Energy-Efficient, Secure Group Key Agreement for Ad Hoc Networks

Thomas R. Halford
TrellisWare Technologies, Inc.
16516 Via Esprillo, Suite 300
San Diego, CA 92127
Email: thalford@trellisware.com

Thomas A. Courtade
Center for Science of Information
Stanford University
Stanford, CA 94305
Email: courtade@stanford.edu

Keith M. Chugg
Ming Hsei Dept. of Electrical Engineering
University of Southern California
Los Angeles, CA 90089
Email: chugg@usc.edu

***Abstract*—Public key cryptography is well-suited to ad hoc networks as it requires no *a priori* secure key distribution mechanism. Recent advances in lattice-based cryptography are enabling the use of public key algorithms (PKAs) in low-power devices. Unfortunately, while many ad hoc networking applications are dominated by multicast traffic, PKAs are inherently unicast: public/private key pairs are generated by data destinations. To fully realize public key cryptography in ad hoc networks, low-power PKAs must therefore be augmented with energy-efficient mechanisms for secure group key establishment.**

Motivated by recent results on information theoretic secrecy, we present a protocol that generates keys for t-sized multicast groups with $O(\log_b t)$ transmissions, where b is a parameter that enables trades between energy efficiency and security. Extensions of this protocol that exploit network topology side information and which permit multilevel security are also described. A cryptosystem employing an energy-efficient PKA and the protocols presented herein could provide the benefits of public key cryptography – i.e., dynamic ad hoc network support – with features currently found only in symmetric systems.

I. Introduction

Security is a paramount concern in wireless networks [1]. The symmetric key algorithms used in static networks such as AES-256 rely on secure, *a priori* key distribution and are therefore ill-suited to dynamic, ad hoc networked applications. Public key algorithms (PKAs) have traditionally been seen as incompatible with such ad hoc networks for two reasons:

1) Public key algorithms are much more computationally complex than symmetric key algorithms (cf., [1]).
2) PKAs are tailored for unicast transmission – i.e., the public encryption key and private decryption key are generated by a unique data destination – yet ad hoc networks are often dominated by multicast traffic [2, 3].

Recent advances in lattice-based cryptography (e.g., the NTRU cryptosystem) are paving the way for the development of energy-efficient PKAs (cf., [4, 5]); the second issue, however, has received considerably less attention in the literature.

Public key cryptosystems can support multicast traffic by replacing t-destination multicast sessions with t parallel unicast sessions. This naïve approach fails to capture the energy savings afforded by multicast tree routing [6] or controlled broadcasting in wireless networks [7]. It is more energy-efficient to instead have the destination nodes securely establish a *group* key pair, thereby allowing information to be encrypted by the source and decrypted by all destination nodes simultaneously. Traditional group key *distribution* protocols, wherein one destination generates and distributes a group key pair to each of the other $t-1$ destinations in turn, require $O(t)$ transmissions[1] [8, 9]. Existing group key *agreement* protocols, wherein the destinations cooperatively establish a key via messaging, also require $O(t)$ transmissions (cf., [10]). In the present work, group key establishment protocols that require a sub-linear number of transmissions are sought.

Our work on energy-efficient group key agreement was motivated by recent results on the universal recovery problem [11], which we review in Section II. In Section III we present a protocol that enables t multicast destinations to securely agree on a common group key pair using $O(\log_b t)$ transmissions, where b is a parameter that enables trades between energy efficiency and security. The transmissions are binary sums of session keys, which are derived from master keys that are randomly distributed amongst the nodes. Our protocol is secure in the information-theoretic sense [12]: an eavesdropper that observes all transmissions can do no better than randomly guessing the established private key. The protocol description in Section III assumes that nodes are loaded with master keys prior to deployment; Section IV describes an extension in which master keys are generated and exchanged on-the-fly.

The number of transmitted messages is a useful metric for energy efficiency in one-hop networks and in ad hoc networks that employ flooding-based protocol stacks (e.g., [7, 13–15]). In Section V, a protocol is presented that is optimized for use in networks that have a controlled broadcasting capability [7]. In particular, we show that network topology side information can be used to reduce the total number of transmissions plus relays required for group key agreement, while enhancing security against undetected compromised nodes. Extensions to support hierarchical security policies are also described. Finally, the security of the proposed protocol in the context of an NTRU-based cryptosystem is analyzed in Section VI.

[1]We say that a function $f(n) = O(g(n))$ if the exists constants n_0 and c such that $f(n) \leq cg(n)$ for all values of $n > n_0$.

II. Information-Theoretic Secrecy Generation

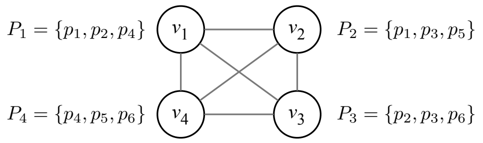
A. Universal Recovery

Suppose that k packets $P = \{p_1, \ldots, p_k\}$ are distributed amongst n nodes. Let $P_i \subseteq P$ be the subset of packets initially available at node i such that $\bigcup_{i=1}^{n} P_i = P$. Note that overlap is permitted between these sets (i.e., we do not require $P_i \cap P_j = \emptyset$). The goal of the universal recovery problem is to distribute all k packets to all nodes with a minimum number of transmissions. In [11], it was shown that the minimum number of transmissions required for universal recovery

$$M\left(\mathcal{G}, \{P_i\}_{i=1}^{n}\right),$$

which is a function of the network topology $\mathcal{G}$ and initial packet distribution $\{P_i\}_{i=1}^{n}$, can be computed in linear time using submodular optimization [16] and network coding [17] if the network is fully connected[2]. In general, not all k packets need to be transmitted network-wide for universal recovery. As a consequence of [12], the difference between k and $M\left(\mathcal{G}, \{P_i\}_{i=1}^{n}\right)$ can be exploited for secrecy generation (cf., [11]).

B. Secrecy Generation by Example



$P_1 = \{p_1, p_2, p_4\}$
v_1
v_2
$P_2 = \{p_1, p_3, p_5\}$
$P_4 = \{p_4, p_5, p_6\}$
v_4
v_3
$P_3 = \{p_2, p_3, p_6\}$

Fig. 1. Universal recovery in this network requires 4 transmissions.

As a concrete example of universal recovery, consider the network illustrated in Figure 1, where $k = 6$ 256-bit packets are distributed amongst $n = 4$ nodes. Universal recovery of $P = \{p_1, \ldots, p_6\}$ can be achieved via four transmissions:

1) Node v_1 transmits the binary sum $t_1 = p_1 + p_2$.
    - Node v_2 recovers $t_1 + p_1 = p_2$.
    - Node v_3 recovers $t_1 + p_2 = p_1$.
2) Node v_2 transmits the binary sum $t_2 = p_3 + p_5$.
    - Node v_3 recovers $t_2 + p_3 = p_5$.
    - Node v_4 recovers $t_2 + p_5 = p_3$.
3) Node v_3 transmits the binary sum $t_3 = p_2 + p_6$.
    - Node v_1 recovers $t_3 + p_2 = p_6$.
    - Node v_2 recovers $t_3 + p_2 = p_6$.
    - Node v_4 recovers $t_3 + p_6 = p_2$ and $t_1 + p_2 = p_1$.
4) Node v_1 transmits the binary sum $t_4 = p_4 + p_5$.
    - Node v_1 recovers $t_4 + p_4 = p_5$ and $t_2 + p_5 = p_3$.
    - Node v_2 recovers $t_4 + p_5 = p_4$.
    - Node v_3 recovers $t_4 + p_5 = p_4$.

Since only $M = 4$ transmissions were required to recover the 6 packets in this example, the results of [11] indicate that 2 packets of common randomness k_1 and k_2 can be generated. In order for k_1 and k_2 to be secure from an eavesdropper that observes the transmissions $t_1, \ldots, t_4$ in the information theoretic sense, we require that two conditions hold [12]:

- Perfect Secrecy Condition: The mutual information between k_1 and k_2 and the set of all transmissions is zero.
- Uniformity Condition: The packets $p_1, \ldots, p_4$ are uniformly and identically distributed among the set of all possible 256-bit strings.

In our example, perfect secrecy holds if $k_1 = p_1$ and $k_2 = p_3$.

C. Partial Recovery

Since group key agreement requires that only a single shared secret packet be established, universal recovery is not necessary. In the example above, the packet p_1 can be shared by two transmissions alone (e.g., $t_1 = p_1 + p_2$ and $t_2 = p_1 + p_4$). As we will see in the next section, secrecy generation by *partial recovery* allows us to circumvent the computationally expensive sub-modular function optimization and network code construction required for universal recovery.

III. Group Key Agreement in Static Networks

A. Secret Agreement via Partial Recovery

Before describing our protocol, we first define a procedure for establishing a secret packet among t nodes. Let $D = \{d_1, \ldots, d_t\}$ be a set of nodes. The set of packets incident at node $d_i \in D$ is denoted P_{d_i} and $P_D = \cup_{i=1}^{t} P_{d_i}$ is the set of packets incident at any node in D. The *occupancy sets*,

$$\mathcal{O} = \{O_j\}_{j \in P_D}, \tag{1}$$

where $O_j \subseteq D$, may be computed at all nodes such that

$$d_i \in O_j \iff p_j \in P_{d_i}. \tag{2}$$

That is to say, the occupancy set O_j is the subset of destination nodes possessing p_j packet initially. We assume[3] that the initial packet distribution is such that for every node pair $(d_i, d_j) \in D \times D$, there exists at least one packet $p_k \in P_D$ such that $\{d_i, d_j\} \subseteq O_k$.

In order to define a series of transmissions used to establish a common secret packet, the nodes first compute a set cover solution $\mathcal{C}$ of D from $\mathcal{O}$ via the standard greedy heuristic.

```
**Input**: $D = \{d_1, \ldots, d_t\}$, $\mathcal{O} = \{O_j\}_{j \in P_D}$.
**Output**: $\mathcal{C} \subseteq \mathcal{O}$.
$U \leftarrow D$, $\mathcal{C} \leftarrow \emptyset$
**while** $U \neq \emptyset$ **do**
    select an $S \in \mathcal{O}$ that minimizes $|S \cap U|$
    $U \leftarrow U \setminus S$, $\mathcal{C} \leftarrow \mathcal{C} \cup \{S\}$
**end**
```

Algorithm 1: Greedy, linear-time approximation of set cover.

[2]Although the networks considered in this work need not be fully connected at the physical layer, cryptographic protocols are often implemented at higher layers that see an abstracted, one-hop topology. Furthermore, the largest-scale ad hoc network deployments to-date use flooding-based stacks [7].

[3]Note that we also assume that nodes have global knowledge of the packet distribution. This can be achieved by establishing a deterministic relationship between node addresses and unique identifiers on the packets (e.g., via a hash of the node address and some unique packet identifier).

This set cover solution can then be used to define a *transmission schedule* $\mathcal{T}$ that establishes the packet p_{j_0} as a shared secret among the t destinations. In particular, the transmissions defined by the outputs of Algorithm 2 are of the form: node d_{i_k} transmits the sum $p_{j_0} + p_{j_k}$ (for $k \in [1, r]$).

```
Input: $\mathcal{O} = \{O_j\}_{j \in P_D}$, $\mathcal{C} \subseteq \mathcal{O}$.
Output: $j_0$, $\mathcal{T} = \{(i_1, j_1), \ldots, (i_r, j_r)\}$.
$j_0 \leftarrow$ packet index corresponding to first element of $\mathcal{C}$
$i_1 \leftarrow$ index of first element of $O_{j_0}$,
$\mathcal{C} \leftarrow \mathcal{C} \setminus O_{j_0}$, $\mathcal{T} \leftarrow \emptyset$, $k \leftarrow 1$
while $C \neq \emptyset$ do
    if $\exists\ O_l \in \mathcal{C}$ such that $|O_{j_{k-1}} \cap O_l| > 0$ then
        $j_k \leftarrow l$
        $i_{k+1} \leftarrow$ index of first element of $O_{j_{k-1}} \cap O_{j_k}$
        $\mathcal{C} \leftarrow \mathcal{C} \setminus O_{j_k}$
    else
        $i_{k+1} \leftarrow$ index of first element of $\mathcal{C}$'s first element
        $j_k \leftarrow$ packet such that $\{d_{i_k}, d_{i_{k+1}}\} \subseteq O_{j_k}$
    end
    $\mathcal{T} \leftarrow \mathcal{T} \cup (i_k, j_k)$
    $k \leftarrow k + 1$
end
```

Algorithm 2: Greedy heuristic for defining a transmission schedule from the set cover solution.

As an example, consider the 5-node network illustrated in Figure 2. Here, Algorithm 1 returns $\mathcal{C} = \{O_1, O_{10}\}$ (the only size two cover of D in $\mathcal{O}$). Algorithm 2 thus initializes $j_0 = 1$ and $i_1 = 1$. Since $O_1 \cap O_{10} = \emptyset$ and d_4 is the first element of O_{10}, $i_2 = 4$ and $j_1 = 3$; since $O_3 \cap O_{10} = \{d_4\}$, $j_2 = 10$ and $i_3 = 5$. The final outputs of Algorithm 2 are thus $j_0 = 1$ and

$$\mathcal{T} = \{(1, 3), (4, 10)\}, \tag{3}$$

which define two transmissions. Specifically, node d_1 transmits the sum $t_1 = p_1 + p_3$, which is used by node d_4 to obtain $p_1 = t_1 + p_3$. Node d_4 then transmits the sum $t_2 = p_1 + p_{10}$, which is used by node d_5 to obtain $p_1 = t_2 + p_{10}$.

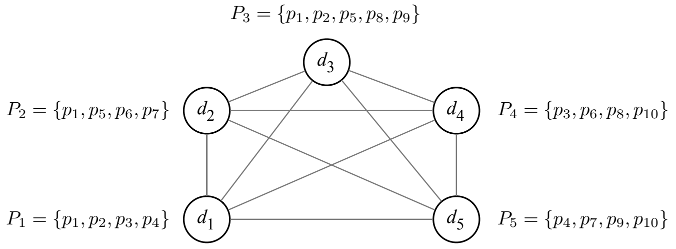

$P_3 = \{p_1, p_2, p_5, p_8, p_9\}$
d_3
$P_2 = \{p_1, p_5, p_6, p_7\}$
d_2
d_4
$P_4 = \{p_3, p_6, p_8, p_{10}\}$
$P_1 = \{p_1, p_2, p_3, p_4\}$
d_1
d_5
$P_5 = \{p_4, p_7, p_9, p_{10}\}$

Fig. 2. Generating a common secret packet among 5 nodes.

B. Protocol Specification

Given an n node network, a total of $\binom{n}{2}$ *pairwise* cryptographic master keys are generated and distributed to the network nodes such that each pairwise key is shared by exactly two nodes. Additionally, a total of $\lceil f(n)/\beta \rceil$ *doping* master keys are generated and randomly distributed throughout the network such that the probability that a given doping key is loaded on a given node is β and the average number of doping keys per node is approximately $f(n)$ (where the function $f(n)$ is a parameter that impacts memory requirements). Intuitively, the doping keys will be used in our protocol to control the size of the set cover solution returned by Algorithm 1, while the pairwise keys are used to ensure that for every node pair (d_i, d_j), we can *efficiently* find a packet p_k such that $\{d_i, d_j\} \subseteq O_k$ in Algorithm 2 (i.e., in linear time). Again, we assume global knowledge of the doping key distribution so that the occupancy sets can be determined at each node.

Suppose that node s wishes to initiate a secure multicast session to t destination nodes $D = \{d_1, \ldots, d_t\}$. This can be done using the following protocol.

G-1: A message containing the destination addresses and a unique session identifier u is transmitted by s over a non-secure wireless channel.

G-2: Upon reception of the secure multicast request, every destination node:

(a) Computes the occupancy sets for all its locally stored master keys (both pairwise and doping) with respect to the destination set $D = \{d_1, \ldots, d_t\}$.

(b) Uses Algorithms 1 and 2 to determine the index of the secret packet j_0 and the transmission schedule parameters $\mathcal{T} = \{(i_1, j_1), \ldots, (i_r, j_r)\}$.

G-3: The destination nodes generate $r + 1$ session keys from the $r + 1$ master keys identified in Step G-2(b) via

$$s_k \leftarrow \phi(p_{j_k}, u) \ \text{ for } 0 \leq k \leq r,$$

where $\phi()$ is a cryptographically secure pseudorandom function (cf., [18, 19]). Node d_i only generates session key s_k if it is loaded with the corresponding master key.

G-4: If $t = 2$, then nodes d_1 and d_2 already share s_0. Otherwise, s_0 is obtained by the other destinations via r non-secure transmissions of the form: node d_{i_k} transmits the sum $s_0 + s_k$ (for $k \in [1, r]$). These transmissions reveal no information about s_0 nor any about the master keys, provided that $\phi()$ is properly designed.

G-5: The destinations independently use s_0 to derive a group key pair for a chosen public key algorithm. Section VI describes how this can be done for NTRU.

G-6: Node d_1 transmits the group public key to the source node s and the secure multicast session can begin.

C. Protocol Discussion

1) Energy Efficiency: Figure 3 compares the number of transmissions required to generate a common secret packet with the protocol defined in Section III-B in a 50-node network for varying levels of β when $f(n) = \log_2(n)$. When only pairwise packets are used, exactly $t - 2$ transmissions are required to generate the common secret packet. When doping keys are added, this $O(t)$ behavior is replaced by a logarithmic dependence on t. Observe that as β increases, the average number of transmissions required to generate a common

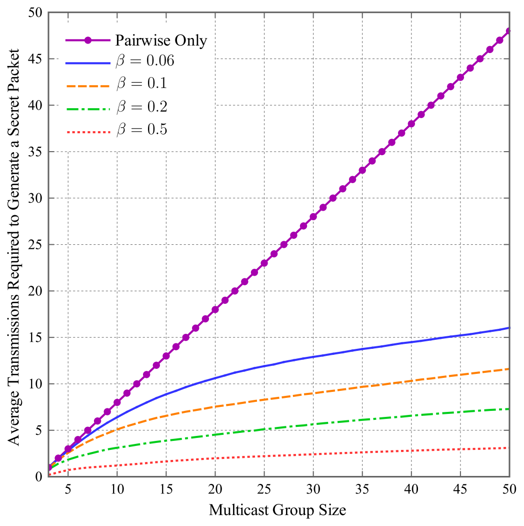


Fig. 3. Number of transmissions required to generate a common secret packet as a function of multicast group size in a 50-node network.

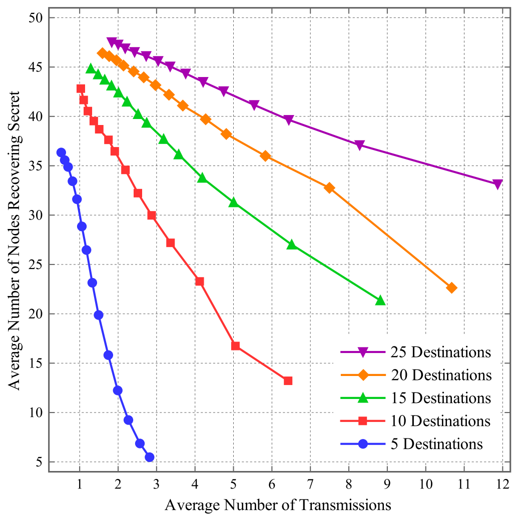


Fig. 4. Average number of nodes that can recover a common packet as a function of the average number of transmissions required to generate it.

secret packet decreases. This is consistent with intuition: as β increases, the occupancy set sizes for the doping keys also increase, thus reducing the size of the set cover solution. Formally, results on the random set covering problem [20] can be used to show that the number of transmissions required to generate a common secret packet depends on β and the multicast group size t as $O(\log_b t)$, where $b = 1/(1-\beta)$.

2) Security Against Undetected Compromised Nodes: In the protocol described in Section III-B, the use of doping keys enables nodes outside of the intended destination to recover the secret key. For example, any node possessing a given master key p_{j_k} (and therefore the corresponding session key s_k) can recover the secret key s_0 from the non-secure transmission of $s_0 + s_k$. As β increases, each doping key is distributed to more nodes and we expect the probability that an undetected compromised node can recover the secret key to increase. The resulting tradeoff between efficiency and security versus undetected compromised nodes is illustrated in Figure 4.

IV. Group Key Agreement in Dynamic Networks

The protocol described in Section III-B is energy-efficient but requires that nodes be loaded with pairwise and doping keys prior to deployment. This is inconsistent with our stated goal of developing an energy-efficient public key cryptosystem for dynamic, ad hoc networks. In this section, we describe a protocol suitable for use in such networks. The pairwise master keys are generated and exchanged on-the-fly while the doping keys propagate through the network via an epidemic model inspired by distributed database maintenance algorithms [21].

A. Protocol Definition

We assume that every node is fielded with the ability to generate cryptographic keys (cf., [22]). Node i initially generates and stores f_i random doping keys where:

$$\Pr\{f_i = m\} = \binom{n}{m}\alpha^m(1-\alpha)^{n-m} \text{ and } \alpha = \frac{f(n)}{\beta n^2}. \quad (4)$$

Observe that, on average, node i stores $\mathbb{E}[f_i] = f(n)/\beta n$ keys initially and there are a total of $f(n)/\beta$ keys in the network. Node i is initialized to the susceptible state with respect to the doping keys generated at all other nodes. Returning to the protocol described in Section III-B, dynamic operation can be achieved by inserting a key exchange procedure between steps G-1 and G-2. Specifically, for every pair of destination nodes $(d_i, d_j) \in D \times D$ such that $i < j$:

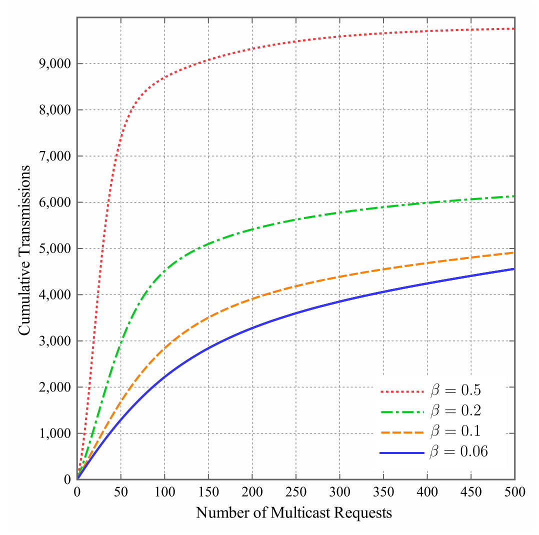
(a) Pairwise Key Generation and Exchange: If this is the first time that d_i and d_j have been involved in the same multicast destination set, d_i generates a pairwise master key and transmits it to d_j via a secure unicast session.

(b) Doping Key Exchange: Let d_i and d_j store the doping key sets P_i and P_j respectively.

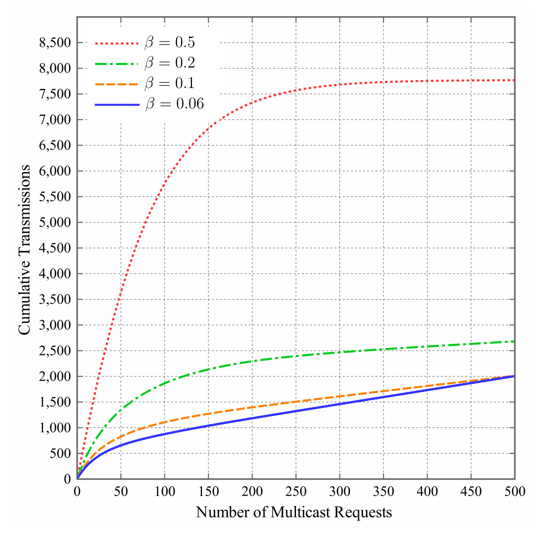
   i. For every $p_k \in P_i$ to which d_j is susceptible, d_j becomes infected by p_k with probability β and immune with probability $1-\beta$. If d_j is infected by m_j packets, then an m_j-transmission secure unicast session is required for doping key exchange.

   ii. For every $p_l \in P_j$ to which node d_i is susceptible, d_i becomes infected by p_l with probability β and immune with probability $1-\beta$. If d_i is infected by m_i packets, then an m_i-transmission secure unicast session is required for doping key exchange.

At steady state, all pairwise master keys will be generated and exchanged, each node will be infected with an average of $f(n)$ doping master keys, and each doping key will be incident on a given node with a probability that approaches β. Observe that this dynamic protocol extension can be readily extended to support finite key lifetimes for increased security.

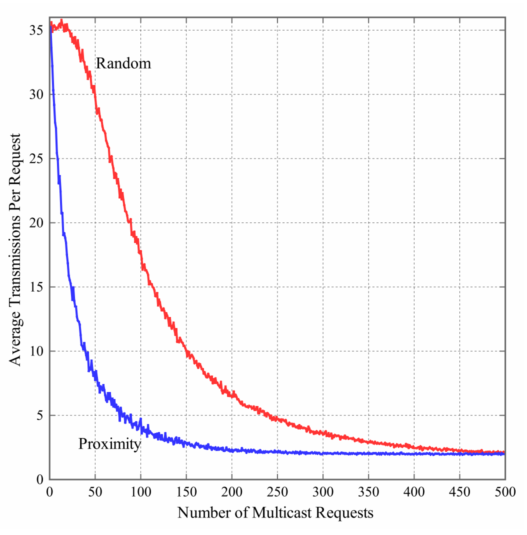


(a)

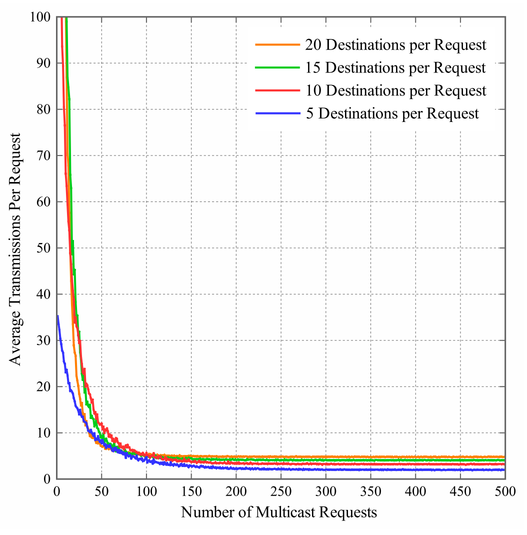


(b)

Fig. 5. Cumulative number of transmissions required over time to establish group key pairs under the (a) random and (b) proximity multicast models.



(a)



(b)

Fig. 6. Average transmissions per request over time in (a) when $t = 5$ and $\beta = 0.1$ and (b) the proximity model alone for varying values of t.

B. Protocol Discussion

1) Multicast Models: In order to study how our dynamic group key agreement protocol behaves over time, we consider two different models for multicast traffic:

- Random Model: The source and destinations for each multicast session are chosen randomly and independently.
- Proximity Model: Nodes are distributed randomly in the unit square. The source node s for each multicast session is chosen randomly and independently. The t closest nodes to s define the destination set.

The random model is somewhat unrealistic – group communications are structured in practice – but it provides a useful worst case for our study; the proximity model is more representative of practical military ad hoc network use cases.

2) Convergence to Steady State: Figure 5 illustrates how the cumulative number of transmissions evolves over time in a 50-node network under the two multicast models. The multicast group size is fixed to 5 and $f(n) = \log_2(n)$. Observe that increasing the infection probability from $\beta = 0.06$ to 0.5 decreases the number of transmissions required to generate group key pairs at steady state but increases overhead due to key exchange. Setting $\beta = 0.1$ appears to offer a good trade between the steady-state and transient behavior. Figure 6(a) compares the speed of convergence under the random and proximity models; as expected, the latter yields faster convergence. Finally, Figure 6(b) illustrates that the convergence rate under the proximity model does not monotonically decrease with increasing t owing to the increasing overlap between multicast destination groups.

V. Protocol Extensions

A. Topology-Aware Group Key Agreement

Recall that the transmission schedule determined via Algorithms 1 and 2 is of the form:

$$\mathcal{T} = \{(i_1, j_1), \ldots, (i_r, j_r)\}\,, \tag{5}$$

where node d_{i_k} broadcasts the sum of the secret packet p_{j_0} and p_{j_k}. The protocols discussed in Sections III and IV sought to minimize the number of network-wide broadcasts r. This is a useful metric for energy efficiency in one-hop networks and in networks that employ flooding-based protocol stacks (e.g., [7, 13–15]). However, in multi-hop networks with traditional link-based protocol stacks – and in flooing-based networks equipped with a controlled broadcasting capability [7] – we must also account for the energy costs of relaying. In this section, we therefore study transmission schedules of the form:

$$\mathcal{T} = \{(i_1, j_1, h_1), \ldots, (i_r, j_r, h_r)\}\,, \tag{6}$$

where destination d_{i_k} transmits $p_{j_0} + p_{j_k}$ to all nodes within a h_k-hop radius. Our goal is to minimize the sum of transmissions and relays required for group key agreement: $\sum_{k=1}^{r} h_k$.

B. Topologically-Aware Secret Agreement

Let $h(s, d_i)$ denote the number of hops separating a source node s and a destination node d_i and let

$$h(s, D) = \max_{d_i \in D} h(s, d_i) \tag{7}$$

denote the number of hops required to transmit a message from s to all nodes in a set of destinations D. All other notation is adopted from Section III-A. Algorithm 3 combines the set cover and transmission schedule computation of Algorithms 1 and 2, respectively, into a single topologically-aware heuristic for group key agreement that was motivated by the standard approximation algorithm for weighted set cover.

```
**Input**: $D$, $\mathcal{O} = \{O_j\}_{j \in P_D}$.
**Output**: $j_0$, $\mathcal{T} = \{(i_1, j_1, h_1), \ldots, (i_r, j_r h_r)\}$.
$j_0 \leftarrow$ packet index corresponding to largest set in $\mathcal{O}$
$S \leftarrow O_{j_0}$, $\mathcal{F} \leftarrow \mathcal{O} \setminus \{O_{j_0}\}$, $k \leftarrow 1$
**while** $S \neq D$ **do**
    $(i_k, j_k) \leftarrow$ indices of occupancy set $O_{j_k} \in \mathcal{F}$ and
                 transmitter $d_{i_k} \in O_{j_k} \cap S$ that maximize
                 the number of nodes that will obtain
                 the secret *per hop*:
                     $\dfrac{|O_{j_k} \setminus S|}{h(d_{i_k}, O_{j_k} \setminus S)}$
    $h_k \leftarrow h(d_{i_k}, O_{j_k} \setminus S)$
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{i_k, j_k, h_k\}$
    $S \leftarrow S \cup O_{j_k}$, $\mathcal{F} \leftarrow \mathcal{F} \setminus \{O_{j_k}\}$, $k \leftarrow k+1$
**end**
```

Algorithm 3: Topologically-aware heuristic for defining a transmission schedule from occupancy sets.

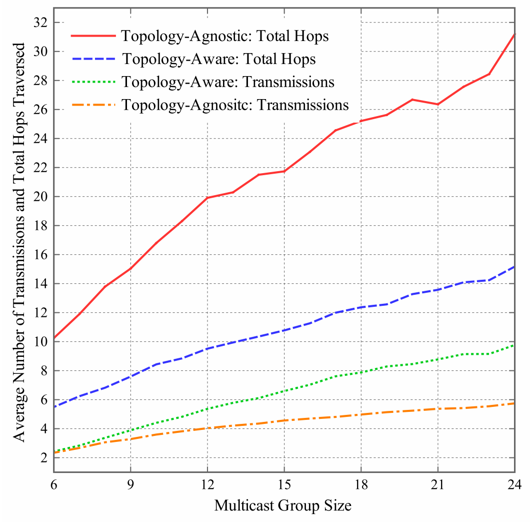


Fig. 7. Average energy required by the topology-aware and -agnostic transmission schedule generation heuristics measured as the number of packets transmitted and the total number of hops that those packets must traverse.

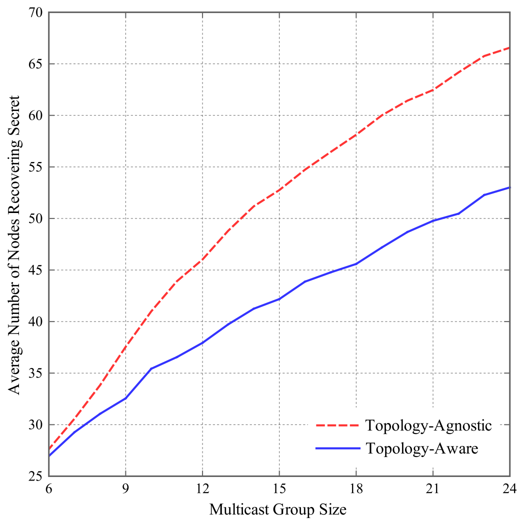


Fig. 8. Average number of nodes obtaining the group key in the topology-agnostic and topology-aware schemes.

C. Topology-Aware Protocol Discussion

We compared the performance to the topology-agnostic (Algorithms 1 and 2) and topology-aware schemes. To induce a random geometric graph topology, $n = 100$ nodes were placed randomly in a unit square and a transmission radius of $r(n) = \frac{3}{2}\sqrt{\frac{\log n}{\pi n}}$ was assumed (cf., [23]). Figure 7 compares the average energy required in terms of number of transmissions r and total hops required $\sum_{k=1}^{r} h_k$ for secret key agreement when $\beta = 0.2$. Although the topology-aware heuristic requires more packet transmissions, it requires approximately half the total hops. Figure 8 illustrates that the topology-aware scheme is also less susceptible to undetected compromised nodes owing to the use of controlled broadcasting.

D. Security-Aware Group Generation

The doping keys used in our protocols enable nodes outside of the intended multicast destination set to obtain the group private key. In networks with rich security policies, this may be unacceptable. For example, suppose that node v in security class $c(v)$ is permitted to receive messages sent only by a node in the same or lower security class. We can then associate a security class $c(O_j)$ with every occupancy set O_j:

$$c\left(O_j\right) = \min_i \left\{c(v_i) | p_j \in P_j\right\}, \tag{8}$$

where the minimization is performed over all network nodes (i.e., not just the destination set D). We can similarly define the security level of a multicast session from source s as

$$c(s, D) = \min_{v \in D \cup \{s\}} c(v). \tag{9}$$

With this notation defined, Algorithm 3 can be readily extended to support a hierarchical security policy.

```
**Input**: $D$, $\mathcal{O} = \{O_j\}_{j \in P_D}$, $c(s, D)$.
**Output**: $j_0$, $\mathcal{T} = \{(i_1, j_1, h_1), \ldots, (i_r, j_r h_r)\}$.
$j_0 \leftarrow$ packet index corresponding to largest set in $\mathcal{O}$
$S \leftarrow O_{j_0}$, $\mathcal{F} \leftarrow \mathcal{O} \setminus \{O_{j_0}\}$, $k \leftarrow 1$
**while** $S \neq D$ **do**
    $(i_k, j_k) \leftarrow$ indices of occupancy set $O_{j_k} \in \mathcal{F}$,
        such that $c(O_{j_k}) \geq c(s, D)$, and
        transmitter $d_{i_k} \in O_{j_k} \cap S$ that maximizes
        number of nodes/hop obtaining the secret:
            $\frac{|O_{j_k} \setminus S|}{h\left(d_{i_k}, O_{j_k} \setminus S\right)}$
    $h_k \leftarrow h\left(d_{i_k}, O_{j_k} \setminus S\right)$
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{i_k, j_k, h_k\}$
    $S \leftarrow S \cup O_{j_k}$, $\mathcal{F} \leftarrow \mathcal{F} \setminus \{O_{j_k}\}$, $k \leftarrow k + 1$
**end**
```

Algorithm 4: Topology- and security-aware heuristic for defining a transmission schedule from occupancy sets.

Observe that any node d_i in any occupancy set O_j used to generate the transmission schedule in Algorithm 4 satisfies $c(d_i) \geq c(s, D)$; therefore, only nodes with sufficiently high security can recover the group private key. Observe also that hierarchical security motivates a hierarchical distribution of the doping keys: the first set of doping keys is distributed to the nodes with highest security level, the second to those in the two highest levels, etc. If such a distribution strategy is not followed, then the security level of the doping keys will be low and Algorithm 4 will typically generate solutions that require roughly $t-2$ transmissions (i.e., pairwise keys will be used rather than doping keys to define the transmissions).

VI. Security Analysis

To this point, the common secret packets used to generate the group public/private key pair in our protocols have been described abstractly. In this section, we establish bounds on the lengths of the master and session keys required to achieve a desired level of security in the context of the NTRU PKA.

A. Algorithmic Key Generation for NTRU

In order to describe key generation in the NTRU cryptosystem, we require some notation. Let

$$R = \mathbb{Z}_q[x]/(x^M - 1) \tag{10}$$

be the polynomial ring with coefficients drawn from the integers modulo q and let $T(m_1, m_2) \subset R$ denote the subset of polynomials with m_1 coefficients equal to 1, m_2 to -1 (mod q), and $M - m_1 - m_2$ equal to 0. Key generation for NTRU begins by choosing two random polynomials: $F \in T(m_f, m_f)$ and $g \in T(m_g, m_g + 1)$. The private key is F while the public key is $h = g/(1 + pF)$ (p is an integer parameter). The security of NTRU arises from the computational intractability of obtaining F from h for properly chosen parameters.

In order to determine the required length of the common secret packet that will be used to derive the group private key in our protocol, two questions must be answered:

1) What values of the NTRU parameters (M, q, m_f) are required to achieve X-bit security?
2) How can a common secret packet be mapped algorithmically to a group private key F?

The first question has been answered in the open literature (cf., [24]). A bound on the base-2 logarithm of $|T(m_f, m_f)|$ lower bounds the number of bits required to generate F:

$$\log_2 |T(m_f, m_f)| \geq m_f \log_2 \left(\frac{M(M - m_f)}{m_f^2} \right). \tag{11}$$

To answer the second question, we leverage an algorithm that was presented in [25] for generating random binary words with a prescribed Hamming weight from variable length random bit strings. Specifically, let the set of q-bit binary words with Hamming weight y be denoted $W(q, y)$. A random element $F \in T(m_f, m_f)$ can then be generated in two steps:

1) Consume a string of l_1 bits to generate a random element $w_1 \in W(M, 2m_f)$. The 0 positions in w_1 define the coefficients in F that are 0.
2) Consume a second, independent string of l_2 bits to generate a random element $w_2 \in W(2m_f, m_f)$. The 1 positions in w_2 determine whether the remaining coefficients in F are $+1$ or -1.

This algorithm was implemented in order gather statistics on the number of bits $l_1 + l_2$ required to generate a random element of $T(m_f, m_f)$. Table I demonstrates that $4X$ bits are more than sufficient to generate a random private key for the NTRU cryptosystem with X-bit security.

TABLE I
Bits required for algorithmic private key generation in the NTRU cryptosystem

Cryptographic Strength	Parameters (M, q, m_f)	Lower Bound	Algorithmic Mean	Max
112-bit	$(659, 2048, 38)$	310	402	415
128-bit	$(761, 2048, 42)$	348	451	463
192-bit	$(1087, 2048, 63)$	512	671	687
256-bit	$(1499, 2048, 79)$	665	886	893

B. Pairwise, Doping, and Session Key Length

The group key generation protocol described in Section III-B uses a cryptographically secure function to generate sets of temporary session keys from the master keys stored in memory. Specifically, $4X$-bit keys $\{s_k\}_{k=0}^{r}$ for session u are derived from the master keys $\{p_k\}_{k=0}^{r}$ via a function

$$s_k \leftarrow \phi\left(p_{j_k}, u\right). \tag{12}$$

Binary sums of the form $s_0 + s_k$ for $1 \leq k \leq r$ are then transmitted over a non-secure channel. By construction of our protocol, s_0 will provide $4X$-bit security in the information theoretic sense if the master keys $\{p_k\}_{k=0}^{r}$ are $4X$-bit secure and if $\phi()$ is a $4X$-bit secure pseudorandom function (PRF). The National Institute of Standards Technology (NIST) provides guidelines for cryptographically secure pseudorandom number generation in [22]. NIST also provides guidelines for secure PRFs. In particular, the Keyed-Hash Message Authentication Code (HMAC) can be used with the session identifier u as an input variable and the master key as the seed parameter. This approach is consistent with NIST recommendations for key separation (i.e., for ensuring that the compromise of a session key derived from a master key does not degrade the cryptographic strength of that master key) [19]. In summary, a $4X$-bit secure HMAC can be used in conjunction with $4X$-bit cryptographic master keys to generate $4X$-bit cryptographic session keys, which in turn can be used to generate the group private key for an X-bit secure NTRU session.

VII. Conclusion

Motivated by a desire to employ public key cryptography in ad hoc networks, this paper described an energy-efficient group key agreement protocol. Whereas existing approaches require $O(t)$ transmissions to establish a group key among t nodes, our protocols require $O(\log_b t)$ transmissions at steady state, where b is a parameter that enables trades between energy efficiency and security against compromised nodes. When combined with an energy-efficient public key algorithm such as NTRU, our protocols could provide the benefits of public key cryptography – e.g., dynamic ad hoc network support – with features currently found only in symmetric key cryptosystems – e.g., energy efficiency and native multicasting.

Our approach was inspired by recent results on the universal recovery problem. Specifically, it was demonstrated in [12] that if nodes communicate for omniscience in an efficient manner (i.e., by using the least amount of transmissions), then the longest secret key possible can be generated. However, this is an unsatisfactory theoretical result in a crucial way: communication for omniscience isn't always necessary to generate secrecy. Indeed, the protocols described herein generate a single packet of secrecy via *partial recovery*. Our work thus motivates an information theoretic question: given an initial distribution of packets that are shared amongst a set of nodes, how many transmissions are required to generate a secret key of a particular length? Future work will address this question.

VIII. Acknowledgements

This material is based upon work funded by DARPA under contract W15P7T-12-C-5013. The authors thank Dr. Anthony Falcone of DARPA for his contributions to this work.

References

[1] X. Chen, *et al.*, "Sensor network security: A survey," *IEEE Comms. Surveys & Tutorials*, vol. 11, no. 2, pp. 52–73, 2009.

[2] K. Ren, W. Lou, B. Zhu, and S. Jajodia, "Secure and efficient multicast in wireless sensor networks allowing ad hoc group formation," *IEEE Trans. Veh. Tech.*, vol. 58, no. 4, pp. 2018–2029, May 2009.

[3] Q. Huang, C. Lu, and G.-C. Roman, "Spatiotemporal multicast in sensor networks," in *Proc. ACM Int. Conf. Embedded Networked Sensor Systems (SenSyS)*, Los Angeles, November 2003.

[4] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A ring based public key cryptosystem," *LCNS*, vol. 1423, pp. 267–288, 1998.

[5] A. C. Atici, *et al.*, "Low-cost implementation of NTRU for pervasive security," in *Intl. Conf. Application-Specific Systems, Architectures and Processors*, Leuven, Belgium, July 2008.

[6] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks," in *Proc. IEEE Conf. Computer Comms.*, 2000, pp. 585–594.

[7] T. R. Halford and K. M. Chugg, "Barrage relay networks," in *Proc. Information Theory and Applications Workshop (UCSD)*, La Jolla, CA, February 2010, (invited).

[8] Y.-M. Tseng and J.-K. Jan, "Anonymous conference key distribution systems based on the discrete logarithm problem," *Computer Communications*, vol. 22, no. 8, pp. 749–754, 1999.

[9] G. Horng, "An efficient and secure protocol for multi-party key establishment," *Computer J.*, vol. 44, no. 5, pp. 463–470, 2001.

[10] Y.-M. Tseng, "A robust multi-party key agreement protocol resistant to malicious participants," *Computer J.*, vol. 48, no. 4, pp. 480–487, 2005.

[11] T. A. Courtade and R. D. Wesel, "Weighted universal recovery, practical secrecy, and an efficient algorithm for solving both," in *Proc. Allerton Conf. Commun., Control, Comp.*, Monticello, IL, September 2011.

[12] I. Csiszar and P. Narayan, "Secrecy capacities for multiple terminals," *IEEE Trans. Information Theory*, vol. 50, pp. 3047–3061, 2004.

[13] A. Scaglione and Y.-W. Hong, "Opportunistic large arrays: Cooperative transmission in wireless multihop ad hoc networks to reach far distances," *IEEE Trans. Signal Proc.*, vol. 51, no. 8, pp. 2082–2092, 2003.

[14] R. Ramanathan, "Challenges: A radically new architecture for next generation mobile ad hoc networks," in *Proc. ACM/IEEE Int'l Conf. Mobile Comp. and Networking*, Cologne, Germany, August 2005.

[15] G. Bumiller, L. Lampe, and H.Hrasnica, "Power line communication networks for large-scale control and automation systems," *IEEE Comms. Mag.*, vol. 48, no. 4, pp. 106–113, 2010.

[16] S. Fujishige, *Submodular functions and optimization*. Berlin: Elsevier Science Publishers, 2010.

[17] S. Jaggi, *et al.*, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Information Theory*, vol. 51, no. 6, pp. 1973–1982, 2005.

[18] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct pseudorandom functions," *Proc. ACM*, vol. 33, no. 4, pp. 210–217, 1986.

[19] NIST Special Publication 800-108, *Recommendation for Key Derivation Using Pseudorandom Functions*. U. S. Dept. of Commerce, 2009.

[20] O. A. Telelis and V. Zissimopoulos, "Absolute $o(logm)$ error in approximating random set cover: An average case analysis," *J. Information Processing Letters*, vol. 94, no. 4, 2005.

[21] A. Demers *et al.*, "Epidemic algorithms for replicated database maintenance," in *Proc. ACM Symp. Princ. Dist. Comp.*, Vancouver, 1987.

[22] NIST Special Publication 800-90, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*. U. S. Dept. of Commerce, 2007.

[23] P. Gupta and P. R. Kumar, *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W. H. Fleming*. Boston, MA: Birkhauser, 1998, ch. Critical power for asymptotic connectivity in wireless networks, pp. 547–566.

[24] P. Hirschhorn, J. Hoffstein, N. Howgrave-Graham, and W. Whyte, "Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches," *LCNS*, vol. 5536, pp. 437–455, 2009.

[25] N. Sendrier, "Encoding information into constant weight words," in *Proc. IEEE Int. Symp. on Inform. Theory*, Adelaide, September 2005.