin, D. Balfanz, M. Malkin and D. Dean, *Self-healing key distribution with revocation.*, in: *IEEE Symposium on Security and Privacy*, 2002, pp. 241–257.

[26] Wallner, D., E. Harder and R. Agee, *Key management for multicast: Issues and architectures*, in: *RFC 2627*, 1999.

[27] Wong, C., M. Gouda and S. Lam, *Secure group communication using key graphs*, in: *ACM SIGCOMM '98*, 1998, pp. 68–79.

[28] Wong, C. and S. Lam, *Keystone: A group key management service*, in: *International Conference on Telecommunications, ICT 2000*, 2000.

[29] Yang, Y., X. Li, X. Zhang and S. Lam, *Reliable group rekeying: a performance analysis*, in: *Proceedings of ACM SIGCOMM '01*, 2001, pp. 27–38.

[30] Yoon, J., M. Liu and B. Noble, *Random waypoint considered harmful.*, in: *INFOCOM '03*, 2003.

[31] Zhang, X., S. Lam, D.-Y. Lee and Y. Yang, *Protocol design for scalable and reliable group rekeying*, in: *Proceedings of SPIE Conference on Sealability and Traffic Control in IP Networks*, 2001.

[32] Zhu, S., S. Setia, S. Xu and S. Jajodia, *GKMPAN: An efficient group rekeying scheme for secure multicast in ad-hoc networks*, in: *Proceedings of the First Annual International Conference on Mobile and Ubiquitos Systems (Mobiquitos '04)*, 2004.

[33] Zhu, S., S. Setia, S. Xu and S. Jajodia, *GKMPAN: An efficient group rekeying scheme for secure multicast in adhoc networks - technical report ISE-TR* (2004).

[34] Zhu, S., S. Xu, S. Setia and S. Jajodia, *Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach*, in: *Proceedings of ICNP '03*, 2003.

	message size	storage size at the receiver	key update size
stateful	O(1)	$O(\log n)$	$O(\log n)$
stateless	$O(r)$	$O(1)$	not required

Table A.1
Parameters for stateful and stateless broadcast encryption schemes: n is the total number of users in the system and r is the number of revoked users

A Communication cost comparison for stateful and stateless broadcast encryption schemes

Let N be the set of users, $n = |\mathsf{N}|$ be the total number of users, and $r = |\mathsf{R}| = n - |\mathsf{G}|$ be the number of revoked users. We consider the three following parameters of a broadcast encryption scheme: the size of a broadcast message, the required storage at the receiver, and the size of a key update (in the case of stateful schemes). Table A.1 presents the parameters of the most efficient stateful [4,5,24] and stateless schemes [11,1,10,12] [3].

We evaluate the overall message transmission cost for both stateful and stateless schemes. Let us consider a period of time in which r users are revoked and d messages, each of size l, are broadcast between two consecutive revocations. Note that when a join or a revocation occur, stateful schemes require the source to broadcast a key update of size $O(k \cdot \log n)$, where k is the size of the session key. Stateless schemes, on the other hand, do not require such updates and instead add a header of size $O(r)$ to each subsequent message. The total transmission cost at the source for a stateful scheme is then $O(r \cdot d \cdot l + r \cdot \log n \cdot k)$ bits. For stateless schemes, the cost is

$$\sum_{i=0}^{r-1} m \cdot (l + c \cdot i) = O(r \cdot m \cdot l + d \cdot r^2 \cdot k).$$

where c is some arbitrary constant.

Thus, when $r = \Omega(\log n)$, stateful schemes become more efficient in terms of communication than stateless schemes. This has important practical implications since, for example, if $n = 2^{10}$ a stateful scheme will become more efficient after $r = 10$ revocations. In a dynamic network of 1024 users, it can be easily assumed that more than 10 users will be revoked.

[3] Recently a stateless scheme has been proposed that achieves constant broadcast size [2]. However, we do not consider it here since its applicability for content distribution in MANETS is limited, as it requires $O(n)$ storage at the receiver.