Journal of Telecommunications System & Management

Bajorski et al., J Telecommun Syst Manage 2014, 3:2
DOI: 10.4172/2167-0919.1000115

Research Article Open Access

Stochastic Analysis and Modeling of a Tree-Based Group Key Distribution Method in Tactical Wireless Networks

Peter Bajorski[1], Alan Kaminsky[2], Michael Kurdziel[3], Marcin Lukowiak[4], Stanisław Radziszowski[2] and Christopher Wood[2]

[1]*Center for Quality and Applied Statistics, Rochester Institute of Technology (RIT), USA*
[2]*Department of Computer Science, RIT, USA*
[3]*RF Communications Division, Harris Corporation, USA*
[4]*Department of Computer Engineering, RIT, USA*

Abstract

A number of key management challenges are encountered when operating tactical communication systems using a group-wide shared key. A large portion of such communications occurs over low bit-rate channels, and all communication channels must be available at any moment for mission action. Current over-the-air rekeying protocols consume too much channel bit-rate to be practical for large tactical radio networks. This caused an off-line pre-placed key (PPK) approach to become most commonly used key distribution method in these environments. Unfortunately, with this key management scheme, revoking group membership requires a full intra-mission rekey, which can be dangerous in a battlefield situation. This paper introduces a new group key distribution method called *Viral Electronic Key Exchange* (VEKE). This paper examines the protocol as an extension to the *Internet Key Exchange* (IKE) protocol, but any electronic key exchange protocol can be used (Ex. IKE v2). A feature of this protocol is a parallel key distribution scheme enabled by propagating the key management role to authenticated nodes while establishing security associations across the network. We performed a comprehensive stochastic analysis to develop a model for computing the expected rekey time across the entire group, taking into account the likelihood of node jamming, channel failures, and message corruption. This model was verified with a Monte-Carlo simulation. Our results confirmed that the VEKE protocol can accomplish an over-the-air rekey in a short period of time, even over low bit-rate systems, while preserving rigid security and channel availability properties of the network. It also allows for the amount of pre-placed public-key material and other preparations necessary in tactical networks to be minimized.

Keywords: Key management; Ultra low bit-rate networks; Wireless electronic key exchange

Scenario

Missions in tactical environments require communication systems with on-demand availability and high reliability. Wireless channels in the frequency bands of HF, VHF, and UHF tactical networks suffer from low bit-rates and high error rates. Note that waveforms designed for HF bands nominally provide 3 kbps bandwidth channels. For VHF/UHF bands, narrowband and wideband channel bandwidths are 5 kbps and 25 kbps respectively. To meet the requirement of high availability and reliability, up to 80% of the raw, over-the-air bit-rate can be consumed by overhead, error correction and integrity checking. Most standard network protocols, no matter what OSI layer they operate in, are optimized for operation on enterprise networks with tens of megabits per second of low error rate channels. Therefore, these protocols will often not provide satisfactory performance over tactical ad hoc networks.

Furthermore, with the growing pervasiveness of decentralized tactical networks, it is becoming increasingly important to ensure that communication among nodes in such networks remains secure. Electronic key management is the mechanism through which common keys are agreed upon or established in order to encrypt and decrypt sensitive data that cannot be sent over the air in plaintext. Currently, group-wide key management schemes rely on either a centralized control station and existing infrastructure to distribute shared keys among nodes in a group, a certain amount of pre-placed information within each node prior to each mission that enables the reconstruction of shared keys, or an expensive and computationally intensive public-key infrastructure to generate pair-wise shared keys on demand. Each of these approaches is accompanied with application-sensitive limitations that make them suitable for various scenarios [1,2]. However, in the context of tactical wireless networks, nodes face a threat of being compromised, which in turn results in a relatively high threat to the group key, or any individual security association that exists between two nodes [3,4].

In addition, a number of key management challenges are encountered when managing tactical communication systems. A large portion of tactical communications occurs over low bit-rate channels that are susceptible to natural and deliberate interference [5,6]. Current over-the-air rekeying protocols are not practical because of the amount of time consumed. Three main problems are encountered. First, up-to-date key material is essential to the security of a mission, so an expedient means of obtaining such material is necessary. Second, communication channel capacity is a limited resource and must be available at any time for mission action. Occupying significant amounts of air time for any maintenance operation, including key management, is unacceptable. This has resulted in an off-line pre-placed key (PPK) approach being the most acceptable key distribution method currently available. Installation of PPK material prior to the start of a mission, even though a manual operation, is straightforward and safe to execute.

***Corresponding author:** Michael Kurdziel, Harris Corporation, RF Communications Div., USA, Tel: 585-355-5863; E-mail: michael.kurdziel@harris.com

Received August 13, 2014; **Accepted** September 09, 2014; **Published** September 16, 2014

Citation: Bajorski P, Kaminsky A, Kurdziel M, Lukowiak M, Radziszowski S, et al. (2014) Stochastic Analysis and Modeling of a Tree-Based Group Key Distribution Method in Tactical Wireless Networks. J Telecommun Syst Manage 3: 115. doi:10.4172/2167-0919.1000115

Intra-mission rekey of these systems is another matter; a security officer must use a "Fill Device" to physically carry and load key material to each node. Under battlefield conditions, this operation can be life threatening to the security officer [7]. Lastly, while enabling group membership is a straightforward process of authenticating a node and then transferring the group key, revoking group membership requires a full intra-mission rekey.

These observations have led research efforts towards a public-key infrastructure (PKI) based key management scheme in which nodes will generate pair-wise keys to communicate securely. However, the problem associated with PKI schemes is that they are still naively used to establish pair-wise security associations between every pair of nodes without making effective use of the limited capacity. Traditional approaches have used PKI schemes to establish security associations for only the minimally required amount of nodes, and then use such associations to securely distribute a common group key that can be used by any node in the group to encrypt and send data to any other node(s) in the group [8-15]. The problem thus reduces to finding an effective security association establishment scheme.

Secure and efficient group key distribution in tactical environments is still an open problem. In contrast, several attractive solutions to the problem of key management in the context of wireless sensor networks (WSNs) have been proposed in the literature [16,17]. The more popular techniques depend on the construction of pair-wise secure channels between nodes in the network so that groups no longer share a common key. While these schemes sacrifice the ability to broadcast messages to the group, they benefit from the fact that if one node is compromised those remaining are not required to reestablish security associations with their neighboring nodes. Unfortunately, these solutions are not applicable in tactical network environments due to the need to broadcast and multicast messages among group participants.

This paper proposes a group key management scheme that addresses the aforementioned availability, reliability, and security requirements without sacrificing performance. Our method, the *Viral Electronic Key Exchange* (VEKE), which is based on the *Internet Key Exchange* (IKE) protocol, features a parallel key distribution scheme enabled by propagating the key management role to authenticated nodes while establishing security associations across the network. We performed a comprehensive stochastic analysis of VEKE to develop a model for computing the expected rekey time across the entire group, taking into account the likelihood of node jamming, channel failures, message corruption, and other realistic availability threats. The correctness of this model was verified with a Monte-Carlo simulation. Our results confirmed that the VEKE protocol can accomplish an over-the-air rekey in a short period of time, even over low bit-rate systems. It also allows for the amount of PPK material and other preparations necessary in tactical networks to be minimized.

The rest of this paper is organized as follows: Section II provides some background into existing key management protocols, with a particular focus on those proposed for use with wireless networks. Section III discusses the behavior of the protocol and how it functions in environments with arbitrarily sized groups of nodes. Section IV discusses role of the protocol in the OSI network stack. Section V then continues to introduce the mathematical model that was used to approximate the behavior of the protocol based on parameters defined by the network environment. Verification using Monte Carlo simulations is presented in Section VI. Finally, the paper is concluded in Section VII with discussion on future work.

Existing Standards and Adopted Protocols

The *Internet Key Exchange* (IKE) protocol [9] is a standard security protocol that is used to conduct a point-to-point authenticated key exchange to establish an IPsec association between two parties in a network. It is considered a hybrid protocol because it is based on the *Internet Security Association and Key Management* (ISAKMP) and *Oakley* protocols-two widely used key management schemes [9]. ISAKMP is responsible for secure session management between two peer nodes in a network; whereas Oakley defines the mechanisms for the actual key exchange over the IKE session. The key exchange mechanism used by both *Oakley* and IKE is the *Diffie-Hellman* Key Exchange protocol, which is a technique for establishing a common key among two (or more) parties by relying on the computational intractability of the discrete logarithm problem.

The IKE protocol is very flexible in that it allows specifying the exact encryption algorithm, hashing MAC algorithm, peer authentication procedure, *Diffie-Hellman* group, and security association lifetime during normal operation [10]. The dynamic nature of the protocol lends itself to the application of peer-to-peer security association establishment in many different environments, including wireless, ad hoc networks, which have traditionally been dominated by protocols of one of the following forms [11]:

1. Centralized group key management protocols
2. Decentralized key management protocols
3. Distributed group key management protocols

Centralized group key management protocols utilize an existing infrastructure, often called the *Key Distribution Center* (KDC), in order to control the set of keys used by members of an entire group. If the KDC is compromised, then all group communication is as well. Decentralized group key management protocols elect specific nodes (or groups of nodes) to act on behalf of a single KDC, thus breaking the problem of key management up into one that targets many smaller groups. While this does not explicitly rely on a single location to oversee key management for the entire group, the subgroup key managers are single points of failure for the entire group and must be chosen and protected carefully.

Distributed group key management protocols are relatively recent schemes that are commonly used in industry, where every node participates in some way to generate a common group key for all members. Many derivatives of this protocol family have been proposed, including the *Group Diffie-Hellman Key Exchange* (G-DH) [8], *Octopus Protocol* [12], and the *Password Authenticated Multi-Party Diffie-Hellman Key Exchange Protocol* (PAMPDHKE) [13]. However, most Diffie-Hellman based protocols are executed recursively in a point-to-point manner between pairs of nodes in the group until a security association is established between all members. In addition, any authentication schemes that are layered on top of such protocols are also point-to-point.

Depending on the context in which these protocols are utilized, there are often many performance requirements that must be satisfied under constraints imposed by either the operators or the physical environment itself. Common constraints include limited channel capacity, limited computational resources on behalf of each node, and limitations on the amount of pre-placed information located within each node at the start of a mission. An additional functional constraint for the protocol is that it is simple to add new members to the group,

but hard to remove a single member from the network group. This is because removing a member requires an entire network rekey.

Aside from the performance requirements for such protocols, they must also be secure against common attacks, including variations of the popular man-in-the-middle attack [8]. These requirements are often fulfilled by relying on the computational intractability of breaking the Diffie-Hellman key exchange protocol. This is commonly referred to as the standard for secure key exchange mechanisms. However, with security comes the cost of performance. It therefore becomes an engineering and mathematical problem to balance the amount of performance required by such protocols with the amount of security they provide.

In light of the performance and security requirements for key management protocols, we propose a mode of operation for the existing IKE protocol, referred to as the *Viral Electronic Key Exchange* (VEKE) protocol. This protocol enables simultaneous IKE security associations between more than one pair of nodes in a group at a time. This approach allows us to attain almost as much parallelism as possible within the limitations of the underlying spanning tree formation of the nodes in the group, because the highly computational portions of the protocol are done in parallel. Furthermore, once unique security associations have been established for all members of the group, we simply use them to distribute a common group key from a single key manager across the corresponding network spanning tree. The security of this mode of operation is directly reducible to that of the IKE protocol, and thus it is possible to attain high security measures and performance by establishing security associations between pairs of nodes in a group in parallel.

Viral EKE Protocol - Parallel Security

Association and key distribution

The IKE protocol is constrained in that it can only construct point-to-point security associations. In the setting of ad hoc networks a need for group-wide associations is needed to transmit a group session key from a single node to all other nodes in the least possible elapsed time. Fortunately, modern waveforms and radios do permit parallelization. Our key distribution protocol takes advantage of this fact and the point-to-point nature of the IKE protocol by distributing the work done among the nodes in the network.

In tactical environments, rekey events are triggered by a *Tactical Operations Center* (TOC). The need for a group-wide rekey is determined using field intelligence information, and thus should only originate from this central point of authority. Since the TOC maintains the authorized user list (AUL), it is responsible for identifying a single node to act as a trusted root key manager to generate and propagate a new group key throughout the network. This is done after sending each node an updated AUL so as to prevent nodes from adding compromised nodes to the group. Since it is usually the case that there is more than one node on the AUL, there is no single node dependency for starting a rekey event in the network. The TOC can initiate a rekey event with any node that is within its range of communication and is on the AUL. When the TOC establishes the Security Association (SA) with this node, it will be authenticated and will then become the trusted key manager. In the tactical environment, the assumed network topology consists of a single node that is identified for initiating the rekey events and the remaining nodes in the group. An example of a spanning tree structure for such a network is shown in Figure 1.

Although a rekey event must be started by a single node, we emphasize that the only constraint for this node is that it is on the group-wide AUL. Therefore, while the key distribution scheme is originally centralized at the key manager to generate and distribute the key among its children, the rekey process is immediately decentralized once this stage is complete. Should this stage fail, the TOC can select another root key manager to begin a rekey event without placing the network in an unknown state.

Once a rekey event is initiated by a trusted root key manager in response to a command sent from the TOC along with a current AUL, this node will then complete IKE security associations with all of its child nodes and piggyback the establishment of this security association with the transmission of the new group key node and most current UAL. This process recursively spreads throughout the network using the underlying spanning tree maintained at the network layer of the individual nodes. Specifically, these newly connected child nodes will then continue to establish security associations with their respective child nodes and forward the group key when completed. This process is repeated until all nodes in the group have been connected. Key distribution is only limited by the physical transceiver properties of each node and multiplexing scheme of the waveform used to transmit data. In the target waveform for this protocol, a Time Division Multiple Access (TDMA) slotting scheme is used to schedule node access to the waveform, but for the analysis of our protocol we do not assume any such constraints.

After a group key has been established among a set of nodes, adding members to the group is a simple process. The AUL is continually updated by the TOC. The TOC will be responsible for removing nodes from the list in the event that a node is compromised. Prospective nodes to be added to the group can actively request to be part of the group or passively wait to be added by a currently connected node. In the active node addition scheme, a node will initiate an IKE security association with a currently connected and authenticated node. Upon completing the IKE exchange and authenticating the new node's identity, the AUL will be inspected to see if this new node can be added to the group. If so, the group key is encrypted using the new IKE session key and forwarded through the secure channel and the AUL is transferred to the new member of the group. In the passive node addition scheme, a member of the group will request to establish an IKE security

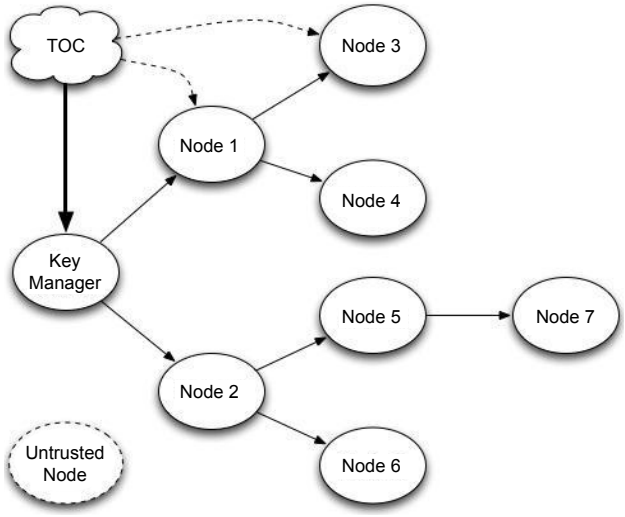

Figure 1: A sample spanning tree of nodes in a wireless network. The TOC can select any of the trusted nodes to serve as the key manager based on the contents of the AUL. The untrusted node would not be chosen to serve as the key manager.

association with an unconnected node. Upon completing this security association and authenticating the node's identity, the AUL will again be inspected and the group key will be forwarded to the new node using the same procedure outlined in the active node addition scheme.

Ad hoc Networks and the OSI Protocol Stack

Each radio in the network supports a full OSI protocol stack complete with full routing capabilities. All radios in the network are mobile. As the nodes move, the lower three layers (Physical, Data Link and Network) work together to establish a common routing table. The proposed VEKE method will be initiated from the Application layer (Layer 7) and will execute at the Network layer (Layer 3). VEKE uses the IKE protocol to achieve Over-The-Net rekeying that is optimized to conditions in the tactical environment through knowledge of the network spanning tree maintained in the routing tables.

The first stage of the IKE protocol is achieved through execution of the Internet Security Association and Key Management Protocol (ISAKMP) Phase 1. This protocol establishes a Security Association (SA) by authenticating the identity of destination nodes against an AUL. VEKE extends this authentication to authorize destination nodes to assume the Key Distribution role. A node is limited to authenticating a maximum number of two child nodes so that 1) a minimum percentage of channel capacity available for mission objectives is guaranteed and 2) a closed form stochastic analysis is possible. Based on the mission-critical nature of tactical communications, it is important that a rekey event does not consume all of the available network channel capacity. Doing so might block more important messages from propagating throughout the network.

After an SA is established between a node and its child nodes, the established session key is used to securely transport a group key and current AUL to the child nodes. The child nodes then repeat the distribution process with additional child nodes who are identified using the network spanning tree information stored in the network routing tables. Child nodes that are a "single hop" away are given preference due to their physical proximity.

Key Distribution Stochastic Model

In this section, an IKE exchange is abstracted into one transmission that completes in an arbitrary time unit that we will refer to as one epoch. Our goal here is to calculate how many epochs on average it will take to distribute the group key to all nodes in the network.

Using the epoch units of time, it is possible to calculate a real-time approximation for the total key distribution time using the physical properties of the communication channel and the specific IKE mode. For example, consider two parties that use a wireless channel with a bit-rate of 85kbps and send an average of 4,000 bytes of data to complete an IKE exchange. Also assume that the computational and data transmission overhead for the IKE exchange is approximately 0.8s. With this information, one can calculate the time for one IKE exchange, which is one epoch, by multiplying the inverse of the bit-rate with the data sent between the two parties and then adding the overhead time to this product, which in this example gives approximately 1 second. This epoch time can then be multiplied by the needed number of epochs to yield a real-time approximation of the total key distribution time. It must be emphasized that the analysis presented in this paper was performed to show that this method will achieve a network wide rekey using any EKE protocol (with any epoch size) that is more efficient than current methods used in tactical networks. For example, one current method requires the TOC to establish authenticated SAs directly with each node in the network. Here a group-wide rekey will require time O(n), where n is the number of nodes in the network. Results for VEKE show an increasing time advantage of O(log n) as the network size increases.

Our stochastic model assumes that a node can establish a connection (IKE exchange) with only one other node (child node) within one epoch and then with another node in a future epoch. Any node cannot establish more than two connections with other nodes. In order to trace the process of the key distribution, we can use a tree spanning the nodes that already received the key. Figure 2 shows an example of a tree spanning Nodes 1-4, while Nodes 5-7 have not yet received the key. It should be emphasized that the spanning tree only shows the specific order how the nodes connected in the given scenario, and node numbering is just for convenience. The spanning tree does not represent any predetermined structure of nodes. For example, the next time when the key is distributed, the key manager might connect first to Node 6.

Let n be the total number of nodes, including the key manager. Then we have $S_1+S_2+S_3+S_4 = n$.

Note that the starting point of the key distribution process is $S^*=(0,0,1,n-1)$. In order to trace the progress of the key distribution, we need to identify which subsequent states S can be attained in this process. In other words, we want to derive transfer equations that would tell us what the possible transfers are from one to another state of the network in one epoch. To this end, let us define G_2 as the number of new connections that are made within a given epoch from the S_2 nodes already having one child connection. In the same fashion, G_3 is the number of new connections made within a given epoch from the S_3 nodes having no connections. If a node with one child connection establishes a new one, it becomes a node with two children. Hence, S_1 increases by G_2 and S_2 is reduced by G_2. At the same time, G_2 new nodes received the key, reducing S_4 by G_2, and increasing S_3 by G_2. These changes are reflected in the following transfer equation:

In order to describe the network state in a concise way, we use four numbers denoted by S_1 through S_4 as described in Table 1. S_1 is the number of nodes having two child connections with nodes that already received the key. S_2 is the number of nodes having one child connection. S_3 is the number of nodes that are already connected (that is, received the key), but they do not have any child connections. S_4 is the number of nodes that have not yet received the key. The network state is then described by a four-dimensional vector $S = (S_1, S_2, S_3,$

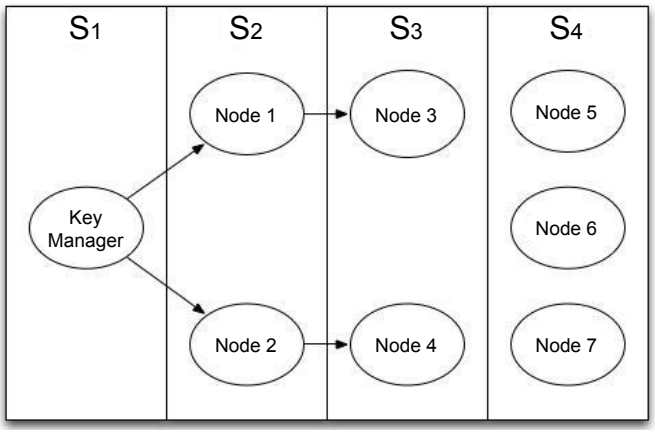

Figure 2: A tree spanning Nodes 1-4, while Nodes 5-7 have not yet received the key.

	Number of completed child connections			Unconnected nodes
	2	1	0	
Number of nodes	S_1	S_2	S_3	S_4

Table 1: Network state variables vector S.

S_4) . For example, the state of the network shown in Figure 2 can be described by S = (1,2,2,3).

Let n be the total number of nodes, including the key manager. Then we have $S_1+S_2+S_3+S_4 = n$. Note that the starting point of the key distribution, we need to identify which subsequent states S can be attained in this process. In other words, we want to derive transfer equations that would tell us what the possible transfers are from one to another state of the network in one epoch. To this end, let us define G_2 as the number of new connections that are made within a given epoch from the S_2 nodes already having one child connection. In the same fashion, G_3 is the number of new connections made within a given epoch from the S_3 nodes having no connections. If a node with one child connection establishes a new one, it becomes a node with two children. Hence, S_1increases by G_2 and S_2 is reduced by G_2. At the same time, G_2 new nodes received the key, reducing S_4 by G_2 and increasing S_3 by G_2. These changes are reflected in the following *transfer equation*:

$$S_1 \to S_1 + G_2$$

$$S_2 \to S_2 - G_2 + G_3$$

$$S_3 \to S_3 + G_2$$

$$S_4 \to S_4 - G_2 - G_3$$

If a node with no child connections establishes a new connection, it becomes a node with one child. Hence, S_2 increases by G_3 and S_4 is reduced by G_3, which is also reflected in the above transfer equations. In each epoch, there is a certain number of new G_2 connections, and they accumulate over multiple epochs. If G_2, k denotes the number of new G_2 connections within the k-th epoch, we can denote by D_2 the cummulative number of connections until a given m-th current epoch, that is, $D_2 = \sum_{k=1}^{m} G_{2,k}$. In the same fashion, we can define D_3 as the cummulative number of G_3 connections until the current epoch.

We can now use the transfer equations (1) for a transfer from S*=(0,0,1, n-1) to an arbitrary admissible state S with the transfer equations written as

$$\begin{aligned} &0 \to D_2 \\ &0 \to D_3 - D_2 \\ &1 \to 1 + D_2 \\ &n-1 \to n-1-D_2-D_3 \end{aligned} \qquad (2)$$

It is now clear that a state S can be fully characterized by a two-dimensional vector D (D_2, D_3) by writing S = (D_2, D_3 - D_2, 1 + D_2, n - 1 - D_2 - D_3). This defines a new state space of vectors D that describes possible states of the network. The starting point, equivalent to S^*, is $D^* = (0,0)$ in the new state space. From the fourth transfer equation in (2), we have $n-1-D_2-D_3 \geq 0$, which means that $D_2 + D_3 \leq n-1$. When all nodes receive the group key, we have $D_2 + D_3 = n-1$ and the key distribution process ends. From the second transfer equation in (2), we have $D_3 - D_2 \geq 0$, which means that $D_3 \geq D_2$. Based on these constraints, Figure 3 shows the area representing admissible states D. In the process of the key distribution, we need to move from the starting point D* = (0,0) to the line $D_2 + D_3 = n-1$.

We now investigate various ways that we can move within one epoch from state D to D + h, where h = (i, j) is the change in the network state within one epoch. Since i is the number of new connections that are made within one epoch from $S_2 = D_3 - D_2$ nodes already having one child connection, we have $0 \leq i \leq D_3 - D_2$. In the same fashion, j is the number of new connections from $S_3 = 1 + D_2$ nodes having no connections, which means that $0 \leq j \leq 1 + D_2$. At the same time, the total number ($i + j$) of new connections cannot be larger than the number $S_4 = n-1-D_2-D_3$ of available nodes without the key. Hence, $i + j \leq n-1-D_2-D_3$. The constraints are summarized as follows:

$$\begin{aligned} &0 \leq i \leq D_3 - D_2 \\ &0 \leq j \leq 1 + D_2 \\ &i + j \leq n-1-D_2-D_3 \end{aligned} \qquad (3)$$

In order to calculate how many epochs it will take to distribute the group key to all nodes in the network, we define a random variable T_D as the number of epochs to reach the last state (with all nodes connected, as on line $D_3 = n-1-1-D_2$ in Figure 3) from state D. Our goal is to calculate the average (mean) time $E(T_{(0,0)})$.

Let $\mathbf{D}_s$ be a random vector describing the state of the network after s epochs. We assume that $\{D_s : s \geq 0\}$ is a stationary discrete-time Markov chain [14,15] with a given set of probabilities

$$p_D(h) = \Pr\{D_{s+1} = D + h \mid D_s = D\} , \qquad (4)$$

where $h = (i, j)$ is the change in the network state within one epoch. These probabilities are assumed to account for the probability of node failure, node link failure, failed messages, node mobility, and targeted jamming (DoS) attacks. Note that $T_D = 0 \Leftrightarrow E(T_D) = 0 \Leftrightarrow D_2 + D_3 = n-1$. For D such that $D_2 + D_3 < n-1$, one can prove the following backward recursive formula

$$E(T_D) = \frac{1}{1 - p_D(h^*)}\left[1 + \sum_{h \in A_D} p_D(h) E(T_{D+h})\right], \qquad (5)$$

where $h^* = (0,0)$ and

$$A_D = \{(i,j) : 0 < i + j \leq n-1-D_2-D_3, \\ 0 \leq i \leq D_3 - D_2, 0 \leq j \leq 1 + D_2\} , \qquad (6)$$

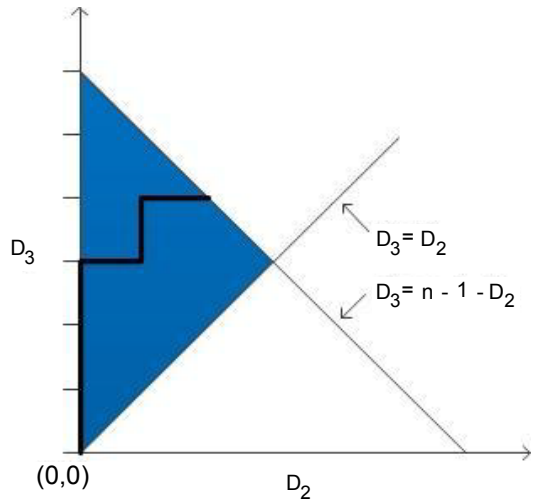

Figure 3: Admissible states D (shaded area), with a possible path in the process of key distribution.

as follows. Note that $A_D \cup \{h^*\}$ is the set of all possible values for the change h in the network state within one epoch. This means that

$$E(T_D) = \sum_{h \in A_D \cup \{h^*\}} p_D(h)\left[E(T_{D+h}) + 1\right]$$

Since $\sum_{h \in A_D \cup \{h^*\}} p_{D(h)=1}$, we obtain

$$E(T_D\left[1 + p_D(h^*)\right] = 1 + \sum_{h \in A_D} p_D(h) E(T_{D+h})$$

which leads to formula (5). For computational purposes, it is convenient to write (5) in the form

$$E(T_D) = \frac{1}{1 - p_D(h^*)}\left\{1 + \sum_{i=0}^{\min\{D_3 - D_2, S_4\}}\left[\sum_{j=k_i}^{m_i} p_D\left((i,j)\right) E\left(T_{D+(i,j)}\right)\right]\right\}$$

where $k_i = \begin{cases} 1 \text{ if } i = 0 \\ 0 \text{ if } i > 0 \end{cases}$, $m_i = \min\{1 + D_2, S_4 - i\}$, $S_4 = n - 1 - D_2 - D_3$

In order to calculate $E(T_{(0,0)})$, one needs to start by setting $E(T_D) = 0$ for D such that $D_2 + D_3 = n - 1$, and then move in the direction of $D^* = (0,0)$ by applying (5) or (7) along the lines parallel to $D_2 + D_3 = n - 1$. In order to apply these formulas, we also need to calculate the transition probabilities $p_D(h)$. To this end, we assume a fixed probability p that a given node without a key is ready to receive the key. We also assume that any node that can still add more child connections will establish such a connection in a single epoch if there is a node ready to receive it. Let k be the number of nodes that can still add more child connections. If there are more than k nodes that are ready to receive the key, then only k of them will connect in a random fashion (each combination with the same probability) in a given epoch. Let $b = i + j$ and $U = \min(1 + D_3, S_4)$. It is easy to see that the following formulas hold.

If $b = U = 1 + D_3$, then

$$p_D(h) = \sum_{k=b}^{S_4} \binom{S_4}{k} p^k (1-p)^{S_4 - k} \tag{8}$$

Otherwise (which really means that $b < 1 + D_3$ or $U < 1 + D_3$), we have

$$p_D = g(i,j) \binom{S_4}{b} p^b (1-p)^{S_4 - b}, \tag{9}$$

where $h = (i,j)$ and

$$g(i,j) = \frac{\binom{D_3 - D_2}{i}\binom{1 + D_2}{j}}{\binom{1 + D_3}{b}} \tag{10}$$

Monte-Carlo Simulation Verification

In order to verify the correctness of the stochastic model, a Monte-Carlo simulation that emulates the discrete time steps of a rekey operation was implemented. The simulation performs the steps shown in Algorithm 1 to rekey a network.

Algorithm 1. Probabilistic group key propagation.

1. Initialize the key manager as the only node with the group key. Initialize all other nodes as unconnected.

2. Set *time = 0*

3. While (no connected nodes exist)

a. Form a queue *A* by a subset of unconnected nodes, each taken with a probability *p*.

b. Form a queue *B* of nodes that are able to accept new children.

c. Shuffle *B* so that the ready parents appear in random order.

d. Set *stopCount* = min{|*A*|, |*B*|} and *count* = 0

e. While (*count* < *stopCount*)

i. Remove the first element from *A* and assign it to *c*.

ii. Remove the first element from *B* and assign it to *d*.

iii. Connect *c* and *d*.

iv. Set *count* = *count* + 1

f. Set *time* = *time* + 1

4. Return *time*

In the context of this procedure, *p* represents the probability that a given node without a key is ready to receive the key as described in Section IV. Also, note that node connections are maintained internally by an adjacency matrix representing the connection status of the network. Unconnected nodes and ready parent nodes are found by iterating over this matrix. As with the stochastic model, a ready parent is one that has strictly less than two connected children [16,17].

This simulation is then repeated through several iterations. The resulting integer time values are averaged to yield the expected rekey time. The simulation was run for *p* starting with 0.1 and increasing to 1.0 in increments of 0.1. The results indicate that the average rekey time matched the theoretical results from the stochastic model almost exactly. As an illustration of the similarity between these two results, the rekey time values from the theoretical model and verification simulation for a probability of *p* = 0.5 with a variety of group sizes are shown in Table 2.

Similar values were found for all values of *p* with the same group sizes (Figure 4 shows a subset of these values). These results provide an independent confirmation of our stochastic model, which can now be used to compute the exact values for the expected rekey time for a network of size n with an uncertainty about the IKE exchange completion expressed through the probability *p*.

To verify that our constraint of two children was not overly restrictive on the protocol's performance, we modified our Monte-

Group Size	Model	Simulation	Difference (%)
5	4.49	4.49	< 0.01
10	6.31	6.33	0.32
15	7.51	7.53	0.27
20	8.45	8.45	< 0.01
25	9.16	9.15	0.11
30	9.77	9.77	< 0.01

Table 2: Stochastic model and simulation results for the expected rekey time in epochs.

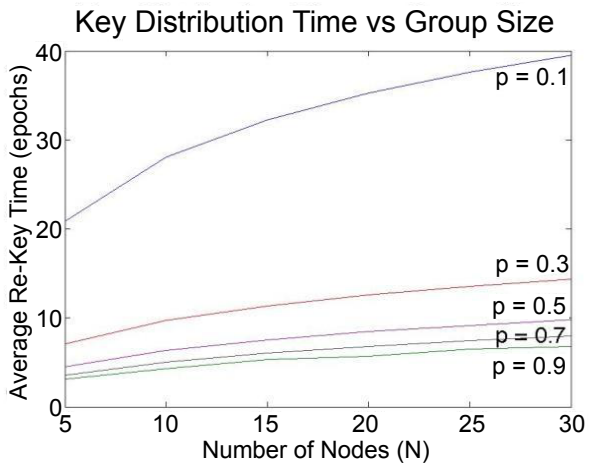

Figure 4: Calculated and expected network rekey times in terms of probability p, p = 0.1, 0.3, 0.5, 0.7, 0.9, that a given node currently without a key is ready to receive it. The values for p = 0.5 are shown in Table 2.

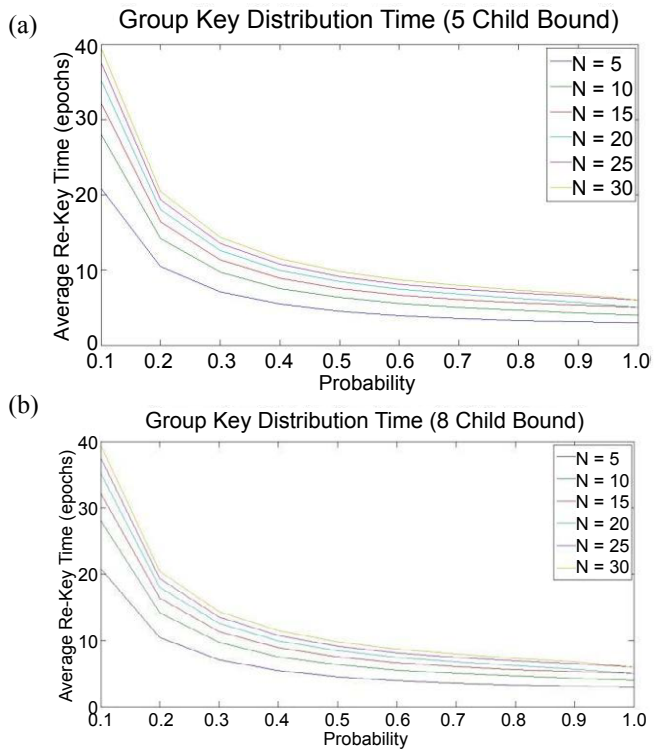

Figure 5: Expected group rekey times for child bounds of 5 (panel (a)) and 8 (panel (b)), respectively. There is no discernible quantitative difference in the performance of the protocol for groups of this order when the child constraint is relaxed.

Carlo simulations by increasing the upper bound on the number of children. The expected timing results from these simulations are shown in Figure 5. Clearly, there is no discernible performance difference as the upper bound is increased to 5 or 8, which justifies our constraint of a maximum of two children from a performance and analysis perspective [18-20].

Conclusions

From the previous discussion, it is clear that a practical method for over-the-air group rekey on low bit-rate nets such as Viral EKE is an enabling technology. Our method allows the amount of PPK material and other mission preparations to be minimized. It is conceivable that a signed public-key certificate installed at the time of manufacture might be all the preloaded material that is required. The results presented in this paper demonstrate that Viral EKE can accomplish an over-the-air rekey in a short period of time even over low bit-rate systems. The method can be implemented so that it is automatic after initiation. Besides being convenient for wireless tactical applications, group rekey will be less error-prone and less likely to introduce issues such as stranded nodes. In the tactical environment, stranded nodes arise when a rekey event initiated by the TOC does not succeed in reaching all the nodes. The nodes without current key material will be "stranded" and not be able to communicate securely with rest of the network. The VEKE method increases the probability of a successful rekey of each node because it does not require that each node be within communication range with the TOC. Rather a node only needs to be within range of any node that already has the key. The method also offers the promise of eliminating the need for physical key refill under battlefield conditions, thus removing the risk of personnel harm or loss. Future work will involve generalizing the stochastic model to support unbounded children and also packet-level interleaving between nodes during rekey events.

Acknowledgements

This work was supported in part by a grant from Harris Corporation, RF Communications Division. The authors would like to acknowledge the valuable contributions, technical advice and support from Michael McPhee and William Skiba.

Peter Bajorski received the B.S./M.S. degrees in mathematics from the University of Wroclaw, Poland, and the Ph.D. degree in mathematical statistics from the Technical University of Wroclaw, Poland. He held positions at Cornell University, the University of British Columbia, Simon Fraser University, and NY State Department of Transportation. Currently, he is Professor and Graduate Program Chair in the Graduate Statistics Department at Rochester Institute of Technology. His research interests include applied stochastic methods in imaging, networking, transportation, and other fields. Other interests include multivariate stochastic methods, regression techniques, and design of experiments. Dr. Bajorski is a member of the American Statistical Society, a senior member of SPIE, and a senior member of IEEE. He is also the author of the book Statistics for Imaging, Optics, and Photonics published in Wiley Series in Probability and Statistics.

Alan Kaminsky. With 35 years of computing experience spanning industry and academia, Alan Kaminsky has developed telephone switching system software at Bell Laboratories, developed real-time embedded control software and fuzzy genetic algorithms at Harris Corporation, taught graduate software engineering as an Assistant Professor at the Rochester Institute of Technology, and worked on printer system architectures at Xerox Corporation. While at Xerox, Alan got involved with Sun Microsystems' Jini Network Technology, led the Jini Printing Working Group industry consortium that defined a draft specification for the Jini Print Service, and was part of the expert group that developed the Java Print Service API released as package javax.print in the standard Java platform. Alan was also one of the original members of the Jini Community Technical Oversight Committee. Now a Full Professor in the Department of Computer Science at the Rochester Institute of Technology, Alan teaches and conducts research in parallel computing, cryptography, and computational science (primary interests), distributed systems, ad hoc networking, and security (secondary interests). Alan invented Parallel Java, an API and middleware for parallel programming in 100% Java on shared memory multiprocessor (SMP, or multicore) parallel computers, cluster parallel computers, and hybrid SMP cluster parallel computers. Alan wrote the textbook Building Parallel Programs: SMPs, Clusters, and Java (Cengage Course Technology, 2010) based on Parallel Java. Alan also wrote the textbook Simulation Simplified (Creative Commons, 2011). Alan has a B.S. in Electrical Engineering from Lehigh University and an M.S. in Computer Engineering from the University of Michigan.

Michael Kurdziel is Sr. Engineering Manager, Core Networking and Cyber Security, for Harris Corporation. His area of technical expertise is secure communications systems design. This includes the design of encryption, key management and authentication systems and algorithms. Dr. Kurdziel has been a member of Harris Corporation's RF Communication technical staff since 1992. He holds Bachelor of Science (1986), Master of Science (1988) and Doctor of Philosophy (2001) Degrees in Electrical Engineering from the State University

of New York at Buffalo. He holds thirteen patents, has two patents pending and has authored/coauthored 15 publications all dealing with military communications applications. He has been a licensed Professional Engineer in the State of New York since 1992.

Marcin Lukowiak obtained the B.S./M.S. degrees in the Department of Control and Systems Engineering, and the Ph.D. degree in the Faculty of Electrical Engineering, both at Poznan University of Technology, Poland. He has almost 16 years of academic experience and currently is an Associate Professor in the Department of Computer Engineering at Rochester Institute of Technology. His professional interests are concentrated in cross-disciplinary areas involving reconfigurable computing, hardware and hardware-software systems, cryptographic engineering, and high performance and heterogeneous computing.

Stanisław Radziszowski is a Professor in the Department of Computer Science since 1995. He earned Ph.D. from the Institute of Informatics at the University of Warsaw. During the years 1980-1984 he worked in IIMAS at the National Autonomous University of Mexico in Mexico City, and since 1984 at the RIT. In the 1990's he held three times 6-week visiting positions at the Australian National University in Canberra, and maintained collaborations with universities in Poland. His main research interest is in combinatorial computing - solving classical problems in combinatorics, graph theory and design theory, usually with the help of massive computations. Bounds on Ramsey numbers are his favorite. His survey titled "Small Ramsey Numbers", which is a regularly updated living article at the Electronic Journal of Combinatorics, became a standard reference in this area. He teaches mostly theory oriented courses, including very popular courses on cryptography, both at undergraduate and graduate levels. His recent work on applied cryptography led to joint projects with Computer Engineering Department.

Christopher Wood is a current PhD student at the University of California Irvine focusing on applied cryptography and computer security. He graduated from the Rochester Institute of Technology in August 2013 with a MS degree in computer science and dual BS degrees in computer science and software engineering, as well as a minor in mathematics. His primary research interests include theoretical and applied cryptography, information privacy and anonymity, computer and network security, computational graph theory, and hardware-software co-design. Beyond academia, he has accumulated over a year of professional software development experience at Xerox PARC, Intel, and L-3 Communications, among other companies.

References

1. Challal Y, Seba H (2005) Group Key Management Protocols: A Novel Taxonomy. International Journal of Information Technology 2: 105-118.
2. Hegland A, Winjum E, Mjølsnes S, Rong C, Kure Ø, et al. (2006) A Survey of Key Management in Ad Hoc Networks. IEEE Communications Surveys & Tutorials 8: 48-66.
3. Kidston D, Li L, Tang H, Mason P (2010) Mitigating Security Threats in Tactical Networks, Information Systems and Technology Panel Symposium, Canada.
4. Zhou L, Haas Z (1999) Securing Ad Hoc Networks, IEEE Network, Special Issue on Network Security.
5. Graham A, Kirkman N, Paul P (2007) Mobile Radio Network Design in the VHF and UHF Bands, Wiley & Sons Inc.
6. Li L, Vigneron P (2010) Properties of Mobile Tactical Radio Networks on VHF Bands, Information Systems and Technology Panel Symposium.
7. Wallner D, Harder E, Agee R (1999) RFC 2627: Key Management for Multicast: Issues and Architectures, NSA Network Working Group, USA.
8. Steiner M, Tsudik G, Waidner M (1996) Diffie-Hellman Key Distribution Extended to Group Communication, 3rd ACM Conference on Computer and Communications Security, ACM Press, New York, USA.
9. Harkins D, Carrel D (1998) RFC 2409: The Internet Key Exchange (IKE), NSA Network Working Group, USA.
10. Zhou J (2000) Further Analysis of the Internet Key Exchange Protocol, Computer Communications, 23: 1606-1612.
11. Rahman R, Rahman L (2008) A New Group Key Management Protocol for Wireless Ad-Hoc Networks, World Academy of Science. Engineering and Technology 2: 464-469.
12. Becker K, Wille U (1998) Communication Complexity of Group Key Distribution, Proceedings of the 5th ACM Conference on Computer and Communications Security, ACM Press.
13. Asokan N, Ginzboorg P (2000) Key Agreement in Ad Hoc Networks, Computer Communications, 23: 1627-1637.
14. Kemeny JG, Snell JL (1976) Finite Markov Chains, Springer-Verlag, USA.
15. Stroock DW (2005) An Introduction to Markov Processes, Springer, USA.
16. Wenliang Du, Jing Deng, Yunghsiang S. Han, Pramod Varshney, Jonathan Katz, et al. (2005) A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks, In The ACM Transactions on Information and System Security (TISSEC) 8: 228-258.
17. Donggang Liu, Peng Ning, Rongfang Li (2005) Establishing Pairwise Keys in Distributed Sensor Networks, in ACM Transactions on Information and System Security 8: 41-77.
18. MIL-STD-188-141B, Department of Defense Interface Standard, Interoperability and Performance Standards for Medium and High Frequency Radio Systems," 1 March 1999, U.S. DoD.
19. MIL-STD-188-143 (1989) Department of Defense Interface Standard, Interoperability and Performance Standards for Tactical Single Channel Ultra High Frequency, U.S. DoD.
20. MIL-STD-188-181B (1999) Department of Defense Interface Standard, Interoperability Standard for Single-Access 5-kHz and 25-kHz UHF Satellite Communications Channels, U.S. DoD.