Robust Group Key Agreement Using Short Broadcasts

Stanisław Jarecki
Department of Computer Science
University of California, Irvine
Irvine, CA 92697
stasio@ics.uci.edu

Jihye Kim
Department of Computer Science
University of California, Irvine
Irvine, CA 92697
jihyek@ics.uci.edu

Gene Tsudik
Department of Computer Science
University of California, Irvine
Irvine, CA 92697
gts@ics.uci.edu

ABSTRACT

A group key agreement protocol (GKA) allows a set of players to establish a shared secret key which can be used to secure a subsequent communication. Several efficient constant-round GKAs have been proposed. However, their performance degrades if some players fail during protocol execution. This is a problem in practice, e.g. for mobile nodes communicating over wireless media, which can loose connectivity during the protocol execution. Current constant-round GKA protocols are either efficient and non-robust or robust but not efficient: Assuming a reliable broadcast communication medium, the standard encryption-based group key agreement protocol can be robust against arbitrary number of node faults, but the size of the messages broadcast by every player is proportional to the number of players. In contrast, non-robust group key agreement can be achieved with each player broadcasting just constant-sized messages.

We propose a novel 2-round group key agreement protocol which tolerates up to T node failures using $O(T)$-sized messages, for any T. To exemplify the usefulness of this flexible trade-off between message size and fault tolerance, we show that the new protocol implies a fully-robust group key agreement with $O(\log n)$-sized messages and expected round complexity close to 2, assuming random node faults. The proposed protocol is secure under the (standard) Decisional Square Diffie-Hellman assumption.

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: General; C.4 [**Performance of Systems**]: Reliability; C.2.2 [**Network Protocols**]: Applications

General Terms

Algorithms, Reliability, Security

Keywords

group key agreement, fault tolerance

CCS'07, October 29–November 2, 2007, Alexandria, Virginia, USA.

1. INTRODUCTION

The growth of group applications triggers the need for group-oriented security mechanisms over insecure network channels. The applications include IP telephony, collaborative workspaces, secure conferences, as well as dynamic coalitions common in law enforcement and disaster rescue scenarios. Standard security services required in such group settings, e.g. confidentiality of group-wide broadcasts, can be very efficiently achieved if all group members share a group-wide secret key.

A group key agreement protocol (GKA) allows n players to create such shared secret key. There are several widely-known efficient constant-round group key agreement protocols [4, 8], but their performance degrades if some of the participating players fail during the protocol execution. This is a serious concern in practice, for example for mobile nodes that communicate over a wireless media, but which can loose connectivity during protocol execution.

Assuming a reliable broadcast medium, a GKA protocol can trivially be made robust to node failures by re-starting the protocol from scratch whenever a faulty player is detected. However, this would multiply all protocol costs by the number of faults, including the round complexity of the protocol. Robust *constant-round* GKA protocols can be achieved by executing parallel instances of any standard, i.e. non-robust, constant-round GKA protocol, one instance for every possible subset of non-faulty players. Such protocol would be robust and constant-round, but its communication and computation costs would grow by an inadmissible factor of 2^n. This gives rise to the question whether there exist constant-round GKA protocols that are robust to node failures at more reasonable efficiency costs.

Previous Work on Robust GKA Protocols. Amir, et al. [1] proposed the first robust GKA protocol based on a group key agreement protocol (called GDH) introduced by Steiner, et al. [9], and a view-based group communication system (GCS) which provides the abstraction of consistent group membership. Since the GCS can detect crashes among the players during the execution of a GDH protocol, the protocol can react accordingly. However, its round complexity is $O(n)$ and it requires $O(n^2)$ broadcasts.

Subsequently, Cachin and Strobl (CS) proposed a constant-round robust GKA protocol which works over asynchronous networks [5], and hence in particular tolerates both node and link failures. The exact communication and infrastructure assumptions of the CS protocol depend on the choice of the consensus subprotocol which the CS protocol invokes. However, assuming a reliable broadcast medium the CS pro-

tocol takes 2 rounds, and each player broadcasts $O(n)$-sized messages and makes $O(n)$ public key operations.[1]

Our Contributions. In this paper we investigate the issue of efficiency versus robustness to node failures, for constant-round GKA protocols working in a reliable broadcast communication medium. As mentioned above, assuming reliable broadcast, the 2-round CS protocol is robust against arbitrary number of node faults, but the size of the messages broadcast by each player is $O(n)$. In contrast, the group key agreement protocol of Burmester-Desmedt (BD) [4] uses only constant-sized messages, but is not robust to any node failures. In this paper we show how to achieve a natural trade-off between message size and the desired level of fault-tolerance in a GKA protocol. Namely, we propose a new 2-round GKA scheme which tolerates up to T node failures using $O(T)$-sized messages, for any T. The new protocol is secure in the standard model under the Decisional Square Diffie-Hellman assumption. To exemplify the usefulness of this flexible trade-off between message size and fault tolerance, we show that in a realistic setting of *random* node faults, this protocol implies a fully-robust GKA protocol with $O(\log n)$-sized messages and expected round complexity close to 2.

Organization. We present our protocols in a modular way: We start from a basic protocol which helps to understand how the proposed robust GKA protocol runs and why it is provably secure. We then modify this basic protocol using two bridge protocols, each of which improves the communication complexity of the previous one, leading to the flexible protocol that supports fault tolerance against up to T failures using $O(T)$-sized messages.

The paper is organized as follows: Section 2 discusses our communication and adversarial settings and presents a definition of a secure GKA protocol in these settings. Section 3 defines the cryptographic assumptions required by our constructions. Section 4 presents our robust GKA protocols. Section 5 compares performance of the proposed scheme with existing schemes, and Section 6 contains the detailed security proof of the protocols presented in Section 4.

2. SECURITY MODEL

Our security model is a standard model for Group Key Agreement protocols executed over authenticated links. Since the players in our GKA protocols do not use long-term secrets, we define GKA security (following [4, 6]), as semantic security of the session key created in a single instance of the GKA protocol executed among honest parties.

Authenticated Links. Our paper is concerned with Group Key Agreement (GKA) protocols in the *authenticated* links model. Note that there are standard and inexpensive compilation techniques which convert any group key agreement protocol into an *authenticated* group key agreement by (1) deriving a unique session-specific nonce at the beginning of the protocol and (2) having each player sign its message together with this nonce [6]. Moreover, while the generic compiler of [6] introduces an extra communication round into the protocol to establish a unique session-specific nonce before the GKA protocol starts, we point out that in the case of the protocols covered here this is unnecessary since the first message of the GKA protocol can be used to derive the unique session id, and hence the [6] compilation introduces no overhead apart of the unavoidable costs of issuing and verifying signatures.

Broadcast Communication and Player Failure. We assume that all communication within the protocol takes place over *reliable* (and authenticated) broadcast channel, where all the non-faulty players have the same view of the broadcasted message (which can be null if the sender is faulty). We assume weak synchrony, i.e., the players have synchronized clocks and execute the protocol in synchronized rounds, and the messages from the non-faulty players must arrive within some time window, which we assume is large enough to accommodate clock skews and reasonable communication delays. The assumption of reliable broadcast communication might be realistic for certain communication scenarios, e.g. Ethernet or wireless communication between close-by players. Otherwise, reliable broadcast must be implemented via a consensus protocol.

We assume an honest but curious adversary which can additionally impose arbitrary stop faults on the (otherwise honest) players participating in the protocol. In other words, we do *not* consider a Byzantine adversary. (We note, however, that using standard zero-knowledge proofs our protocols can easily be strengthened to tolerate malicious insiders at small constant factor increase in communication and computation cost.) Additionally, the adversary can make each player stop at an arbitrary moment in the protocol execution, but any such node failure cannot violate the contract imposed by the reliable broadcast assumption. Throughout the paper we assume that these stop faults are scheduled in *arbitrary* way by the adversary, except in the last section when we consider a weaker model of *random* faults which occur independently at every node with some fixed probability.

DEFINITION 1. *(**GKA Security**) Consider an adversary algorithm $\mathcal{A}$ which observes an execution of the GKA protocol between n honest players, and, depending on bit b, is given the session key computed by this protocol (if $b = 1$) or a value chosen at random from the same domain as the sessions keys (if $b = 0$). The adversary $\mathcal{A}$ outputs a single bit b'. We define adversary's advantage in attacking the GKA protocol as:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{GKA}} = |\Pr[b' = b] - 1/2|$$

where the probability goes over the random execution of the protocol, the adversary $\mathcal{A}$, and the random choice of bit b.

*We call a GKA protocol (ϵ, t)-*secure *if for all adversaries $\mathcal{A}$ who run in time t it holds that $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{GKA}} \leq \epsilon$.*

3. CRYPTOGRAPHIC SETTING

Let $\mathbb{G}$ be a cyclic group of prime order q, and let g be its generator. We assume the DDH and Square-DDH problems are hard in $\mathbb{G}$. For example, $\mathbb{G}$ could be a subgroup of order q in the group of modular residues $\mathbb{Z}_p^*$ s.t. $p-1$ divides q, $|p| = 1024$ and $|q| = 160$, or it can be a group of points on an elliptic curve with order q for $|q| = 160$. For more examples of groups where DDH and square-DDH assumptions are assumed to hold see [2].

[1] Assuming reliable broadcast, the CS protocol works as follows: First every player broadcasts its public encryption key. Then every player picks its contribution to the shared key, encrypts it under each broadcasted public key, and broadcasts a message containing the resulting n ciphertexts. The shared key is computed by each player as the sum of all broadcasted contributions.

DEFINITION 2. *The DDH problem is (ϵ, t)-hard in $\mathbb{G}$ if for every algorithm A running in time t we have:*

$$\left| \Pr[x, y \leftarrow \mathbb{Z}_q : A(g, g^x, g^y, g^{xy}) = 1] - \Pr[x, y, z \leftarrow \mathbb{Z}_q : A(g, g^x, g^y, g^z) = 1] \right| \leq \epsilon$$

DEFINITION 3. *The Square-DDH problem is (ϵ, t)-hard in $\mathbb{G}$ if for every A running in time t we have:*

$$\left| \Pr[x \leftarrow \mathbb{Z}_q : A(g, g^x, g^{x^2}) = 1] - \Pr[x, z \leftarrow \mathbb{Z}_q : A(g, g^x, g^z) = 1] \right| \leq \epsilon$$

4. ROBUST GROUP KEY AGREEMENT PROTOCOLS

We describe our two-rounds robust GKA protocol that tolerates T faults with $O(T)$-sized messages, in three steps: In Sections 4.1 and 4.2, solely for presentation purposes, we explain how the non-robust GKA protocol of Burmester-Desmedt (BD) [4] generalizes to a *(fully) robust* 2-round GKA protocol at the cost of increasing the length of the constant-sized messages of the BD protocol to $O(n^2)$-sized messages. We call this robust generalization of the BD protocol *BD-RGKA* and show that the protocol remains secure under the same DDH assumption required for the underlying BD protocol. Next, in Section 4.3, using the technique of node-doubling we show that the *BD-RGKA* protocol can be modified to retain full robustness with message size reduced to $2n$ group elements. Moreover, with randomness re-use we can further reduce the message size to just n group elements per player. We call the resulting protocol *RGKA* and show that it is secure under the Square-DDH assumption. This leads to our main contribution, the *T-RGKA* protocol shown in Section 4.4.1, which is a version of the above *RGKA* protocol in which each player broadcasts only $2T$ group elements. This protocol remains secure under the same Square-DDH assumption, but its resilience is reduced to $O(T)$ faults. (More precisely, the *T-RGKA* protocol tolerates all faults except two separate sequences of T or more consecutive faults.) Finally, we exemplify the usefulness of the efficiency vs. fault-tolerance trade-off offered by the *T-RGKA* protocol by showing that it implies a fully robust GKA protocol with $2 + \delta$ expected rounds and messages of size $O(\log n + \log(1/\delta))$, if the node faults are random and occur at a constant rate.

4.1 Overview: Adding Robustness to Burmester-Desmedt GKA

Since our fault-tolerant protocol is based on the GKA protocol proposed by Burmester and Desmedt (BD) [4], we need to first overview the BD GKA protocol to describe our modifications of it. The BD GKA protocol proceeds in two rounds (see Figure 1): First each player P_i broadcasts a public counterpart $z_i = g^{t_i}$ of its contribution t_i to the key. In the second round each P_i broadcasts $X_{[i-1,i,i+1]} = g^{t_i t_{i+1} - t_{i-1} t_i}$ (which it can compute as $X_{[i-1,i,i+1]} = (z_{i+1}/z_i)^{t_i}$). Given the set of values $X_{[n,1,2]}, X_{[1,2,3]}, ..., X_{[n-1,n,1]}$, each player P_i can use its contribution t_i to locally compute the common session key $\mathsf{sk} = g^{t_1 t_2 + t_2 t_3 + ... + t_n t_1}$.

We will call value $X_{[i-1,i,i+1]}$ a *gadget*, the $t_i t_{i+1}$ part of its exponent the *left hand*, and the $t_{i-1} t_i$ part of the exponent, which is multiplied by minus one, the *right hand* of this gadget. A gadget $X_{[i-1,i,i+1]}$ corresponds to a path of length two connecting nodes P_{i-1}, P_i, and P_{i+1}. Using this graph terminology, we say that two gadgets are *connectable* if the left hand of one gadget is the same as the right hand of the other. For example, for every i, gadgets $X_{[i-1,i,i+1]}$ and $X_{[i,i+1,i+2]}$ are connectable. We say that a sequence of gadgets *forms a path through the graph* if each two consecutive gadgets in the sequence are connectable. By inspecting the formula for deriving the secret key in the BD GKA protocol we can observe that each player derives the same key because the set of gadgets broadcasted in the second round of the protocol forms a Hamiltonian cycle (a.k.a. a "circular path") through the graph of all players. In Figure 2 we show an example of four gadgets $X_{[4,1,2]}$, $X_{[1,2,3]}$, $X_{[2,3,4]}$, and $X_{[3,4,1]}$, created by the BD GKA protocol executed in a group of four players.

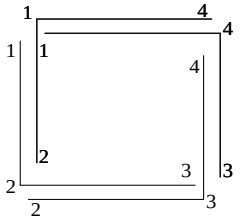
[**Round 1**]:
Each player P_i picks a random $t_i \in Z_q$ and broadcasts $z_i = g^{t_i}$.

[**Round 2**]:
Each P_i broadcasts its gadget value $X_{[i-1,i,i+1]} = (z_{i+1}/z_{i-1})^{t_i}$, where the indices are taken in a cycle.

[**Key Computation**]:
Each P_i computes the key as $\mathsf{sk}_i = (z_{i-1})^{n t_i} \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdots X_{i-2}$, where $X_i = X_{[i-1,i,i+1]}$.

(Note that for all i we have $\mathsf{sk}_i = g^{t_1 t_2 + t_2 t_3 + ... + t_n t_1}$.)

Figure 1: Burmester-Desmedt's Group Key Agreement Protocol (BD GKA)



Gadgets $X_{[4,1,2]}$, $X_{[1,2,3]}$, $X_{[2,3,4]}$, $X_{[3,4,1]}$, sent in a BD GKA protocol involving four players, form a Hamiltonian cycle in the graph of four nodes.

Figure 2: Gadgets in a BD GKA Protocol for $n = 4$

Idea for Adding Robustness to the BD GKA Protocol. The reason why the BD GKA protocol is not robust against any single fault is that missing a gadget would break the circular path in the graph of players, and without a sequence of connectable gadgets which covers the set of all nodes it's not clear how to compute a common secret key. However, such a circular path would always exist if in the second round the players sent out *additional* gadget values in such a way that even if some players exhibit faults in the broadcast stage the gadgets broadcasted by the alive players can be ordered so that they form a circular path through all alive players. This indeed would trivially be the case if instead of broadcasting just the $X_{[i-1,i,i+1]}$ gadget, each player

P_i broadcasted n^2 gadgets $X_{[k,i,j]}$ for all k, j in $[1, ..., n]$: For any set of alive players which complete this broadcast round, one can form a circular path which transverses all of them from the gadgets broadcast by these players.

4.2 Robust GKA with $O(n^2)$ Message Size

We show the GKA protocol which follows the above idea, denoted *BD-RGKA*, in Figure 3. The protocol is robust against any set of faults, and it remains secure under the same DDH assumption used by the basic BD GKA protocol. In other words, broadcasting all the additional information in the second round does not diminish the security of the protocol.

Note that in the BD GKA protocol the session key $\mathsf{sk} = g^{t_1t_2+t_2t_3+...+t_nt_1}$ is computed according to a fixed circular order among the participating players while in the *BD-RGKA* protocol the session key is computed as $\mathsf{sk} = g^{t_{a_1}t_{a_2}+ +t_{a_2}t_{a_3}+...+t_{a_m}t_{a_1}}$, where $P_{a_1}, ..., P_{a_m}$ are players which remain alive after the second broadcast round. Note that since we assume reliable broadcast and synchrony, each player has the same view of the list of alive players and their messages. The alive players are ordered s.t. $a_1 < a_2 < ... < a_m$, but this order is arbitrary: It can be determined by player id's, but it can also be determined, for example, according to the z_i values sent in the first round of the protocol.

[**Round 1**]:

1.1 Each P_i picks a random $t_i \in Z_q$ and broadcasts $z_i = g^{t_i}$.

[**Round 2**]:

2.1 Let ActiveList be the list of indices of all players who complete Round 1.

2.3 Each P_i computes $X_{[k,i,j]} = (z_j/z_k)^{t_i}$ for all pairs (k, j) s.t. $k, j \in \mathsf{ActiveList}$ and $k \neq j$.

2.3 Each P_i broadcasts $\{X_{[k,i,j]}\}_{k,j\in\mathsf{ActiveList}}$.

[**Key Computation**]:

3.1 Let ActiveList be the list of indices of all players who complete Round 2.

3.2 Each P_i sorts the players in ActiveList in the same order; wlog, we assume that the live players are ordered as $\{P_{a_1}, ..., P_{a_m}\}$, where $m \leq n$.

3.3 Each P_{a_i} computes the session key $\mathsf{sk}_{a_i} = (z_{a_{i-1}})^{m\cdot t_{a_i}} \cdot X_{a_i}^{m-1} \cdot X_{a_{i+1}}^{m-2} \cdot \cdots \cdot X_{a_{i-2}}$, where $X_{a_i} = X_{[a_{i-1},a_i,a_{i+1}]}$.

(Note that $\mathsf{sk}_{a_i} = g^{t_{a_1}t_{a_2}+t_{a_2}t_{a_3}+...+t_{a_m}t_{a_1}}$.)

Figure 3: The *BD-RGKA* Protocol: Robust GKA with $O(n^2)$-sized Messages

For the proof of the following theorem, see Section 6.

THEOREM 1. *Assuming that the DDH problem is $(\epsilon_{\mathsf{ddh}}, t_{\mathsf{ddh}})$-hard in group $\mathbb{G}$, the* BD-RGKA *protocol is a (ϵ', t')-secure Group Key Agreement for $\epsilon' \leq n^2 \cdot \epsilon_{\mathsf{ddh}}$ and $t' = t_{\mathsf{ddh}} - O(n^2) \cdot t_{ex}$, where t_{ex} is the cost of exponentiation in $\mathbb{G}$.*

4.3 Robust GKA with $O(n)$ Message Size

Step 1: $n^2 \rightarrow 2n$ Reduction by Node-Doubling. The *BD-RGKA* protocol achieves full robustness by increasing the message size by a factor of n^2, but this overhead can be reduced to the factor of n as follows. In the *BD-RGKA* protocol every player P_i created n^2 separate gadgets $X_{[k,i,j]}$, for every pair P_k, P_j of a possible left and a possible right neighbor in the cycle that eventually spans all the players remaining alive. We can decrease the number of gadgets created by every player to $2n$ and yet handle any possible pattern of node failure if we split each network node into two nodes, and we ask every player P_i to operate on behalf of two consecutive nodes U_{2i-1} and U_{2i}. The gadgets P_i creates correspond to n gadgets of the form $X_{[2k,2i-1,2i]}$, for all k, which are made "on behalf of" node U_{2i-1}, and n gadgets of the form $X_{[2i-1,2i,2j-1]}$, for any j, made "on behalf of" node U_{2i}. In other words, all gadgets made for U_{2i-1} have U_{2i} fixed as their right neighbor and all gadgets made for U_{2i} have U_{2i-1} fixed as their left neighbor. In this way, the protocol can still handle any set of player failures because, for any i, nodes U_{2i-1} and U_{2i} either both fail or they are both alive, since both are operated by the same player P_i.

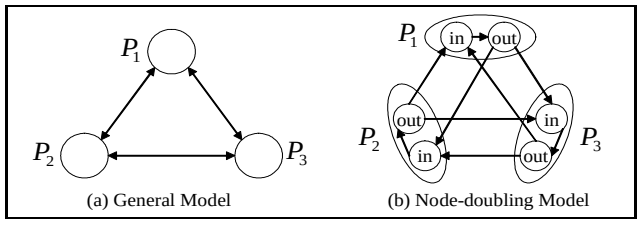


Figure 4: Two different models of fully connected network for 3 players.

Figure 4 shows pictorially how this *node-doubling* technique works. We show the edges between the nodes as directed edges, and every pair of edges (P_k, P_i) and (P_i, P_j) corresponds to a gadget $X_{[k,i,j]}$. In part (a) we see the regular graph made by 3 players each operating a single node, and in part (b) we see the same set of players but now each player P_i operates two nodes, the **in** node U_{2i-1} and the **out** node U_{2i}. Each **in** and **out** pair within the same player is connected by an edge, and each **in** node has $n-1$ incoming edges while each **out** node has $n-1$ outgoing edges. Observe that one can form a Hamiltonian cycle connecting any set of nodes $P_{a_1}, ..., P_{a_m}$ using a proper set of these directed edges. Therefore the players can perform the *BD-RGKA* protocol where (1) each player P_i plays the part of two consecutive nodes U_{2i-1} and U_{2i}, and (2) in step (2.3) of the protocol player P_i broadcasts a restricted set of gadgets, namely $X_{[2k,2i-1,2i]}$ and $X_{[2i-1,2i,2j-1]}$, for each k and j. Thus there's only $2(n-1)$ gadgets broadcasted by each player, and the protocol remains robust against any pattern of player failures, while the security of this protocol is implied by the security of the *BD-RGKA* protocol, since the latter reveals strictly more information.

Step 2: Re-using the Secret Contributions. We can reduce the message size of the above protocol by a factor of two by having the two virtual nodes U_{2i-1} and U_{2i} use the same secret contributions $t_{2i-1} = t_{2i}$. This change implies that the gadgets created for the **in** nodes are inverses of the corresponding gadgets created for the **out** nodes. If we denote the indices of two internal nodes administered by P_i as i and i' (instead of $2i-1$ and $2i$, respectively) and both use the

same contribution, i.e, $t_i = t_{i'}$, then we have $X_{[i',i,2k-1]} = (X_{[2k,i',i]})^{-1}$ because $X_{[i',i,2k-1]} = g^{t_i t_{2k-1} - t_i t_{i'}} = (g^{t_i t_{i'} - t_i t_{2k-1}})^{-1} = (X_{[2k-1,i,i']})^{-1}$, but the last term is equal to $(X_{[2k,i,i']})^{-1}$ because player P_k also sets $t_{2k-1} = t_{2k}$. We show the resulting protocol *RGKA* in Figure 5, describing only the substeps which are different from *BD-RGKA* in Figure 3. In the protocol description, the indices of node **in** and **out** in P_i are i, i', respectively.

[**Round 1**]: same as in *BD-RGKA* in Figure 3.

[**Round 2**]: same as in *BD-RGKA* in Figure 3, except:

2.3 Each P_i broadcasts $X_{[k,i,i']} = (z_i/z_k)^{t_i}$ for all $k \in \mathsf{ActiveList}$. Define $X_{[i,i',k]}$ as $(X_{[k,i,i']})^{-1}$.

[**Key Computation**]: same as in *BD-RGKA*, except:

3.3 Each P_{a_i} computes $\mathsf{sk}_{a_i} = (z_{a_{i-1}})^{m \cdot t_{a_i}} \cdot X_{a_i}^{m-1} \cdot X_{a_{i+1}}^{m-2} \cdot \cdots \cdot X_{a_{i-2}}$ as in the *BD-RGKA* protocol, but here X_{a_i} is defined as $X_{a_i} = X_{[a_{i-1},a_i,a_{i'}]} \cdot X_{[a_i,a_{i'},a_{i+1}]}$.

(Note that the resulting key is exactly as in *BD-RGKA* protocol because $X_{a_i} = g^{t_{a_i} t_{a_{i+1}} - t_{a_{i-1}} t_{a_i}}$.)

Figure 5: The *RGKA* Protocol: Robust GKA with $O(n)$-sized Messages

The adversary's view of the *RGKA* protocol is similar to a subset of adversary's view of the *BD-RGKA* protocol, but it is not the same because in this version of the *BD-RGKA* protocol the contributions of two consecutive nodes are equal. Therefore the new protocol requires a separate security argument to show that this pattern of correlations between players' contributions does not impinge on the security of the agreed-on key, and we show that the protocol is indeed secure, but under the Square-DDH assumption instead of the standard DDH assumption. For the proof of the following theorem, see Section 6.

THEOREM 2. *Assuming that the Square-DDH problem is (ϵ, t)-hard in group $\mathbb{G}$, protocol* RGKA *is a (ϵ', t')-secure Group Key Agreement for $\epsilon' \leq (n^2 + n) \cdot \epsilon$ and $t' = t - O(n^2) \cdot t_{ex}$, where t_{ex} is the cost of exponentiation in $\mathbb{G}$.*

4.4 Further Reduction of Message Size

In this section we show two robust GKA protocols, *T-RGKA* and *RGKA*'. The first protocol, *T-RGKA*, is the main contribution of this paper, and it is a 2-round protocol with $O(T)$-sized messages which tolerates up to T failures for any $T < n$. (In fact, it tolerates a much larger class of failures, as explained below.) Second, we use this T-robust protocol to construct a *fully* robust GKA protocol *RGKA*', with $O(\log n + \log(1/\delta))$ communication complexity and with expected $2 + \delta$ number of rounds, assuming the *random fault model* where each player can fail with some fixed constant probability ν. The *RGKA*' protocol simply repeats the *T-RGKA* protocol, with T set to approximately $(\log n + 1/2 \log(1/\delta))/(\log(1/\nu))$, until the *T-RGKA* protocol succeeds. Intuitively, T is set in such a way that the chance that the *T-RGKA* protocol fails is about $1/\delta$, so that the expected number of protocol repetitions until success is δ, resulting in $2 + \delta$ expected number of rounds.

4.4.1 T-robust GKA with $O(T)$ Message Size

Here we describe a *partially robust* GKA protocol *T-RGKA* which has $O(T)$-sized messages and tolerates all patterns of player faults *except* if there are two separate sequences of consecutive players who all fail and both sequences have at least T players. Thus in particular the protocol is secure against any set of $2T$ failures. The *T-RGKA* protocol is a very simple modification of the *RGKA* protocol of the previous section: The *T-RGKA* protocol proceeds just like *BD-RGKA*, except each player P_i broadcasts only T gadgets instead of $n - 1$, namely it broadcasts only gadgets $X_{[k,i,i']}$ for $|k - i| \leq T$, instead of for all k. (We assume, for simplicity of notation, that all players successfully pass the first round of the protocol.) Using graph terminology, the gadgets created by each player form a partially connected graph instead of a fully connected graph. More precisely, the resulting graph is the *T-th power of circle* among n nodes, denoted C_n^T:

DEFINITION 4. *The $\mathbf{T}$th power of a graph G, denoted G^T, is a transitive closure of graph G applied T times. Namely, $G^1 = G$, and G^T is defined recursively for every $T \geq 2$ as follows: $(u, v) \in G^T$ if and only if either $(u, v) \in G^{T-1}$ or there exists w s.t. $(u, w) \in G^{T-1}$ and $(w, v) \in G$. Hence the $\mathbf{T}$th power of a circle, C_n^T is defined as $C_n^T = \{(i, j)\ s.t.\ |j - i| \leq T\}$.*

For example, here are graphs C_{10}^1, C_{10}^2, and C_{10}^3:

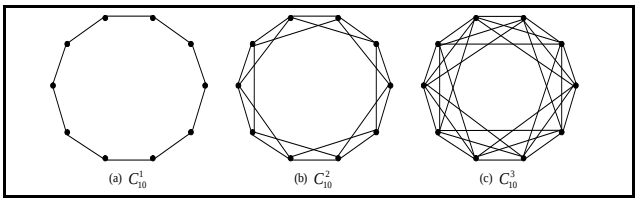


Figure 6: Examples of the T-th power of a circle.

In the Tth power of a circle, there is always a circular path that connects all alive players unless T consecutive players fail. For example, assuming the players are indexed as 1,...,10 in order, if players 1, 2, 5, and 7 are faulty in C_{10}^3 then one of possible circular paths connecting the remaining nodes is $3 - 4 - 6 - 8 - 9 - 10 - 3$.

However, the resulting algorithm in fact can handle a larger class of failures. Namely, it can withstand all failures except of *two* separate sequences of consecutive T (or more) failures. To enable this stronger robustness we relax the way the key is computed, so that the key is associated not necessarily with a Hamiltonian cycle but just with a Hamiltonian path through the graph. To make minimal changes to the algorithm, we can identify Hamiltonian paths with cycles in which every node is visited twice: once in the forward direction and then once more on the way back. For example, if 4 consecutive players 1, 2, 3, and 4 are faulty in C_{10}^1, there is a Hamiltonian path $5 - 6 - 7 - 8 - 9 - 10$ among the remaining players, and the corresponding "cycle" is $5 - 6 - 7 - 8 - 9 - 10 - 9 - 8 - 7 - 6 - 5$. This change enables robustness against a larger class of failures because given any subset of alive players S in the Tth power of a circle C_n^T, there is always a Hamiltonian path that connects the players in S except when the set of faulty players $C_n^T \setminus S$ includes *two* separate sequences of nodes, each containing T

(or more) nodes. We depict the resulting *T-RGKA* protocol in Figure 7.

[**Round 1**]: same as in *RGKA* in Figure 5.

[**Round 2**]: same as in *RGKA* in Figure 5 except:

2.3 Each P_i broadcasts $X_{[k,i,i']} = (z_i/z_k)^{t_i}$ for T nearest neighbors to the right and T nearest neighbors to the left among players $k \in$ ActiveList.

[**Key Computation**]: same as in *RGKA* in Figure 5, except that the cycle through the alive nodes can be constructed either from a true Hamiltonian cycle or from a Hamiltonian path taken twice (see an example in the text above). Wlog, we assume that the path is formed as $\{P_{a_1}, P_{a_2}, \cdots, P_{a_m}\}$, where for some i, j we can have $a_i = a_j$.

Figure 7: The *T-RGKA* Protocol: T-robust GKA with $O(T)$-Sized Messages

The security of the *T-RGKA* protocol which uses a Hamiltonian cycle to compute the key is implied by the security argument for the *BD-RGKA* protocol. If the *T-RGKA* protocol constructs a key by making an artificial cycle from a Hamiltonian path taken twice, as explained above, a very similar security argument still applies: The only difference is that the resulting key contains each contribution of the form $t_{a_i} t_{a_{i+1}}$ twice, but that corresponds to squaring the key that is created if a true Hamiltonian path is used instead. Since squaring in a prime-order group is a permutation of the group elements, the security argument given for the *BD-RGKA* scheme implies the security of the key computed in this way as well.

4.4.2 Fully Robust GKA Protocol with $O(\log n)$ Messages in the Random Fault Model

In this section, we show another robust GKA protocol, called *RGKA*', which is fully robust but *not* constant-round. *RGKA*' simply repeats the *T-RGKA* protocol above, with some parameter T, which we fix below, until *T-RGKA* succeeds. (In fact, only the second round of the *T-RGKA* protocol needs to be repeated, since the security of the *BD-RGKA* protocol implies that the same contribution t_i can be used in all these instances of the *T-RGKA* protocol.) Repeating the protocol increases the number of rounds and the protocol communication complexity. However, we will argue that if the faults are random and occur with rate ν, then for any parameter δ, the expected number of rounds in the *RGKA*' protocol can be $2+\delta$, and the expected communication complexity per player can be $2(T+\delta)$ group elements, for $T = O((\log n + \log(1/\delta))/\log(1/\nu))$. Assuming that the node faults are random and that they are independent of each other might seem unrealistic, but recall that the order among the participating players can be determined by the messages sent in the first round of the protocol, and therefore the usual dependencies between failures of nodes which are physical neighbors do not apply to the neighbors in the logical order created by in the protocol.

We claim that if we set $T \approx (\log n + 1/2 \log(1/\delta))/\log(1/\nu)$ then the expected number of rounds in *RGKA*' is $2+\delta$. Since a *T-RGKA* protocol succeeds exists except if at least two sequences of T consecutive nodes fail, the probability that a single execution of the *T-RGKA* protocol fails is upper-bounded by

$$f \leq n^2/2 * \nu^{2T} \quad (1)$$

The expected number of rounds is then $\delta = 2(1/(1-f)) - 2 = 2f/(1-f) \approx 2f$. Therefore by equation 1, we can upper-bound threshold T necessary to achieve parameters δ and ν as $T \leq (\log n + 1/2 \log(1/\delta))/\log(1/\nu)$.

5. EFFICIENCY ANALYSIS

We first summarize the relevant aspects of protocol efficiency.

Performance Criteria.

- **Resilience:** the number or pattern of faults that the protocol tolerates.
- **Round Complexity:** the number of rounds.
- **Communication Complexity:** the (expected) total bit-length of all messages sent in the protocol. (Since we assume a broadcast communication medium, we measure the bit-length of messages sent over a broadcast channel.)
- **Computational Complexity:** the amount of computation that must be performed *per player* in the protocol. We will restrict ourselves to counting only the number of cryptographic operations (e.g. exponentiations and public-key operations) since these operations dominate the computational cost.

We compare the protocols we propose with non-robust BD protocol [4] and the encryption-based group key agreement protocol - the simplified version of CS protocol [5]. (Refer to Section 1.) Table 1 compares efficiency of the BD, the encryption-based GKA, denoted by "CS", and the *BD-RGKA*, *RGKA*, *T-RGKA*, and *RGKA*' protocols shown in the previous section. Of these six protocols, BD is not robust against even a single failure, *T-RGKA* is robust against at least $2T$ failures (see subsection 4.4.1 above for the exact robustness condition), and all others are fully robust. The costs of all the protocols *except RGKA*' are independent of the failure pattern, while the efficiency parameters given for the *RGKA*' protocol are expected values, where the expectation is taken given *random* player faults, occurring at some fixed rate given by constant ν.

Throughout the comparison we assume the same communication model of reliable broadcast and weak synchrony. We denote the number of players participating in the protocol as n. In measuring the broadcast communication complexity, we use $\mathbf{t}$, the security parameter, to denote the size of a single group element and/or a public-key ciphertext, since in practice these are comparable. For the computation costs, **pk** stands for public key operations, such as encryption and decryption, and **ex** stands for an exponentiation operation in group $\mathbb{G}$.

The conclusion we'd like to draw from this comparison is the following. First of all, all protocols run in two rounds, and *RGKA*' runs in expected $2+\delta$ rounds, for any δ, if T is set as $O(\log n + \log(1/\delta))$. (See Section 4.4.2 above for more discussion.) Given the comparable round complexities, the remaining important criterion is communication complexity. (It is also computational complexity per player, but as the table shows, the latter follows the communication very

	Rounds	Communication	Computation
BD	2	$2n\mathbf{t}$	3 **ex**
CS	2	$(n+n^2)\mathbf{t}$	$2n$ **pk**
BD-RGKA	2	$n^3\mathbf{t}$	n^2 **ex**
RGKA	2	$n^2\mathbf{t}$	n **ex**
T-RGKA	2	$(1+2T)n\mathbf{t}$	$(2+2T)$ **ex**
RGKA'	$2+\delta$	$O(\log n, \log(1/\delta))n\mathbf{t}$	$O(\log n)$ **ex**

Table 1: Complexity Comparison between provably secure protocols for robust GKA protocols.

closely). In this aspect, the $O(\log n)$ communication complexity of the *RGKA*' protocol is vastly superior over the $O(n)$ complexity of the CS protocol. In exact terms, the communication complexity of the *RGKA*' protocol is very close to $2T+1$ group elements, where T is approximately $(\log n + \log(1/\delta))/\log(1/\nu)$. For example, if the player failure probability ν is bounded by 0.1 and $n = 10^3$, the *T-RGKA* protocol works successfully with probability at least 99% (in which case the expected number of rounds in *RGKA*' is about 2.01), with only $T = 4$. If $\nu = 0.01$ and everything else is the same then already $T = 2$ is enough. If the player failure probability is very high, e.g. $\nu = 0.5$, our T grows only to $T = 13$, and the message size of $2T$ group elements is still very small compared to the $O(n)$ message sizes incurred by the CS protocol.

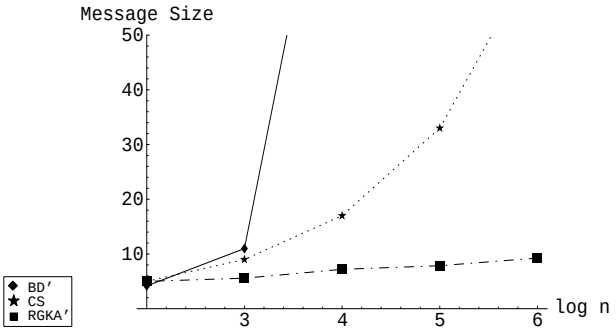


Figure 8: Message Size Comparison with $\nu = 1/4$

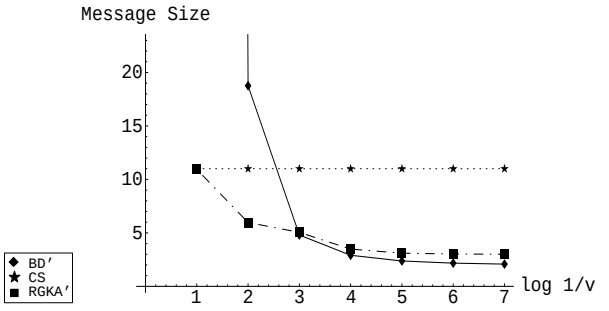


Figure 9: Message Size Comparison with $n = 10$

We also compare *RGKA*' with CS and repeated BD, called BD', for (n, ν) pairs. BD' simply repeats the BD protocol until BD succeeds. In particular, we extract T value optimal in message size for each (n, ν) pair and compare the message size. We do not compare round complexities but the expected number of rounds in the *RGKA*' protocol is at most about 2.1 for optimal T values, assuming player fault

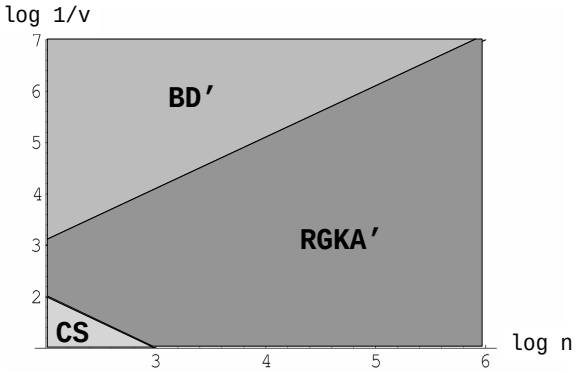


Figure 10: Best Scheme from the point of view of Communication Complexity

rate ν of at most 1/2. On the other hand, for large ν and/or n values, the round complexity of the BD' protocol can be very high.

Figure 8 shows the message sizes of each protocol for different n's and a fixed ν, taking $\nu = 1/4$ as an example. The message size in BD significantly increases, that in CS less, and that in *RGKA*' is the least. Figure 9 shows the message sizes of each protocol for different ν's and a fixed n, taking $n = 10$ as an example. BD' provides slightly better communication complexity than *RGKA*' for low rate of faults, but its performance significantly degrades for higher fault rates.

Figure 10 shows which protocol outperforms the others in terms of communication complexity, for different (n, ν) parameters. CS has the smallest message size only if ν is very high while BD' outperforms the other schemes at very low fault rates. However, even there the *RGKA*' protocol has only slightly higher message complexity, while it beats BD's in expected round complexity.

6. SECURITY PROOFS

In this section we show the proofs of the security claims about the robust GKA protocols presented in Section 4.

6.1 Theorem 1: Security of the *BD-RGKA* Protocol

PROOF. Consider adversary's view of a single execution of the *BD-RGKA* protocol among n players. Let $\{P_{a_1}, ..., P_{a_m}\}$ be the set of live players in the final key computation step. The joined distribution of the transcript T of the protocol and the resulting session key sk is as follows. (To simplify notation we define $X_{a_i} = X_{a_{i-1},a_i,a_{i+1}}$.)

$\mathsf{Real} =$

$$\left\{ \begin{array}{c} t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{2,1,3} = \frac{g^{t_3 t_1}}{g^{t_2 t_1}},....,X_{n-2,n,n-1} = \frac{g^{t_{n-1} t_n}}{g^{t_{n-2} t_n}} \\ \mathsf{T} = (z_1,...,z_n, X_{2,1,3},...,X_{n-2,n,n-1}) \\ \mathsf{sk} = (g^{t_{a_1} t_{a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \\ \\ : (\mathsf{T}, \mathsf{sk}) \end{array} \right\}.$$

Consider the following modified distribution Fake_1, which is like Real, except that all occurrences of $t_1 t_2$ are replaced by $c_{1,2}$.

$\mathsf{Fake}_1 =$

$$\left\{ \begin{array}{c} c_{1,2}, t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{2,1,3} = \frac{g^{t_3 t_1}}{g^{c_{1,2}}},....,X_{n-2,n,n-1} = \frac{g^{t_{n-1} t_n}}{g^{t_{n-2} t_n}} \\ \mathsf{T} = (z_1,...,z_n, X_{2,1,3},...,X_{n-2,n,n-1}) \\ \mathsf{sk} = \left\{ \begin{array}{ll} (g^{c_{a_1,a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} & \text{if } (a_1,a_2) = (1,2); \\ (g^{t_{a_1} t_{a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} & \text{Otherwise.} \end{array} \right. \\ \\ : (\mathsf{T}, \mathsf{sk}) \end{array} \right\}.$$

A standard reduction argument shows that for any algorithm $\mathcal{A}'$ running in time $t_{\mathsf{ddh}} - O(n^2) \cdot \epsilon_{\mathbf{ex}}$ we have:

$|\Pr[(\mathsf{T}, \mathsf{sk}) \leftarrow \mathsf{Real} : \mathcal{A}'(\mathsf{T}, \mathsf{sk}) = 1] - \Pr[(\mathsf{T}, \mathsf{sk}) \leftarrow \mathsf{Fake}_1 : \mathcal{A}'(\mathsf{T}, \mathsf{sk}) = 1]| \leq \epsilon_{\mathsf{ddh}}$.

We next make the following additional modification Fake_2, which is like Fake_1, except that all occurrences of $t_1 t_3$ are replaces by $c_{1,3}$.

$\mathsf{Fake}_2 =$

$$\left\{ \begin{array}{c} c_{1,2}, c_{1,3}, t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{2,1,3} = \frac{g^{c_{1,3}}}{g^{c_{1,2}}},....,X_{n-2,n,n-1} = \frac{g^{t_{n-1} t_n}}{g^{t_{n-2} t_n}} \\ \mathsf{T} = (z_1,...,z_n, X_{2,1,3},...,X_{n-2,n,n-1}) \\ \mathsf{sk} = \\ \left\{ \begin{array}{l} (g^{c_{a_1,a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \text{ if } (a_1,a_2) = (1,2) \text{ or } (1,3); \\ (g^{t_{a_1} t_{a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \text{ Otherwise.} \end{array} \right. \\ \\ : (\mathsf{T}, \mathsf{sk}) \end{array} \right\}.$$

Again, a standard argument shows that for any algorithm $\mathcal{A}'$ running in time $t_{\mathsf{ddh}} - O(n^2) \cdot \epsilon_{\mathbf{ex}}$ we have:

$|\Pr[(\mathsf{T}, \mathsf{sk}) \leftarrow \mathsf{Fake}_1 : \mathcal{A}'(\mathsf{T}, \mathsf{sk}) = 1] - \Pr[(\mathsf{T}, \mathsf{sk}) \leftarrow \mathsf{Fake}_2 : \mathcal{A}'(\mathsf{T}, \mathsf{sk}) = 1]| \leq \epsilon_{\mathsf{ddh}}$.

Proceeding in the similar way, since the number of all possible DH tuples is $n(n-1)/2$, we acquire the following distribution:

$\mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2} =$

$$\left\{ \begin{array}{c} c_{1,2}, c_{1,3},..., c_{n-2,n}, c_{n-1,n}, t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{2,1,3} = \frac{g^{c_{1,3}}}{g^{c_{1,2}}},....,X_{n-2,n,n-1} = \frac{g^{c_{n-1,n}}}{g^{c_{n-2,n}}} \\ \mathsf{T} = (z_1,...,z_n, X_{2,1,3},...,X_{n-2,n,n-1}) \\ \mathsf{sk} = (g^{c_{a_1,a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \\ \\ : (\mathsf{T}, \mathsf{sk}) \end{array} \right\}.$$

It follows, by the standard hybrid argument, for any $\mathcal{A}'$ running in time $t_{\mathsf{ddh}} - O(n^2) \cdot \epsilon_{\mathbf{ex}}$ we have:

$$|\Pr[(\mathsf{T}, \mathsf{sk}) \leftarrow \mathsf{Real} : \mathcal{A}'(\mathsf{T}, \mathsf{sk}) = 1] - \Pr[(\mathsf{T}, \mathsf{sk}) \leftarrow \mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2} : \mathcal{A}'(\mathsf{T}, \mathsf{sk}) = 1]| \leq \frac{n(n-1)}{2} \cdot \epsilon_{\mathsf{ddh}}. \quad (2)$$

In experiment $\mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2}$, the values $c_{1,2}$,...,$c_{n-1,n}$ are constrained by T according to the following $n(n-1)(n-2)/2$ equations:

$$\begin{array}{c} \log_g X_{2,1,3} = c_{1,3} - c_{1,2} \\ \log_g X_{2,1,4} = c_{1,4} - c_{1,2} \\ \log_g X_{2,1,5} = c_{1,5} - c_{1,2} \\ \vdots \\ \log_g X_{n-3,n,n-2} = c_{n-2,n} - c_{n-3,n} \\ \log_g X_{n-3,n,n-1} = c_{n-1,n} - c_{n-3,n} \\ \log_g X_{n-2,n,n-1} = c_{n-1,n} - c_{n-2,n} \end{array}$$

FACT 1. *In the above set of equations, there are at most $n(n-1)/2 - 1$ linearly independent equations.*

PROOF. Set $n(n-1)(n-2)/2$ to be l_r and $n(n-1)/2$ to be l_c. We rewrite the set of the above l_r equations using a single $l_r \times l_c$ matrix equation:

$$b = Ax$$

where $b = (\log_g X_{2,1,3}, \log_g X_{2,1,4}, \cdots, \log_g X_{n-2,n,n-1})$, $x = (c_{1,2}, c_{1,3}, \cdots, c_{n-2,n}, c_{n-1,n})$,

$$A = \begin{pmatrix} -1 & 1 & 0 & 0 & \cdots & 0 \\ -1 & 0 & 1 & 0 & \cdots & 0 \\ -1 & 0 & 0 & 1 & \cdots & 0 \\ & & \vdots & & & \\ 0 & \cdots & -1 & 0 & 1 & 0 \\ 0 & \cdots & -1 & 0 & 0 & 1 \\ 0 & \cdots & 0 & 0 & -1 & 1 \end{pmatrix}.$$

To compute the number of linearly independent equations, we compute the dimension of either the row space or the column space of A. Note that for a matrix, the row space and the column space have the same dimension (called the rank of the matrix). Since there are fewer columns than rows, we consider the column space. The matrix A can be represented by l_c vertical l_r-vectors, i.e.,

$$A = (v_1, v_2, ..., v_{l_c})$$

where v_i is the i-th column vector. Since

$$v_1 + v_2 + ... + v_{l_c} = 0,$$

one of the column vectors can be expressed by the rest of the vectors, e.g , $v_1 = -v_2 - v_3 - ... - v_{l_c}$. Therefore, the rank of A, $rank(M) \leq l_c - 1$, where $l_c = n(n-1)/2$. □

FACT 2. *In the above set of equations, there are at least $n(n-1)/2 - 1$ linearly independent equations.*

PROOF. Given a solution of any variable, e.g, $c_{i,j}$, all the other variables can be solved as well: Note that $c_{i,j}$ leads to $c_{1,i}, \cdots, c_{n,i}$ from equations of $X_{k,i,j}$ where $k \neq i, k < j$ and $X_{j,i,k}$ where $k \neq i, k > j$. In this way we can compute all the remaining variables $c_{1,2}, \cdots, c_{n-1,n}$. Therefore, since given a solution to one variable $c_{i,j}$ all other $n(n-1)/2 - 1$ variables in the x vector can be recovered given $b = Mx$, the rank of M must be at least $n(n-1)/2 - 1$. □

Finally, sk is defined as $g^{c_{a_1,a_2} + c_{a_2,a_3} + \cdots + c_{a_m,a_1}}$; equivalently, we have

$$\log_g \mathsf{sk} = c_{a_1,a_2} + c_{a_2,a_3} + \cdots + c_{a_m,a_1}$$

This equation is independent from the set of equations determining all the values X. The reason is as in Fact 2 above: Given the value $\log_g(\mathsf{sk})$ and the vector b of X values allows one to recover at least one variable $c_{i,j}$ (e.g. the sum of the first $n-1$ rows and $\log_g(\mathsf{sk})$ reveals $c_{1,2}$, which, following the observation of Fact 2, leads to a recovery of all the other

variables c in vector x. Since all $n(n-1)/2$ variables c can be recovered from values X together with sk, matrix M together with the equation for sk must have rank $n(n-1)/2$. Since by Facts 1 and 2 matrix M by itself has rank $n(n-1)/2-1$, it follows that the equation for sk is linearly independent from the others. Consequently, key sk must be distributed independently from view T. This implies that for any adversary $\mathcal{A}$ we have:

$\Pr[(\mathsf{T},\mathsf{sk}_0) \leftarrow \mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2}; \mathsf{sk}_1 \leftarrow \mathbb{G}; b \leftarrow \{0,1\} : \mathcal{A}'(\mathsf{T},\mathsf{sk}_b) = b] = 1/2$.

In other words, no adversary can tell a real from random key in game $\mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2}$. Since by equation (2) we know that views Real and $\mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2}$ are distance $n(n-1)/2$ apart, a simple hybrid argument implies that for any algorithm $\mathcal{A}'$ running in time $t_{\mathsf{ddh}} - O(n^2)\cdot\epsilon_{\mathbf{ex}}$ we have:

$|\Pr[(\mathsf{T},\mathsf{sk}_0) \leftarrow \mathsf{Real}; \mathsf{sk}_1 \leftarrow \mathbb{G} : \mathcal{A}'(\mathsf{T},\mathsf{sk}_0) = 0] - \Pr[(\mathsf{T},\mathsf{sk}_0) \leftarrow \mathsf{Real}; \mathsf{sk}_1 \leftarrow \mathbb{G} : \mathcal{A}'(\mathsf{T},\mathsf{sk}_1) = 1]| \leq n(n-1)\cdot\epsilon_{\mathsf{ddh}}$.

6.2 Theorem 2: Security of the *RGKA* Protocol

Proof. The proof is similar with that of Theorem 1, except the square-DDH assumption is additionally required in the hybrid argument. Again we denote the set of live players in the final key computation step by $\{P_{a_1},...,P_{a_m}\}$ where $m \leq n$. To simplify notation we define $X_{a_i}{=}X_{a_{i-1},a_i,a_i}\cdot X_{a_i,a_i,a_{i+1}}$. The distribution of the transcript T and the resulting session key sk is given by:

$\mathsf{Real} =$

$$\left\{ \begin{array}{c} t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{1,1,2} = \frac{g^{t_2 t_1}}{g^{t_1^2}}, X_{1,1,3} = \frac{g^{t_3 t_1}}{g^{t_1^2}}, ..., X_{n,n,n-1} = \frac{g^{t_{n-1} t_n}}{g^{t_n^2}} \\ \mathsf{T} = (z_1,...,z_n, X_{1,1,2}, X_{1,1,3}, ..., X_{n,n,n-1}) \\ \mathsf{sk} = (g^{t_{a_1} t_{a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \\ \\ : (\mathsf{T},\mathsf{sk}) \end{array} \right\}.$$

Consider the following modified distribution Fake_1, which is like Real, except that all occurrences of $t_1 t_2$ are replaced by $c_{1,2}$.

$\mathsf{Fake}_1 =$

$$\left\{ \begin{array}{c} c_{1,2}, t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{1,1,2} = \frac{g^{c_{1,2}}}{g^{t_1^2}}, X_{1,1,3} = \frac{g^{t_3 t_1}}{g^{t_1^2}}, ..., X_{n,n,n-1} = \frac{g^{t_{n-1} t_n}}{g^{t_n^2}} \\ \mathsf{T} = (z_1,...,z_n, X_{1,1,2}, X_{1,1,3}, ..., X_{n,n,n-1}) \\ \mathsf{sk} = \left\{ \begin{array}{ll} (g^{c_{a_1,a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} & \text{if } (a_1,a_2) = (1,2); \\ (g^{t_{a_1} t_{a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} & \text{Otherwise.} \end{array} \right. \\ \\ : (\mathsf{T},\mathsf{sk}) \end{array} \right\}.$$

A standard argument shows that for any algorithm $\mathcal{A}'$ running in time $t_{\mathsf{ddh}} - O(n^2)\cdot\epsilon_{\mathbf{ex}}$ we have:

$|\Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Real} : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1] - \Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Fake}_1 : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1]| \leq \epsilon_{\mathsf{ddh}}$.

Proceeding in the similar way, since the number of all possible DH tuples is $n(n-1)/2$, we acquire the following distribution:

$\mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2} =$

$$\left\{ \begin{array}{c} c_{1,2},\cdots,c_{n-1,n}, t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{1,1,2} = \frac{g^{c_{1,2}}}{g^{t_1^2}}, X_{1,1,3} = \frac{g^{c_{1,3}}}{g^{t_1^2}}, ..., X_{n,n,n-1} = \frac{g^{c_{n-1,n}}}{g^{t_n^2}} \\ \mathsf{T} = (z_1,...,z_n, X_{1,1,2}, X_{1,1,3}, ..., X_{n,n,n-1}) \\ \mathsf{sk} = (g^{c_{a_1,a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \\ \\ : (\mathsf{T},\mathsf{sk}) \end{array} \right\}.$$

By the standard hybrid argument, for any $\mathcal{A}'$ running in time $t_{\mathsf{ddh}} - O(n^2)\cdot\epsilon_{\mathbf{ex}}$:

$|\Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Real} : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1] - \Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2} : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1]| \leq \frac{n(n-1)}{2}\cdot\epsilon_{\mathsf{ddh}}$.

We next consider the following different modification Fake'_1, which is like $\mathsf{Fake}_{\mathsf{n}(\mathsf{n}-1)/2}$, except that all occurrences of t_1^2 are replaced by $c_{1,1}$.

$\mathsf{Fake}'_1 =$

$$\left\{ \begin{array}{c} c_{1,1}, c_{1,2},\cdots,c_{n-1,n}, t_1,...,t_n \leftarrow \mathbb{Z}_q; z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{1,1,2} = \frac{g^{c_{1,2}}}{g^{c_{1,1}}}, X_{1,1,3} = \frac{g^{c_{1,3}}}{g^{c_{1,1}}}, ..., X_{n,n,n-1} = \frac{g^{c_{n-1,n}}}{g^{t_n^2}} \\ \mathsf{T} = (z_1,...,z_n, X_{1,1,2}, X_{1,1,3}, ..., X_{n,n,n-1}) \\ \mathsf{sk} = (g^{c_{a_1,a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \\ \\ : (\mathsf{T},\mathsf{sk}) \end{array} \right\}.$$

Proceeding in the similar way, we replace all t_i^2 by $c_{i,i}$, and the final distribution is:

$\mathsf{Fake}'_{\mathsf{n}} =$

$$\left\{ \begin{array}{c} c_{1,1},\cdots,c_{n,n}, c_{1,2},\cdots c_{n-1,n}, t_1,...,t_n \leftarrow \mathbb{Z}_q; \\ z_1 = g^{t_1},...,z_n = g^{t_n} \\ X_{1,1,2} = \frac{g^{c_{1,2}}}{g^{c_{1,1}}}, X_{1,1,3} = \frac{g^{c_{1,3}}}{g^{c_{1,1}}}, ..., X_{n,n,n-1} = \frac{g^{c_{n-1,n}}}{g^{c_{n,n}}} \\ \mathsf{T} = (z_1,...,z_n, X_{1,1,2}, X_{1,1,3}, ..., X_{n,n,n-1}) \\ \mathsf{sk} = (g^{c_{a_1,a_2}})^m \cdot (X_{a_2})^{m-1} \cdots X_{a_m} \\ \\ : (\mathsf{T},\mathsf{sk}) \end{array} \right\}.$$

By simple reduction argument, for all i:

$|\Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Fake}'_{\mathsf{i}} : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1] - \Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Fake}'_{\mathsf{i}+1} : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1]| \leq \epsilon_{\mathsf{sddh}}$.

Taking all the above, by the standard hybrid argument, for any $\mathcal{A}'$ running in time $t_{\mathsf{sddh}} - O(n^2)\cdot\epsilon_{\mathbf{ex}}$, we have:

$$|\Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Real} : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1] - \Pr[(\mathsf{T},\mathsf{sk}) \leftarrow \mathsf{Fake}'_{\mathsf{n}} : \mathcal{A}'(\mathsf{T},\mathsf{sk}) = 1]| \leq \frac{n(n-1)}{2}\cdot\epsilon_{\mathsf{ddh}} + n\cdot\epsilon_{\mathsf{sddh}} \leq n(n+1)/2\epsilon_{sddh} \quad (3)$$

(The last inequality follows because by trivial reduction of the DDH problem to the Square DDH problem we have $\epsilon_{ddh} \leq \epsilon_{sddh}$, for an adversary making two additional exponentiation operations.)

Now, in experiment $\mathsf{Fake}'_{\mathsf{n}}$, the values $c_{1,1}, \cdots, c_{n-1,n}$ are constrained by T according to the following $n(n-1)$ equations:

$$\begin{array}{c} \log_g X_{1,1,2} = c_{1,2} - c_{1,1} \\ \log_g X_{1,1,3} = c_{1,3} - c_{1,1} \\ \vdots \\ \log_g X_{1,n,n-1} = c_{n-1,n} - c_{n,n} \end{array}$$

Fact 3. *In the set of above equations, there are* $n(n+1)/2-1$ *linearly independent equations.*

We omit the proof of fact 3 because it is based on the same logic as in the proof of fact 1 and fact 2. Similarly, the fact that the equation for $\log_g(\mathsf{sk})$:

$$\log_g \mathsf{sk} = c_{a_1,a_2} + c_{a_2,a_3} + \cdots + c_{a_m,a_1},$$

is linearly independent from the above $n(n+1)/2$ equations implies that the value of sk is independent of the view T. This implies that, for any adversary $\mathcal{A}$:

$\Pr[(\mathsf{T}, \mathsf{sk}_0) \leftarrow \mathsf{Fake}'_{\mathsf{n}}; \mathsf{sk}_1 \leftarrow \mathbb{G}; b \leftarrow \{0,1\} : \mathcal{A}(\mathsf{T}, \mathsf{sk}_b) = b] = 1/2$.

As in the proof of theorem 1, it follows from the above equation and equation (3), that for any $\mathcal{A}'$ running in time $t_{\mathsf{sddh}} - O(n^2 \cdot \epsilon_{\mathbf{ex}})$, we have:

$|\Pr[(\mathsf{T}, \mathsf{sk}_0) \leftarrow \mathsf{Real}; \mathsf{sk}_1 \leftarrow \mathbb{G} : \mathcal{A}'(\mathsf{T}, \mathsf{sk}_0) = 0] - \Pr[(\mathsf{T}, \mathsf{sk}_0) \leftarrow \mathsf{Real}; \mathsf{sk}_1 \leftarrow \mathbb{G} : \mathcal{A}'(\mathsf{T}, \mathsf{sk}_1) = 1]| \leq n(n-1) \cdot \epsilon_{\mathsf{ddh}} + 2n \cdot \epsilon_{\mathsf{sddh}}$. □

7. CONCLUSION

In this paper, we proposed a novel 2-round Group Key Agreement protocol that tolerates up to T node failures using (reliable) broadcasts of $O(T)$-sized messages. To authors' knowledge, it is the first GKA protocol that offers a natural trade-off between message size and the desired level of fault-tolerance. In particular, we showed that the new protocol implies a fully-robust group key agreement with $O(\log n)$-sized messages and expected round complexity close to 2, assuming random faults. The new protocol is secure under the (standard) Decisional Square Diffie-Hellman assumption.

8. REFERENCES

[1] Y. Amir, Y. Kim, C. Nita-Rotaru, J. Schultz, J. Stanton, and G. Tsudik. Exploring robustness in group key agreement. In Proc. 21st IEEE International Conference on Distributed Computing Systems, pp. 399.409, 2001.

[2] D. Boneh. The decision Diffie-Hellman problem. In *Proc. of Third Algorithmic Number Theory Symposium*, LNCS vol. 1423, pages 48–63, 1998.

[3] E. Bresson, O. Chevassut, D. Pointcheval, and J. Quisquater. Provably Authenticated Group Diffie-Hellman Key Exchange In *Proceedings of the 8th ACM conference on Computer and communications security (CCS'01)*, 2001

[4] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In *Advances in Cryptology - EUROCRYPT 1994*, 1994.

[5] C. Cachin and R. Strobl. Asynchronous group key exchange with failures. In *Proc. 23rd ACM Symposium on Principles of Distributed Computing (PODC 2004)*, pages 357-366, July 2004.

[6] J. Katz and M. Yung. Scalable Protocols for Authenticated Group Key Exchange In *Advances in Cryptology - ASIACRYPT 2003*. 2003

[7] Y. Kim, A. Perrig, and G. Tsudik. Group Key Agreement Efficient in Communication. IEEE Transactions on Computers, 53(7):905.921, July 2004.

[8] D. Steer, L. Strawczynski, W. Diffie, and M. Wiener. A Secure Audio Teleconference System. Advances in Cryptology | Crypto '98, LNCS vol. 403, Springer-Verlag, 1990, pp. 520-528.

[9] M. Steiner, G. Tsudik, and M. Waidner. Key Agreement in Dynamic Peer Groups. IEEE Trans. on Parallel and Distributed Systems 11(8): 769-780 (2000).