GAnGS: Gather, Authenticate 'n Group Securely*

Chia-Hsin Owen Chen†, Chung-Wei Chen‡, Cynthia Kuo§, Yan-Hao Lai*, Jonathan M. McCune§, Ahren Studer§◇, Adrian Perrig§, Bo-Yin Yang†, Tzong-Chen Wu**

† Academia Sinica, Nankang, Taipei TW
‡ National Tsing Hua University, Hsinchu TW
§ CyLab/ECE/EPP at Carnegie Mellon University, Pittsburgh, PA USA
* National Chung Hsing University, Taichung TW
** National Taiwan University of Science and Technology, Taipei TW
◇Correspondence should be addressed to astuder@cmu.edu

ABSTRACT

Establishing secure communication among a group of physically collocated people is a challenge. This problem can be reduced to establishing authentic public keys among all the participants – these public keys then serve to establish a shared secret symmetric key for encryption and authentication of messages. Unfortunately, in most real-world settings, public key infrastructures (PKI) are uncommon and distributing a secret in a public space is difficult. Thus, it is a challenge to exchange authentic public keys in a scalable, secure, and easy to use fashion.

In this paper, we propose GAnGS, a protocol for the secure exchange of authenticated information among a group of people. In contrast to prior work, GAnGS resists Group-in-the-Middle and Sybil attacks by malicious insiders, as well as infiltration attacks by malicious bystanders. GAnGS is designed to be robust to user errors, such as miscounting the number of participants or incorrectly comparing checksums. We have implemented and evaluated GAnGS on Nokia N70 phones. The GAnGS system is viable and achieves a good balance between scalability, security, and ease of use.

Categories and Subject Descriptors C.2.0 [Computer – Communication Networks]: General – *security and protection*; H.1.2 [Models and Principles] User/Machine Systems – *human factors*

General Terms: Security, Human Factors

*This research was supported in part by the iCAST project, National Science Council, Taiwan under the Grant NSC96-3114-P-001-002-Y, CyLab at Carnegie Mellon under grants DAAD19-02-1-0389 and MURI W 911 NF 0710287 from the Army Research Office, grants CCF-0424422 and CNS-0627357 from the National Science Foundation, and General Motors through the GM-CMU Collaborative Research Laboratory. The views and conclusions contained here are those of the authors and should not be interpreted as representing the official policies or endorsements, either express or implied, of ARO, CMU, GM, iCast, NSF, or the U.S. Government or any of its agencies.

MobiCom'08, September 14–19, 2008, San Francisco, USA.

1. INTRODUCTION

Humans often form groups when meeting in person: potential collaborators exchange ideas and data after a conference, opposing parties communicate during contract negotiation, and political cabinets meet and later disseminate strategies during a campaign. In all of these scenarios, members of a group want to communicate without allowing others to access the information. Communication is secured using cryptographic keys, which enable the encryption and authentication of messages. The challenge is to establish these cryptographic keys in a secure, usable manner.

The goal of this work is to help establish keys within a group over a wireless medium. Wireless communication is invisible to humans and thus fails to indicate the source of information to the user. An attacker can thus easily perform a Man-in-the-Middle (MitM) attack and expose keys to outsiders or establish different keys within the group. Prior work relies on pre-existing associations or secrets within the group to secure wireless channels. In addition, prior work assumes that users can accurately count or compare strings of hexadecimal digits. Unfortunately, these assumptions often do not hold, and an alternative solution is necessary to establish group keys.

Prior work on group communication leverages a Public Key Infrastructure (PKI) to authenticate public keys or a group password to secure communication. However, a PKI may be impractical outside corporate settings, and a shared password needs to be securely exchanged between group members. When an ad-hoc group meets in a public space (e.g., an airport), an attacker can intercept all communication (electronic, spoken, or written), most likely observe the group's password and thus subvert the group. To protect wireless communication with no prior associations or shared secrets, *Demonstrative Identification (DI)* allows a user to indicate via physical actions which two devices should communicate (e.g., touching devices together, one device taking a picture of the other, etc.) [3, 21, 22, 24]. However, current DI schemes only work for pairwise communication. We extend DI to the group setting and introduce the notion of *Physical Articulation to Authenticate Legitimate Parties (PAALP)*. With PAALP, members of a group can verify the information they have wirelessly received originated from the expected physical devices. In essence, a protocol supporting PAALP (a "PAALPable" protocol) enables conversion of physical presence within a group into digital trust.

Even with a PKI or a group password, user errors can cause vulnerabilities when establishing keys. Prior works

assume humans can accurately count the number of members in the group and compare random strings (tasks which cannot be automated by a device). Other work has shown that users often commit errors in these simple tasks [15, 28]. When groups members fail to compare accurately, an attacker can split the group as part of a Group in the Middle (GitM) Attack [16]. When group members count incorrectly, an outsider can add himself to the group or an insider can add multiple identities (a Sybil attack [8]) without changing the checksum values used to verify successful completion. Given these errors, we would like to design a protocol that is secure in the presence of miscounts (*Enumeration Error Proof (EEP)*) and resilient to failed comparisons (*Comparison Error Proof (CEP)*).

We present GAnGS (Gather, Authenticate 'n Group Securely), a system that gathers, distributes, and verifies authentic information among a group of collocated mobile devices. GAnGS exchanges group members' public keys such that each group member obtains the authentic public key of every other member. Knowing every other members' public key can be used to encrypt messages or distribute group keys among any subset of the group. Our key insight is that verifying the group list via physical interaction in randomly assigned subgroups reduces user error while providing a probabilistic security guarantee.

GAnGS is PAALPable, EEP, and CEP. As such, a group using GAnGS can exclude outsiders and detect GitM and Sybil attacks. Achieving these usability goals presents a tradeoff for security. For 100% EEP and CEP, each device must interact with every other device (for a total of $O(n^2)$ interactions). GAnGS provides greater than 95% probability of attack detection while requiring only $O(n)$ interactions.

Contributions. With PAALP, we extend the idea of physically identifying parties which communicate wirelessly via physical actions in a group setting. We quantify the user effort in our PAALPable system by comparing the number of interactions between devices in a run of the protocol. These physical interactions provide resilience to user error and contribute the resilience to counting and comparison errors, EEP and CEP respectively. We present the first implementation of a system to bootstrap scalable, secure group communication and evaluate our approaches on Bluetooth-enabled Nokia N70 camera phones.

2. PROBLEM DEFINITION

The goal of GAnGS is to collect and distribute a list containing authentic copies of group member's public keys.

2.1 Design Requirements

We place the following requirements for scalability, security, and human error resistance on GAnGS:

- **Scalability.** GAnGS must distribute authentic information for groups of two to 25 individuals. (Group dynamics shift with larger groups, and the vast majority of productive groups have fewer than 25 members [12].) Group members must perform less interactions than if each device were to interact with every other device.
- **Security.** A group of ℓ physical members is formed successfully when each member possesses a group list Λ. Λ contains each user's authentic information, where user $\mathbb{X}$ contributes information $I_{\mathbb{X}}$. A successful authenticated exchange ensures the following properties:
  1. *Consistency.* All group members acquire the same information $I_{\mathbb{X}}$ from member $\mathbb{X}$. This includes "Inclusivity" in that when a member $\mathbb{X}$ has a copy of the list, the list must include $I_{\mathbb{X}}$.
  2. *Exclusivity.* Λ contains information from the ℓ intended group members, and no other individuals.
  3. *Uniqueness.* Each member $\mathbb{X}$ can only contribute one piece of information ($I_{\mathbb{X}}$) to Λ.
- **Resistance to Human Error.** As group size increases, the likelihood that a user will make a mistake also increases. Groups must be able to exchange authentic information accurately. In the event of user error(s), GAnGS should fail safely (discard the list).

2.2 Assumptions

We assume that:

- Group members are physically located in the same room during group formation;
- Group members can distinguish legitimate members from non-members;
- Members possess devices that support the same physical method for exchanging information (e.g., our implementation of GAnGS requires Bluetooth support, a camera, and programmable software); and
- Members accurately count less than ten individuals.

2.3 Attacker Model

Attackers may be located inside or outside of the room before, during, and after GAnGS is performed. If attackers are in the same room as group members, information displayed on a screen or written down may be seen by the attackers, and verbal communication may be heard by the attackers. With their devices, attackers can overhear, intercept, and inject any wireless communication. As part of an insider attack, a member of the group may be an attacker.

An attacker's goal is to add unintended entities' information to the list of public keys, edit or remove valid members' information from the list, and/or contribute multiple identities to the list (a Sybil attack [8]). An attacker can also perform a denial of service (DoS) attack such that GAnGS fails to complete successfully. The impact of a DoS attack is limited, since the attacker would fail to compromise the group's communication. GAnGS addresses attacks against a group's list, but not DoS.

3. BACKGROUND

This section reviews two existing authentication methods, Seeing-is-Believing (SiB) and Random Art, used by GAnGS.

3.1 SiB for Demonstrative Identification

Seeing-is-Believing (SiB) is an authentication scheme based on two-dimensional barcodes and camera-equipped mobile devices [21]. In SiB, one device displays a barcode encoding a piece of information, and a second device takes a picture of the barcode. The act of bringing the two devices together identifies precisely which two devices should communicate with one another (demonstrative identification [3]), providing robustness against man-in-the-middle attacks.

Consider Alice and Bob, both equipped with camera phones. Alice's phone can convey her public key $K_{\mathbb{A}}$ to Bob's phone

by encoding a commitment $h = hash(K_{\mathbb{A}})$ to her public key in a barcode, and displaying the barcode on-screen. Bob can then use his phone's camera to take a photograph of the barcode, obtaining h over the visual channel. We say that Alice is using her device to *show* her public key, and Bob is using his device to *find* Alice's public key.

Alice's device can send her full public key $K_{\mathbb{A}}$ to Bob's device via any untrusted medium, e.g., a wireless Bluetooth connection. When Bob's device receives Alice's public key $K'_{\mathbb{A}}$ via Bluetooth, it can verify the authenticity of the key using h: $h \stackrel{?}{=} hash(K'_{\mathbb{A}})$. To perform mutual authentication with SiB, Alice and Bob switch roles and repeat the protocol. This time Bob *shows* his public key and Alice *finds* it.

The SiB protocol requires $hash()$ to be a cryptographic hash function. To defend against a man-in-the-middle M, it must be infeasible for M to find a second pre-image K_x such that $hash(K_{\mathbb{A}}) = hash(K_x)$. Given a secure hash function, a successful attack on SiB requires an attacker to interfere with the process of photographing the barcode without being noticed. The security of the SiB protocol is based on the difficulty of stealthily interposing between the two devices while they are taking a picture of each other.

3.2 Random Art / Hash Visualization

Many authentication schemes require the parties involved to compare hashes, checksums, or other human non-understandable data [6, 29]. However, people are not good at rigorously performing such comparisons [28]. *Hash visualization* is a technique that creates structured images based on input data [23]. Similar input data outputs different images; comparing the images greatly reduces the human effort required to verify the equality of input data. GAnGS' hash visualization algorithm relies on *Random Art*, an algorithm that creates a structured image based on a pseudorandom bit sequence derived from the input through cryptographic means, with the goal of producing disparate images.

4. GAnGS PROTOCOLS

GAnGS provides users with a secure way to exchange authentic information among members of a group. It excludes information from non-group members and limits each group member to a single identity. GAnGS operates in three phases:

1. **Collection.** Information is collected from prospective group members to compose a pre-authenticated[1] list, Π. Π may include data from outsiders or Sybil entities.
2. **Distribution.** Π is distributed to all potential group members.
3. **Identification.** The group splits into randomly assigned subgroups to authenticate the information in Π. The protocol for dividing the group is described in Section 4.2. Once users gather into their subgroups, they use a Ring-based variant of GAnGS, GAnGS-R, to verify identities in Π. The physical interaction within the subgroup makes GAnGS PAALPable.

We describe two variants of GAnGS for the Collection and Distribution Phases in Section 4.1. GAnGS-P uses an untrusted Projector to assist in Collection, and GAnGS-T builds an ad hoc Tree without external infrastructure.

[1]Balfanz et al. originally used the term *pre-authenticated* to refer to authentic information exchanged via a location-limited channel (e.g., infrared or SiB), as opposed to the wireless channel [3]. Our use differs in that pre-authenticated information has been exchanged but has not yet been verified as authentic.

Used for all variants of GAnGS	
n	Number of identities in pre-authenticated list
a	Number of attacker identities in pre-authenticated list, $0 \leq a \leq n$
ℓ	Number of legitimate group members, $\ell + a = n$
$\mathbb{X}$	A potential group member
$K^+_{\mathbb{X}}$	$\mathbb{X}$'s public key
X	$\mathbb{X}$'s device
FN_X	Friendly (human-readable) name for X
BT_X	Bluetooth address of Device X
N_X	X's selected cryptographic nonce
$I_{\mathbb{X}}$	Information of potential group member $\mathbb{X}$
Π	Pre-authenticated list of group members' information, $\{I_1, \ldots, I_n\}$
Λ	Authenticated list of group members' information, $\{I_1, \ldots, I_\ell\}$
Used for GAnGS-T	
R	Root node
P	Parent node
C	Child node
Π_i	Pre-authenticated list of group members' information for one path through a GAnGS-T tree
Used for GAnGS-R	
s	Configured number of individuals that are assigned to a subgroup
π	Subgroup members' information based on Π
$\hat{\pi}$	Collected subgroup members' information

Table 1: Notation

The Identification Phase verifies that each subgroup list satisfies the security properties of consistency, exclusivity, and uniqueness (discussed in Section 2.1). If GAnGS-R detects an attack, the group must discard Π as inauthentic. An attacker can continue interrupting the protocol as a Denial of Service attack against GAnGS. Fortunately, GAnGS-R can help identify an attacker. Once the attacker is mitigated, the group can rerun the protocol with higher probability of success. When all subgroups succeed, Π becomes an authenticated list, Λ. It is important to note that the security of GAnGS-R assumes there is at least one honest, non-malicious person in each subgroup. There is a small probability that a subgroup of only malicious entities (e.g., multiple Sybil identities) may exist. Under such a scenario, the malicious entities can violate the necessary properties and compromise the security of the group's list. We analyze this class of attack in Section 5 and describe how to reduce the probability of a successful attack.

4.1 Collection & Distribution of Information

Here, we present the GAnGS-P and GAnGS-T protocols for collecting a list of pre-authenticated information.

4.1.1 GAnGS-P: Projector-based Collection and Distribution

GAnGS-P utilizes a projector[2] to help collect and distribute the list Π. The projector (or other display visible to all members) is untrusted but runs a GAnGS application.

[2]We actually use a computer whose output is displayed by a projector, but we say "the projector" for ease of exposition.

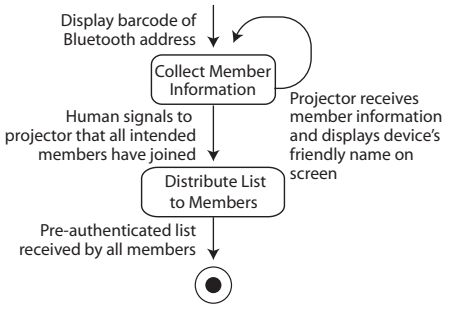


Figure 1: Flow Diagram for the Projector in GAnGS-P

The projector application simply Collects information from devices and Distributes the compiled list of information to devices. The projector allows all users to connect to one device with demonstrative identification despite the limitations of Bluetooth piconets (a maximum of 7 connections).

The flow diagrams for a projector and a device participating in GAnGS-P are shown in Figures 1 and 2(a), respectively.

Collection Phase. When the projector application first starts, it displays a barcode encoding its Bluetooth address. Prospective group members use their devices to photograph the barcode, thereby obtaining the necessary network information to connect to the projector.

Each device X connects to the projector and performs the following operations:

1. Generate a cryptographic nonce N_X.
2. Transmit $\mathbb{X}$'s information, $I_{\mathbb{X}}$, to the projector. $I_{\mathbb{X}}$ includes $\mathbb{X}$'s public key $K^+_{\mathbb{X}}$, her device's (X's) Bluetooth address BT_X, nonce N_X, and friendly (human-readable) name FN_X (e.g., "Alice's Device").

When the projector receives information $I_{\mathbb{X}}$ from a device X, it updates its display to include X's friendly name FN_X. This update, combined with a message displayed on X, demonstrates to the human user that the projector received her information.

After each member has sent her information to the projector, the projector enters the Distribution Phase. Currently, this transition is initiated by a human who is operating the projector application. (An alternative design is to allow any member of the group to initiate the transition with her mobile device. However, this gives an outsider – who may not be physically present in the room – the opportunity to prevent valid members from submitting their information.)

Distribution Phase. The projector application assembles the pre-authenticated list Π of all group members' information and transmits Π to each entity in the list. Upon receiving Π from the projector, each device verifies that its information is present in the list (i.e., device X verifies $I_{\mathbb{X}} \in \Pi$). If the information is not in the list, the device will raise an alert and stop GAnGS.

After GAnGS-P, every device has a list Π containing information about prospective group members. Π may be different across group members. Π may contain identities from a malicious projector, outsiders, or Sybil identities of malicious insiders. If a malicious party sends different Πs to different group members or inserts unwanted identities, more rigorous verification of the received information with GAnGS-R (see Section 4.3) will detect the attack.

4.1.2 GAnGS-T: Tree-based Collection and Distribution

There may be cases where a group would like to exchange information, without any supporting infrastructure. Ideally, a group member could act as a hub to pass information between all nodes in the group. However, Bluetooth piconets are limited to seven active slave connections, preventing all group members from connecting simultaneously. Thus, GAnGS-T builds an ad hoc tree structure out of group members to Collect and Distribute a group's pre-authenticated list Π. The flow diagrams for devices participating in GAnGS-T are shown in Figures 2(b) and 2(c).

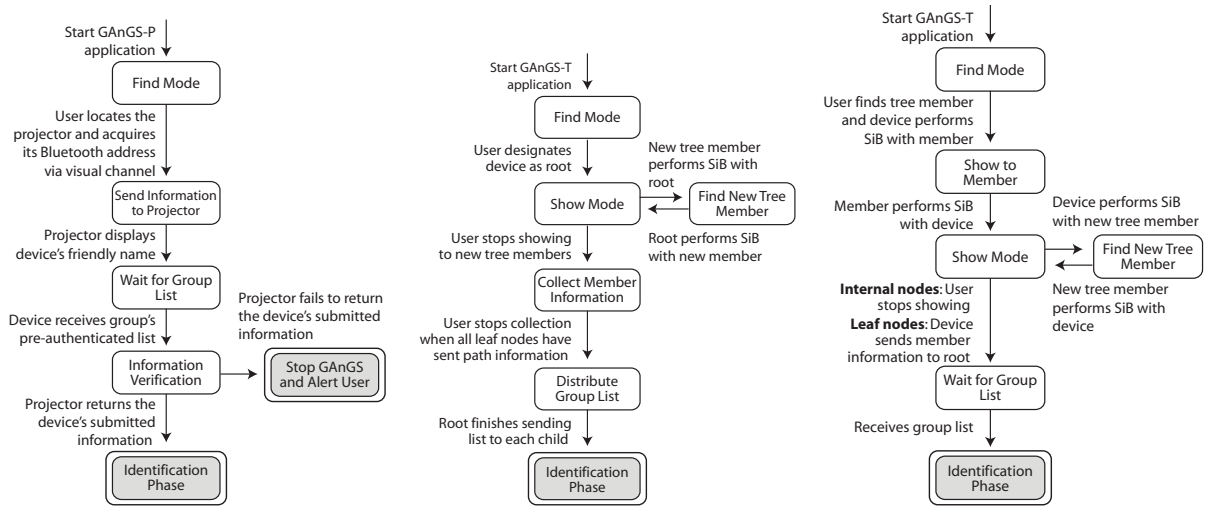
Collection Phase. To initiate GAnGS-T, the group selects one member to be the root of the tree. The root member indicates to her device that it is the root, R. R performs SiB with another member's device, which becomes the root's child in the tree. SiB ensures R and the child node C exchange information ($I_{\mathbb{R}}$ and $I_{\mathbb{C}}$) and not an attacker which adds themselves to the tree. (Specifically, R and C exchange public keys ($K^+_{\mathbb{R}}$ and $K^+_{\mathbb{C}}$), Bluetooth addresses (BT_R and BT_C), friendly names (FN_R and FN_C), and randomly generated nonces (N_R and N_C)). Once SiB is complete, R's device keeps track of the path in the tree ($I_{\mathbb{R}}||I_{\mathbb{C}}$), and signs the path using $\mathbb{R}$'s private key. After generating information about the path from the root to node C ($I_{(Path,\mathbb{C})}$) and R's signature for the path ($\sigma_{\mathbb{R}}(I_{(Path,\mathbb{C})})$), the root sends these two values to C. The signature ensures that only devices that are in the tree can add new members to the tree. Without signatures in GAnGS-T, outsiders could overhear a path on the tree and append themselves as legitimate members.

The remainder of the group members join the tree in a recursive fashion. New members join the tree and then add their own children to the tree. This joining process is detailed in Figure 3. Note that no device should appear in the tree more than once.

To ensure that tree members only perform SiB with new tree members, GAnGS-T requires that tree members and new tree members perform SiB in a set pattern. During the Collection Phase, devices switch between *find mode* or *show mode*. A device in *find mode* can only take pictures of devices that are showing a barcode (i.e., devices in *show mode*). This ensures that only a new tree member can initiate SiB with a tree member. Nodes that are already tree members cannot perform SiB with each other, since both devices are showing barcodes. New member nodes cannot perform SiB with each other, since both devices are trying to take pictures and their devices do not display barcodes.

Once every node has joined the tree, each leaf signs the path in which it appears as the last node ($\sigma_{Leaf}(I_{(Path,Leaf)})$), and sends the path and all of the relevant signatures (i.e., the root's signature, the root's child's signature, ..., and the leaf's signature) to the root as a pre-authenticated path list (Π_i) of the group. Once the root has received path and signature lists from all of the leaves, the root member presses a button on her device to initiate the Distribution Phase.

Distribution Phase. The root merges all of the path lists $\Pi_1, \Pi_2, ..., \Pi_n$ to form the larger group list Π and begins the distribution phase. Before sending Π to its children, the root verifies its information in each Π_i is unmodified and



(a) Flow Diagram for a Member Device in GAnGS-P

(b) Flow Diagram for the Root Device in GAnGS-T

(c) Flow Diagram for a Non-Root Device in GAnGS-T

Figure 2: Flow Diagrams

Parent (P) signs child node (C) into the Tree after exchanging information via SiB:
$\mathbb{X}$'s data $I_{\mathbb{X}}$ is her public key, her device's Bluetooth address, friendly name, and nonce $(K^+_{\mathbb{X}}, BT_X, FN_X, N_X)$.
$I_{(Path,\mathbb{X})}$ is the list of information for nodes on the path from the root to X $(I_{Root}||I_{Root's\_Child}||...||I_{X's\_Parent}||I_{\mathbb{X}})$.
$\sigma_{\mathbb{X}}(I_{(Path,\mathbb{Y})})$ is $\mathbb{X}$'s signature on the path including X's current child Y $(\text{Sign}_{K^{-1}_{\mathbb{X}}}(I_{(Path,\mathbb{Y})}))$.

1. $P: I_{(Path,\mathbb{C})} = I_{(Path,\mathbb{P})}\|\|I_{\mathbb{C}}$	P appends C's information to the path list.
2. $P: \sigma_{\mathbb{P}}(I_{(Path,\mathbb{C})}) = \text{Sign}_{K^{-1}_{\mathbb{P}}}(I_{(Path,\mathbb{C})})$	P signs the path list.
3. $P: \Sigma(Path) = \sigma_{\mathbb{R}}(I_{Path,Root's\_Child})\|\|...$ $\|\|\sigma_{\mathbb{P}'s\_Parent}(I_{Path,\mathbb{P}})\|\|\sigma_{\mathbb{P}}(I_{(Path,\mathbb{C})}))$	P appends its signature to the signature list $(\Sigma(Path))$ which it received from its parent.
4. $P \xrightarrow{BT} C: I_{(Path,\mathbb{C})}, \Sigma(Path)$	P sends C the path and signature lists.
5. C: if(!Verify$(\sigma_{\mathbb{P}}(I_{(Path,\mathbb{C})}), I_{(Path,\mathbb{C})}, K^+_{\mathbb{P}}))$ restart join	C verifies the signature and path are properly formed.

Figure 3: Parent Node P Adds a Child Node C to the Group Tree in GAnGS-T

that the signatures it received are valid. To verify Π_i, the root must check all of the signatures in each Π_i. Using the notation from Figure 3, the root verifies its own signature for the root and its child:
Verify$(\sigma_{\mathbb{R}}(Path, \mathbb{R}'s_Child), I_{\mathbb{R}}||I_{\mathbb{R}'s_Child}, K^+_{\mathbb{R}})$.
If this information is correct, the root has a self-authenticated copy of its child's public key which it can use to verify a copy of the grand-child's public key. The root continues to verify information in the path list until it reaches the leaf's signature, which verifies that the leaf was the last node in the path list. Once all of the signatures in all of the lists have been verified, the root knows the lists only contain group members (or that group members signed information for outsiders). The root transmits Π (the concatenation of the path lists and their signatures) to its children. Each child verifies the signatures and path lists using the same approach as the root and retransmits the information to its children.[3]

At the end of GAnGS-T, each device has a list Π that contains a potential collection of the group members' information. If all of the group members are honest, Π will match all of the properties from Section 2.1. Legitimate members will refrain from changing information in Π (consistency), only add legitimate members to the tree (exclusivity), and only contribute a single identity (uniqueness). If there are malicious insiders in the group, this list may contain outsiders, a group member may have multiple identities in the list, or the group's list could be inconsistent (i.e., an attacker could join the original group tree and act as the root for any of his children, removing the children from the larger group).

To detect these attacks, we use the GAnGS-R protocol as a mechanism to verify the information from GAnGS-T is correct. In Section 5, we discuss how GAnGS-R addresses the aforementioned attacks and find that with reasonable parameters over 95% of these attacks are detected.

4.2 Subgroup Formation

To verify that Π meets the security properties of consistency, exclusivity, and uniqueness, we assign the members listed in Π into subgroups. Each subgroup i possesses a sublist π_i, which it authenticates using GAnGS-R, which we describe in Section 4.3. Random subgroups and GAnGS-R provides a probabilistic security guarantee, parametrized by the number of devices in each subgroup, the number of honest group members, and the number of malicious identities

[3]Nodes that do not perform SiB with the root must rely on GAnGS-R to authenticate the root's information.

in Π. Section 5 presents our analysis.

A malicious node would like to elude detection by creating a subgroup that is composed solely of attacker-controlled identities. Thus, no device – be it a member device or the projector – should be able to influence subgroup assignments. Nonces alone do not prevent an attacker from controlling the subgroup assignments; the last member to join could eavesdrop on the other members' transmissions and select a nonce that results in a favorable subgroup assignment. To ensure random assignment, GAnGS-P & GAnGS-T incorporate a commitment scheme: each node commits to a randomly selected value and only reveals it after all nodes have committed to their values. (We excluded the commitment scheme from the prior subsections to help simplify presentation and focus on the unique aspects of the individual protocols.)

The commitment scheme has two steps: commit and reveal. During the commit step of the protocol, device X generates a random nonce N_X, but only includes the hash of the nonce ($h(N_X)$) in $\mathbb{X}$'s information ($I_{\mathbb{X}}$) as a commitment to N_X. This commit step works in conjunction with the initial collection and distribution of information in GAnGS-P or GAnGS-T. Once a node has received Π – including all other commitments – the node can reveal its original nonce N_X. A node can verify that another node did not change its nonce by checking that the hash commitment matches the value in the group information list. To change a nonce after providing a commitment, the node must find a collision in the hash function (which is computationally infeasible for a secure hash function). The reveal step requires an additional round of wireless communication after distribution of Π, but ensures that a malicious party cannot control subgroup assignment.

To direct subgroup formation, we use a pseudo-random number generator (PRNG) seeded using Π and the revealed nonces. Since every device contributes a nonce, a single well-behaved node that submits a truly random nonce suffices to prevent a malicious insider from creating predictable PRNG output. Figure 4 shows the algorithm we use to split Π into subgroups of size s, where subgroup i has pre-authenticated sublist π_i. Note that the number of identities in Π may not be evenly divisible by s, in which case we merge the undersized group with a group containing s members (Step 7 in Figure 4). After the algorithm has assigned subgroups, each user's device indicates the number of the subgroup they should join.

4.3 GAnGS-R: Ring-based Verification

After the group uses GAnGS-P or GAnGS-T to collect Π, it needs to verify that Π is a valid group list (Λ) that meets the security properties specified in Section 2.1. GAnGS-R provides a simple mechanism to achieve this goal within smaller subgroups. Provided that each subgroup meets the necessary properties, there is a high probability that the group as a whole will attain the same properties. To verify that the information from other members of the subgroup is correct and that each member is present only once, we can use Seeing-is-Believing (SiB) [21] or some other pairing method to securely acquire other subgroup member's data and detect when we pair with an individual more than once. However, pairing with every other member of a subgroup of size n requires $\frac{n(n-1)}{2}$ pairing operations. This approach is inefficient, may confuse users (i.e., "have I already paired with you?"), and take too long. Instead, we propose leveraging other group members to help collect and distribute the subgroup members' information.

GAnGS-R works in three steps. Users first count the number of members in their subgroup to detect outsiders or identities without corresponding physical bodies (i.e., Sybil identities). This is simple since a subgroup is s members (where s is 5 or fewer).[4] Next, each member in the subgroup performs unidirectional SiB to collect, sign, and pass on her neighbor's subgroup information. After $n-1$ SiB exchanges (Figure 5), the last member of the subgroup has a complete subgroup list and begins a distribution and verification step. During this step, signatures from other subgroup members prove the appropriate members are in the subgroup. If incorrect members are in the subgroup, the signatures will be absent or will not verify when using public keys from Π. After verifying signatures, a comparison of random art in the subgroup ensures that previously acquired group and subgroup information is consistent. This approach assumes that groups are small enough so that users can reliably count how many people are present and compare a final random art image with other members of the subgroup. Prior work has shown that subgroups of less than 10 individuals can count and compare correctly [15].

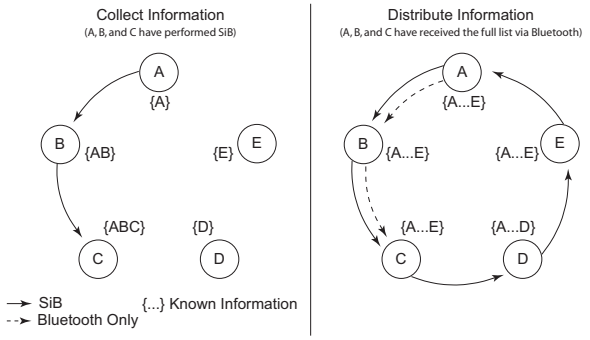


Figure 5: Intermediary phases of the two communication steps of GAnGS-R (subgroup size $s = 5$).

Counting. Within each subgroup, the users' devices know how many users should be present based on the subgroup's list π. Since the subgroup is small, users can count how many subgroup members are present. After the user enters the number of people present in the subgroup, the device verifies that this is the expected number. We discuss the attacks detected when the number of subgroup members is different than expected in the following security analysis. Small groups and verification of subgroup size (as opposed to a human counting a group the device does not know about) provide the Enumeration Error Proof (EEP) property to GAnGS. If a human miscounts, the protocol will halt and discard the list. Once users verify that the subgroup contains the appropriate number of physical group members, the collection step of the protocol begins.

Collection. During the collection step, the subgroup gathers authentic copies of information from the members of the subgroup into a new subgroup list $\hat{\pi}$. Once the collection step is complete, subgroup members obtain $\hat{\pi}$ from known sources, ensuring that entities from outside of the subgroup have not modified $\hat{\pi}$.

[4]When the group size is not divisible by s one subgroup may have as many as $2s-1$ members.

Randomly order the identities in Π into a list R:

1. $PRNG.seed(\Pi \| Nonces)$	Seed PRNG using the list of members' information and nonces.
2. $S = \Pi \quad R = \emptyset$	Initialize S to the list of all group members, R to an empty list.
3. $while(S \neq \emptyset)$	While there are unassigned members...
4. $\quad j = PRNG.rand() \mod \|S\|$	Select a random member j of S (treating S as 0-indexed array).
5. $\quad S = S \backslash I_j \quad R = R \| I_j$	Remove the information for member x from list S, and append the information to R.

Assign R's members into subgroups with information π_i:

6. $for(i = 0...\|R\| - 1) \quad Assign(I_i, \pi_{\lfloor \frac{i}{s} \rfloor})$	Generate sublists for each subgroup (treating R as 0-indexed array).
7. $if(\|R\| \mod s \neq 0) \quad Merge(\pi_{\lfloor \frac{\|R\|}{s} \rfloor - 1}, \pi_{\lfloor \frac{\|R\|}{s} \rfloor})$	Merge the undersized group with a full-sized group.

Figure 4: Splitting the group's pre-authenticated list (Π) into randomly assigned subgroups with sublist π_i of size s.

To start the collection step, a randomly assigned subgroup leader (A in Figure 5) performs a unidirectional SiB (uSiB) with the neighbor on his right[5] (B) to transfer a fresh copy of $\mathbb{A}$'s authenticated information (i.e., $I_{\mathbb{A}}$ with a new nonce N_A) and a signature for that data. In uSiB, only one device takes a picture of the other device, and information only flows securely in one direction (the direction of the arrows in Figure 5). After the uSiB transfer is done, B has an authenticated copy of A's information and A's signature for that information. Note that A has not yet learned B's information. Next, B appends her own information to the information received from A to form $\hat{\pi}$, signs this new list, and performs uSiB with B's neighbor C to transfer a copy of the ordered list and the collection of signatures (the barcode shows the hash of $\hat{\pi}$ so far, not just B's information). Group members continue to perform uSiB and append their information to $\hat{\pi}$ and sign the list until the last member of the group (E) has every group member's information and signature. When the last member of the group (E) performs uSiB with the leader (A), A receives an ordered list ($\hat{\pi}$), which includes A's own information and a set of signatures. The leader's device receiving its own information signals the end of the collection step and the beginning of the distribution step of the protocol.

Distribution and Verification. In the distribution and verification step of GAnGS-R, subgroup members' devices distribute and verify the information in $\hat{\pi}$ and forward a newly signed copy of $\hat{\pi}$ to the device with which they previously performed uSiB. To verify $\hat{\pi}$ contains the right information, every node checks that each entry in $\hat{\pi}$ contains the same public key and Bluetooth address as the entries in the expected list π.

If $\hat{\pi}$ is correct, the node verifies the different signatures from the subgroup. The signatures allow the receiving device to verify that it received $\hat{\pi}$ from the proper source (i.e., B signs $\hat{\pi}$ so C knows that a malicious outsider M did not inject a modified version of $\hat{\pi}$) and that the other group members in π are actually present during GAnGS-R (e.g., proves to C that nodes A, D and E are present). The original signature from the collection step is insufficient since a malicious party could replay messages from a prior run (i.e., an attacker can reuse A's initial signature from a prior run since A only signs A's own data). Once the node verifies the list and signatures, the node signs $\hat{\pi}$ and transmits the list $\hat{\pi}$, other nodes' signatures, and its own signature to the next device. Once every node has received $\hat{\pi}$ and verified the signatures, each device presents a random art image based on a combination of Π, $\hat{\pi}$, and the subgroup's signatures. At this time, group members should hold their phones together to allow a simple comparison of the images. This final check ensures the members of the subgroup have the same information. Holding the phones together for this comparison of a random art provides a Comparison Error Proof (CEP) property to GAnGS. Here at least one subgroup member has to accurately compare the few images when all phones are placed close together. This is an improvement over prior works where every member has to compare a hexadecimal string with every other member in the group. Even if the string is read aloud for comparison, a lazy or rushed member may ignore the comparison, exposing the group to an attack.

[5]We could use the left neighbor instead of the right neighbor. The important point is that every member of the subgroup performs only one find and one show.

5. ANALYSIS OF GAnGS

In this section, we first analyze GAnGS-R's effectiveness in authenticating members of small groups. (As group sizes increase, members may count incorrectly.) Next, we quantify and bound the probability that an attack on GAnGS goes undetected, assuming that the subgroup assignment is truly random. The section ends with an analysis of the efficiency of GAnGS with respect to the number of interactions between users.

5.1 Security Enforcement via GAnGS-R

Designed for the Identification Phase of GAnGS, GAnGS-R verifies the following properties:

- Exclusivity: Only physically present members are in the group;
- Uniqueness: Each member has a single identity; and
- Consistency: Every member has the same group lists, i.e., Π for the whole group, and π and $\hat{\pi}$ for subgroups.

Let us recap three key properties of GAnGS-R and how they defend against various attacks.

1. Continuous physical presence is required. Subgroup members must be physically proximate while taking pictures of each other's phones. Within such small groups, legitimate members will prevent outsiders from participating. In addition, they will notice if a single person attempts to participate in two subgroups simultaneously.

2. Each subgroup is small enough to count accu-

rately, and devices know how many members should be present. If the number of people gathered in a subgroup is different from the devices' expected number of individuals, GAnGS-R will fail. This indicates at least one outsider or Sybil identity was injected into the group list.

3. Subgroups are sequentially numbered, starting with one. If different people in the room somehow had different group and subgroup lists, there will be multiple subgroup 1's, multiple subgroup 2's, etc. Members of duplicated subgroups will find each other and combine groups. There are two possible outcomes. First, a combined group will have the wrong number of members. Alternatively, an attacker injects enough outsider or Sybil identities into both groups so that a combined subgroup contains the exact number of expected members. This inconsistency will be detected by signature verification and Random Art comparison.

Thus, if a subgroup contains at least one legitimate identity, the subgroup will detect an attack. If every subgroup contains at least one legitimate member, every sublist π (and, by extension, Π) fulfills all of the properties necessary to be a valid group list Λ. In the next section, we analyze the probability that at least one legitimate member is assigned to each subgroup.

5.2 Probability of Attack Detection

GAnGS provides a probabilistic guarantee that the group will detect an invalid list ($\Pi \neq \Lambda$). With randomly assigned subgroups, there is only a small chance that a subgroup will contain only malicious entities and an attack will go undetected during GAnGS-R. In this section, we analyze the probability of detecting outsiders who add their identities to the list or a malicious insider launching a Sybil attack.

When a list of ℓ legitimate identities (including potential malicious insiders) and a malicious identities is divided into subgroups of size s, each entry in Π is randomly assigned into one of g groups where $g = \lfloor \frac{\ell+a}{s} \rfloor$. Subgroup assignment reduces to a set partition problem.

When the number of potential group members is a multiple of subgroup size ($\ell + a \mod s = 0$), the number of potential subgroup assignments is

$$\frac{(\ell+a)!}{g!(s!)^g} \tag{1}$$

When one subgroup contains more members than other subgroups (i.e., $\ell + a \mod s \neq 0$ and one subgroup has $s + (\ell + a \mod s)$ members), the number of potential subgroup assignments is

$$\frac{(\ell+a)!}{(g-1)!(s!)^{g-1}(s+(\ell+a \mod s))!} \tag{2}$$

Malicious Outsiders. Outsiders who want to join the group need to form their own subgroup(s) to remain undetected.

When the number of legitimate members is not a multiple of s and malicious parties fail to contribute a multiple of s identities, at least one subgroup will contain both malicious and legitimate parties. Legitimate members will recognize the outsider as an invalid group member and detect this attack.

However, when the number of legitimate members is not a multiple of s and malicious parties contribute a multiple of s identities, there is a small probability of a successful attack. The number of potential subgroup assignments which elude detection is

$$\frac{\ell!}{(s+(\ell \mod s))!(\lfloor \frac{\ell}{s} \rfloor - 1)!(s!)^{\lfloor \frac{\ell}{s} \rfloor - 1}} \cdot \frac{a!}{(a/s)!(s!)^{a/s}} \tag{3}$$

Thus, malicious outsiders are detected with probability

$$1 - \frac{\text{Eq. (3)}}{\text{Eq. (2)}} = 1 - \frac{\ell! a!}{(\ell+a)!} \cdot \frac{(g-1)!}{(\lfloor \frac{\ell}{s} \rfloor - 1)! \frac{a}{s}!} \tag{4}$$

When the number of legitimate members and the number of outsiders are both multiples of s, the number of possible subgroup assignments where a outsiders and ℓ legitimate members do not exist in the same subgroups is

$$\frac{\ell!}{\frac{\ell}{s}!(s!)^{\ell/s}} \cdot \frac{a!}{\frac{a}{s}!(s!)^{a/s}} \tag{5}$$

Thus, malicious outsiders are detected with probability

$$1 - \frac{\text{Eq. (5)}}{\text{Eq. (1)}} = 1 - \frac{\ell! a! g!}{(\ell+a)! \frac{\ell}{s}! \frac{a}{s}!} \tag{6}$$

When the number of legitimate members is a multiple of s but the number of outsiders is not a multiple of s, the number of possible subgroup assignments which elude detection is the same as Eq. 4, but with ℓ and a reversed. Thus, the probability of attack detection is

$$1 - \frac{\ell! a!}{(\ell+a)!} \cdot \frac{(g-1)!}{\frac{\ell}{s}!(\lfloor \frac{a}{s} \rfloor - 1)!} \tag{7}$$

Comparing Equations 6 and 7, we can evaluate the optimal strategy for outsiders when the number of legitimate identities is fixed. It is best for outsiders to contribute a multiple of s identities, i.e., P(detect $a = 2s$) < P(detect $a = 2s - 1$). The reason for this non-intuitive result – that groups are more likely to detect fewer attackers – is that attackers must be in the same subgroup(s). When $a \mod s = 0$, the attackers can be part of any subgroup. However, when $a \mod s \neq 0$, the attackers must be in the one larger subgroup. For subgroups smaller than 4 members, attacks are always more likely to succeed with a multiple of s attackers. For example, with a subgroup size of 3, an attack is more likely to succeed with 6 outsiders than with 4 or 5 outsiders. With larger subgroups, it is better to have $s + 1$ than $2s$ attackers. The probability of 2 subgroups comprised only of attackers is smaller, but the probability of detecting $s + 2$ attackers is greater than the probability of detecting $2s$ attackers.

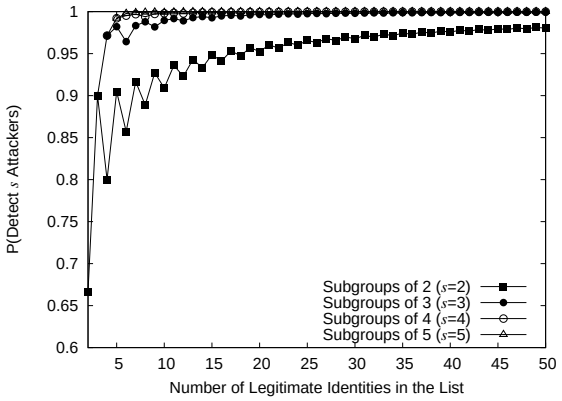


Figure 6: Probability of Detecting Outsiders

Figure 6 plots the probability of a group of ℓ members detecting s attackers (same number of attackers as members in each subgroup) for different subgroup sizes. *With a subgroup size of 4 or larger, the probability of detecting an attack is greater than 95% for all attacks.* As the size of the subgroup or the size of the list increases, the chance of detecting an attack also increases. As the subgroup size increases, more malicious entities must be assigned to the same subgroup for an attack to go undetected, which is less probable. As the total group size increases, there are more subgroups, increasing the chance that the malicious entities will be separated into different subgroups. The sawtooth pattern in Figure 6 is a result of the different scenarios where the number of identities in the list is or is not evenly divisible by the group size. When there is one subgroup with more members ($\ell + a \mod s \neq 0$), an attack can only succeed if the s attackers are placed in one of the $g - 1$ subgroups of size s. When this is the case, the probability of attack detection is larger than when each subgroup has exactly s members and the attackers could exist in any of the g subgroups.

Sybil Attacks. Legitimate members are less likely to detect Sybil attacks because an insider who generates the fake identities can pose as her true identity or any of the Sybil identities. For the group to detect the attack, the subgroup assignments must force the attacker to be in two places simultaneously. If an attacker must pose as a Sybil identity in one group and as her true identity or another Sybil identity in the same or another group, legitimate members will notice the attack.

An attack will go undetected if the attacker is in a subgroup with only Sybil identities and one Sybil identity is assigned to a subgroup with legitimate members. If the legitimate members fail to identify the Sybil identity or malicious insider by name, the legitimate members will believe the attacker represents the Sybil identity, and neither the legitimate members or the subgroup of Sybil identities fail to raise an alert.

The attacker's added flexibility of posing as any of the Sybil identities or herself increases the number of subgroup assignments that elude detection. When a malicious insider adds a Sybil identities, it is similar to the situation in Equations 5 or 3 with a outsiders. However, since the malicious insider can pose as any of the a Sybil identities, the number of missed configurations increases by a factor of $a + 1$. For example, the probability of detecting a Sybil attack with a Sybil identities and a multiple of s group members ($\ell \mod s = 0$) where the attacker is one of the ℓ members is

$$1 - (a+1) \cdot \frac{\text{Eq. (5)}}{\text{Eq. (1)}} = 1 - (a+1) \cdot \frac{\ell! a! g!}{(\ell + a)! \frac{\ell}{s}! \frac{a}{s}!} \qquad (8)$$

A Sybil attack is more likely to elude detection than an outsider attack. However, using a subgroup size of 4 or 5 still provides over 95% probability of detection when the group has 5 or more members.

5.3 Number of Interactions

The number of interactions between users in a PAALPable system should be limited. If the system requires a large number of interactions, users will consider the system cumbersome and resort to interacting with every other member or simply fail to adopt the system. If users must perform interactions in sequence, users will consider the system slow.

Figures 7 & 8 contain the total and sequential number of interactions required for GAnGS with subgroups of size 5 as compared to using pairwise SiB. We consider a bidirectional SiB as two interactions for these plots. GAnGS-T and GAnGS-P outperform pairwise SiB in both metrics with $O(n)$, $O(n)$, and $O(n^2)$ total interactions and $O(\log(n))$, $O(1)$, and $O(n)$ sequential interactions, respectively. For smaller groups (< 15), pairwise SiB has fewer sequential interactions because of the sequential nature of GAnGS-R.

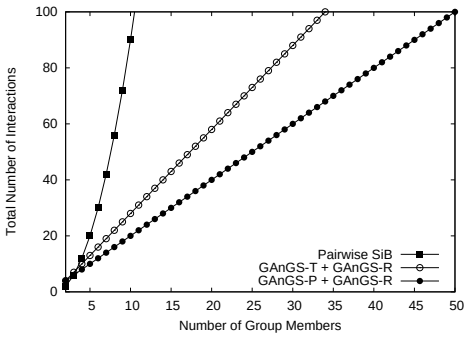


Figure 7: Total Interactions Performed to Collect Group Info

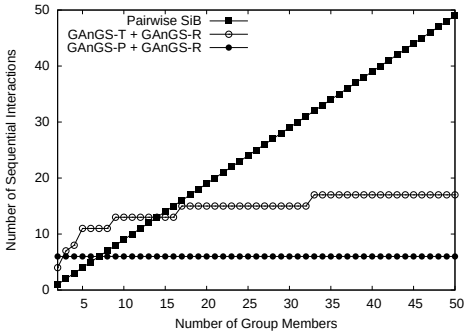


Figure 8: Sequential Interactions Performed to Collect Group Info

6. IMPLEMENTATION AND EVALUATION

We have fully implemented GAnGS-P and GAnGS-T for the Collection and Distribution Phases, and GAnGS-R for the Identification Phase. We describe our implementation, present macrobenchmark results from complete runs of our schemes, and include microbenchmark results to better understand the sources of overhead.

6.1 Implementation Details

Our implementation is a C++ program for Symbian OS v8.1a running on Nokia N70 smart phones, though it should be compatible with other camera- and Bluetooth-equipped smartphones running the same or a more recent version of Symbian OS. We use the same implementation of Seeing-is-Believing everywhere, and the same implementation of GAnGS-R after both GAnGS-P and GAnGS-T. The Symbian Installation System (SIS) binary for SiB with the necessary cryptographic libraries[6] is 51 KB. The SIS files for

[6] `http://xyssl.org/`

Operation	Avg (s)	Stdev (s)
Mutual SiB exchange	7.05	2.45
Unidirectional SiB	4.15	2.03
Mode Switch (*find*↔*show*)	1.19	0.35

Table 2: Results from 50 SiB exchanges

GAnGS-P + GAnGS-R and GAnGS-T + GAnGS-R are 64 KB and 79 KB, respectively.

The computer controlling the projector for GAnGS-P is a Windows XP laptop running our Java-based projector application. We use the BlueCove Java library for Bluetooth[7] to enable the same Bluetooth networking code that executes on the mobile phone to work with J2SE on Windows XP.

6.2 Macrobenchmarks

The most important performance characteristic of GAnGS is the user-perceived overhead. If GAnGS incurs too much overhead, it may evade adoption. We begin with a discussion of the cost of performing Seeing-is-Believing between two users, and then consider the scalability implications of SiB and how the GAnGS protocols offer a significant improvement.

Seeing-Is-Believing. We perform an experiment where we time a mutual SiB exchange between two people. Table 2 shows the results of our experiment. An interesting result from our experiments is that the user perceives the time for the *mode switch* (row 3) from *find* to *show* (or *show* to *find*) as being a more disruptive overhead than the time required for the unidirectional SiB (position the phones for barcode recognition and the subsequent connection and transmission of data (row 2)). Users are actively engaged in aiming the devices, whereas they are idly waiting during the mode switch. Section 6.3 includes additional details on the Bluetooth connection overhead.

The variations in timing is partly caused by equipment. *Older, heavily-scratched phones takes upwards of three times as long as the better-maintained ones to transmit via SiB.*

Overall GAnGS-T + GAnGS-R Performance. Figure 9 shows a breakdown of a GAnGS-T Collection Phase run with eight devices. Each half of the SiB sessions between each of the devices is illustrated. The solid gray areas in each device's bar indicate idle time from the user's perspective, even though the device may already be in *show* or *find* mode. The tree-structure formed by these devices is visible. The time required to complete each SiB exchange varies, with the first half of the exchange taking longer. This results from the need for users to move around to interact and varying user ability to photograph barcodes.

The Distribution Phase ran once the tree was constructed (i.e., all devices reached the end of Figure 9), requiring 62 seconds. Finally, the Identification Phase (Figure 5) begins with a run of GAnGS-R. We omit a detailed figure due to space constraints. Performing unidirectional SiB in each subgroup concurrently required 59 seconds. Distribution of the collected information required 18 seconds, and the final random art generation required 6 seconds.

A typical run of GAnGS-T + GAnGS-R with 15 (resp. 20) phones, tree-making took 120s (205s); distribution took 94s

[7] `http://code.google.com/p/bluecove/`

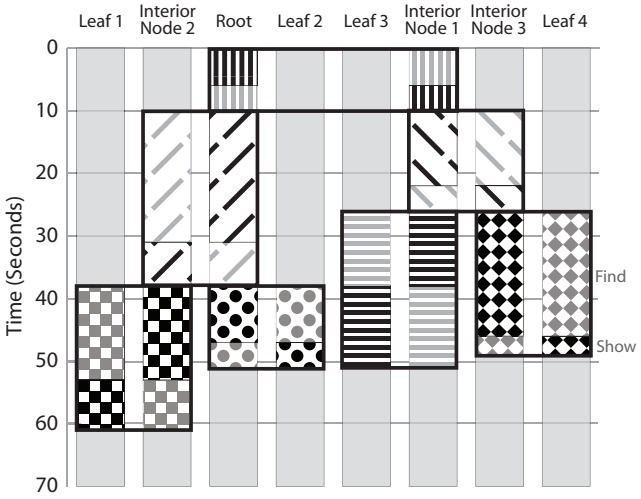


Figure 9: Duration of Collection Phase from one run of GAnGS-T with an eight-node tree. Black patterns indicate the devices are in *show* mode; gray patterns indicate *find* mode. Solidly shaded areas represent users' waiting time.

(117s); verification took 207s (145s) – note, verification time stage is now dominated by regrouping whose timing varying a lot: the last person to find the right group took 142s (96s) here. The 3-member GAnGS-R ran 37s (39s) seconds, the random art took 28 (8) seconds (also variable).

Overall GAnGS-P + GAnGS-R Performance. We performed a run of GAnGS-P + GAnGS-R with eight devices. The runtime can be broken down based on the three phases of GAnGS: sending data to the projector (Collection), receiving data from the projector (Distribution), and verifying the data with GAnGS-R (Identification). These phases took 128, 65, and 45 seconds, respectively. The three stagaes in a typical run with 15 people took 373, 220, and 52 (4-member GAnGS-R) seconds, respectively.

Our projector application is currently capable of accepting only one Bluetooth connection at a time. The performance of GAnGS-P suffers from this limitation because devices can only transmit their data to the projector one at a time. For example, each device averaged only 7 seconds to transmit its data to the projector during the Collection Phase. Likewise, the projector can only send the resulting group list Π to the group members one at a time during the Distribution Phase. There is nothing fundamental about this limitation, and we expect to improve our implementation of the projector-controlling software to operate at the Bluetooth limit of seven concurrent connections to a desktop-class device. We also plan to experiment with multiple Bluetooth adapters in the projector-controlling computer simultaneously, perhaps allowing concurrent communication with all group members.

6.3 Microbenchmarks

In this section, we discuss the overhead of Bluetooth communication and standard cryptographic operations.

Table 3 shows experimental results for transmitting 256 bytes over a Bluetooth connection between two Nokia N70s.

Table 4 shows the overhead experienced on the Nokia N70

Operation	Avg (s)	Stdev (s)
Connect	1.13	0.34
Send 256B	0.015	0.00

Table 3: Time to open a Bluetooth socket between two Nokia N70s, transmit 256 bytes, and receive an acknowledgment. Data from 50 trials.

Operation	Avg (s)	Stdev (s)
RSA KeyGen	10.74	8.35
RSA Sign	0.23	0.01
RSA Verify	0.02	0.01
Random Art	4.44	3.93

Table 4: Overhead of cryptographic operations used by our implementation on the Nokia N70. RSA operations were performed using 1024-bit keys. Data from 50 trials.

for each of several cryptographic operations that are commonly used in all of our protocols. The large standard deviations in the RSA key generation and Random Art image generation are inline with expectation, since the operations are non-deterministic based on random input.

7. RELATED WORK

This work is preceded by protocols that establish authentic information between two devices, which is often refereed to as "pairing". Proposed strategies include: password entry on one or both device(s) [19, 20]; string comparison that uses the human as a channel to ensure authentic exchange of information [17, 19, 20, 30]; audio-based comparison where the human user compares the strings via audio representation [10]; visual-based comparison of graphics that encode data [9,23]; shaking devices to create shared entropy pools [11, 18]; common properties of the wireless channel to establish authentic or secret information [6,7]; and location-limited channels [3, 22, 24].

Researchers have also proposed numerous key agreement protocols for groups, which rely on a PKI that issues certificates to each user [5, 13, 14, 25–27]. These protocols all assume a common trusted certification authority (CA); the CA is needed so that group members can authenticate other members' certificates. Unfortunately, this assumption is invalid in many settings. Different organizations may not have any trusted authorities in common, or group members may lack certificates entirely.

Within the PGP community, *key signing parties* may be held to authenticate groups of users [4]. The purpose of a key signing party is to extend the web of trust: users gather in a physical location to verify the identity of other attendees (e.g., using a passport or driver's license) and sign the PGP certificates linking attendees' names and public keys. The proposed methods are suitable for forming groups – but cumbersome. Attendees print their names and key fingerprints on slips of paper, to be verified manually by other attendees. Alternatively, a coordinator compiles a list of attendees in advance, and each attendee must be verified at the party. For large groups, comparing each attendee's key fingerprint is awkward and error-prone.

GAnGS is complementary to PKI-based schemes, as it can be used to establish the authenticated certificates needed to set up the group key.

The most closely related work to GAnGS is research on establishing keys for groups [1,2,29]. In contrast to GAnGS, all of these schemes rely on the end users to provide an accurate count and verify the members of the group. Also, prior work does not implement their schemes in a real-world system, which would raise numerous practical issues.

Finally, there is research using location-limited channels to exchange keys [3,24]. Talking to Strangers [3] uses demonstrative identification over a location-limited channel (e.g., infrared) to exchange authenticated public keys. Talking to Strangers may be used for groups, but it lacks a step for member verification. Thus, the scheme is vulnerable to malicious members who mount Sybil attacks; the multiple identities of one member would go undetected. The Resurrecting Duckling protocol [24] leverages a direct physical connection between devices for key setup. In the protocol, a mother duck (i.e., the group leader) defines and distributes a key to the ducklings (i.e., the other members of the group). During setup, a policy is uploaded. The policy specifies what actions a duckling will take. Thus, the mother duck's policy can direct the ducklings to support group communication. Unfortunately, this requires that the mother duck is completely trusted. In addition, there are several practical issues with using Resurrecting Duckling for groups. First, imprinting ducklings is a sequential operation. Every duckling needs to touch the mother duck, and she becomes a choke point in the group formation process. Second, the scheme requires a special interface that supports physical contact. Finally, like most other group schemes, Resurrecting Duckling has not been implemented in a real-world system to the best of our knowledge.

8. DISCUSSION

8.1 Group Management

Members of ad hoc groups often join or leave after group formation. A successful run of GAnGS distributes a set of authentic public keys; nodes can use the public keys to establish a group key [5, 13, 14, 25–27]. If members want to remove one member from the group, the group can exclude the undesired member from the group key establishment protocol. If members want to add a new member to the group, the new member can perform pairwise exchanges with other group members to securely exchange public keys. Once group members have the new member's key, the group can run a group key establishment protocol to generate a new key.

8.2 Small Groups

For small groups, pairwise exchanges require fewer user interactions than GAnGS to collect group information (see Section 5.3). When a group is small enough such that its members can count each other accurately, the group can use GAnGS-R directly, skipping GAnGS-P and GAnGS-T.

9. CONCLUSION & FUTURE WORK

GAnGS is the first fully implemented system to securely distribute authentic information in a group of imperfect operators that lack prior associations. In GAnGS, the physical interaction or PAALP enables group members to collect and distribute authentic information while achieving

resiliency to counting and comparison errors (Enumeration Error Proof (EEP) and Comparison Error Proof (CEP)).

Resilience to user errors presents a tradeoff between usability, efficiency, and security. With pairwise exchanges, a user can collect group information in $O(n)$ interactions with 100% attack detection and no counting or comparison. In GAnGS, we use randomly assigned subgroups to balance these goals. Subgroups with 5 members achieves a balance such that: a user has to perform at most $O(\log(n))$ operations, counting and comparison is less susceptible to errors, and probability of attack detection is 95% or greater.

To achieve tolerance to operator errors, we had to sacrifice some of GAnGS's scalability and ease-of-use. For future work, we plan to improve the execution time of GAnGS and to investigate more efficient designs that reduce communication between devices and interactions between users, without sacrificing security or usability. We also plan to investigate ways to recover from errors and attacks such that after detecting an error in a subgroup, the entire group does not need to rerun the protocol.

10. REFERENCES

[1] Abdalla, M., Bresson, E., Chevassut, O., and Pointcheval, D. Password-based group key exchange in a constant number of rounds. In *Public Key Cryptography (PKC)* (2006), pp. 427–442.

[2] Asokan, N., and Ginzboorg, P. Key-agreement in ad-hoc networks. *Computer Communications 23*, 17 (Nov. 2000), 1627–1637.

[3] Balfanz, D., Smetters, D., Stewart, P., and Wong, H. Talking to strangers: Authentication in adhoc wireless networks, 2002. In Symposium on Network and Distributed Systems Security (NDSS).

[4] Brennen, V. A. The keysigning party howto. `http://cryptnet.net/fdp/crypto/keysigning_party/en/keysigning_party.html`, 2008.

[5] Burmester, M., and Desmedt, Y. Efficient and secure conference key distribution. In *Security Protocols—International Workshop* (Apr. 1997), vol. 1189, pp. 119–129.

[6] Cagalj, M., Capkun, S., and Hubaux, J.-P. Key agreement in peer-to-peer wireless networks. *IEEE (Special Issue on Cryptography) 94* (2006), 467–478.

[7] Castelluccia, C., and Mutaf, P. Shake them up! a movement-based pairing protocol for cpu-constrained devices. In *Proceedings of ACM/Usenix Mobisys* (2005).

[8] Douceur, J. R. The Sybil attack. In *First International Workshop on Peer-to-Peer Systems (IPTPS)* (Mar. 2002).

[9] Ellison, C., and Dohrmann, S. Public-key support for group collaboration. *ACM Trans. Inf. Syst. Secur. 6*, 4 (2003), 547–565.

[10] Goodrich, M. T., Sirivianos, M., Solis, J., Tsudik, G., and Uzun, E. Loud and clear: Human-verifiable authentication based on audio. In *International Conference on Distributed Computing (ICDCS)* (2006), p. 10.

[11] Holmquist, L. E., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M., and Gellersen, H.-W. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *Proceedings of Ubicomp* (2001).

[12] Jay, A. How to run a meeting. *Harvard Business Review 54* (1976), 43–57.

[13] Just, M., and Vaudenay, S. Authenticated multi-party key agreement. In *Advances in Cryptology – (ASIACRYPT)* (1996), vol. 1163, pp. 36–49.

[14] Kim, Y., Perrig, A., and Tsudik, G. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)* (Nov. 2000), pp. 235–244.

[15] Kuo, C. *Reduction of End User Errors in the Design of Scalable, Secure Communication*. PhD thesis, Carnegie Mellon University, 2008.

[16] Kuo, C., Studer, A., and Perrig, A. Mind your manners: Socially appropriate wireless key establishment for groups. *Proceedings of First ACM Conference on Wireless Network Security (WiSec)* (Mar. 2008).

[17] Laur, S., and Nyberg, K. Efficient mutual data authentication using manually authenticated strings. In *Cryptology and Network Security (CANS)* (2006), pp. 90–107.

[18] Lester, J., Hannaford, B., and Gaetano, B. Are you with me? - using accelerometers to determine if two devices are carried by the same person. In *Proceedings of Pervasive* (2004).

[19] Linksky, J. et al. Simple Pairing Whitepaper, revision v10r00. `http://www.bluetooth.com/NR/rdonlyres/0A0B3F36-D15F-4470-85A6-F2CCFA26F70F/0/SimplePairing_WP_V10r00.pdf`, August 2006.

[20] Lortz, V., Roberts, D., Erdmann, B., Dawidowsky, F., Hayes, K., Yee, J. C., and Ishidoshiro, T. Wi-Fi Simple Config Specification, version 1.0a. Now known as Wi-Fi Protected Setup, February 2006.

[21] McCune, J. M., Perrig, A., and Reiter, M. K. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proceedings of the IEEE Symposium on Security and Privacy* (May 2005).

[22] NFC Forum. NFC Forum: Specifications. `http://www.nfc-forum.org/specs/`.

[23] Perrig, A., and Song, D. Hash visualization: A new technique to improve real-world security. In *Proceedings of the 1999 International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC)* (July 1999), pp. 131–138.

[24] Stajano, F., and Anderson, R. J. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Security Protocols Workshop* (1999), pp. 172–194.

[25] Steer, D., Strawczynski, L., Diffie, W., and Wiener, M. A secure audio teleconference system. In *Advances in Cryptology (Crypto)* (1990), vol. 403, pp. 520–528.

[26] Steiner, M., Tsudik, G., and Waidner, M. Key agreement in dynamic peer groups. *IEEE Transactions on Parallel and Distributed Systems 11*, 8 (Aug. 2000), 769–780.

[27] Tzeng, W., and Tzeng, Z. Round-efficient conference-key agreement protocols with provable security. In *Advances in Cryptology – (ASIACRYPT)* (2000), vol. 1976, pp. 614–628.

[28] Uzun, E., Karvonen, K., and Asokan, N. Usability analysis of secure pairing methods. In *Usable Security (USEC)* (Feb. 2007).

[29] Valkonen, J., Asokan, N., and Nyberg, K. Ad hoc security associations for groups. In *Security and Privacy in Ad-Hoc and Sensor Networks (ESAS)* (2006), pp. 150–164.

[30] Vaudenay, S. Secure communications over insecure channels based on short authenticated strings. In *Advances in Cryptology (Crypto)* (2005), pp. 309–326.