Ho-Po Key: Leveraging Physical Constraints on Human Motion to Authentically Exchange Information in a Group

Ghita Mezzour, Ahren Studer, Michael Farb, Jason Lee, Jonathan McCune, Hsu-Chun Hsiao, Adrian Perrig

December 8, 2010

CMU-CyLab-11-004

CyLab
Carnegie Mellon University
Pittsburgh, PA 15213

Ho-Po Key: Leveraging Physical Constraints on Human Motion to Authentically Exchange Information in a Group

ABSTRACT

Establishing a secure communication channel among a group of people is highly desirable. Such a secure channel can be boostrapped by physically meeting and authentically exchanging public keys. Recently, a new class of group key exchange protocols [8,21] that leverage physical constraints on human mobility was proposed. In this paper, we present Ho-Po Key, a new protocol for the authentic exchange of information within a physically collocated group of people. Ho-Po Key introduces a novel technique for the verification of the security properties of the information collected by group members. Group members physically form a ring. The position in the ring of each member is randomly assigned based on the information collected from all members. While standing in the ring, members compare short word lists with their neighbors. The verification technique in Ho-Po Key detects attacks by both outsider and insider adversaries. Outsiders are detected by group members if they physically stand in the ring with other members. Similarly, attacks by insiders are detected since an insider is unable to stand simultaneously in two positions in the ring. We demonstrated that the verification within the ring is surprisingly easy and fast via user-studies. We implemented Ho-Po Key on Motorola A855 Droid and Apple iPhone 3GS smartphones. The iPhone application is submitted to the iPhone application store and is waiting for approval, whereas the Android application is freely available on the Android market store.

1. INTRODUCTION

In a number of scenarios, a group of people who meet in person later want to communicate securely using mediums that are vulnerable to injection or eavesdropping of data. An example are researchers who meet at a conference and want to share research ideas or data without the fear of leaking information or violating the IRB process by revealing human subject data. However, this trust-after-meeting model can also apply to more than just "expert users". People who meet at a bar may want to send confidential text messages later. In these scenarios, the authentic exchange of public keys can help protect the secrecy of subsequent digital exchanges, but faces many challenges. One major challenge relates to the users running the protocol: They are novices, willing to spend little time and effort on the protocol. Time constraints rule out an approach where each member pairwise interacts with each other member, because this becomes cumbersome in large groups. Moreover, users are likely to make mistakes when asked to perform complicated tasks. For example, users are likely to miscount the members of a large group. Even worse, users may be tempted to skip some security-sensitive tasks. For example, when asked to indicate whether two short word lists are similar, users may indicate "yes" without performing the comparison.

Another significant challenge is related to the environment in which the protocol may be run. In an open environment like at a conference or a bar, adversaries may be present. Such adversaries can easily overhear the group's communication and observe all their actions. Such an adversary may be interested in injecting its information in the collected group information. For example, if the collected information consists of public keys used to configure an access control list, an adversary who successfully adds itself to the collected information has access to the group's resources. The adversary may also be interested in running a Group-in-the-Middle attack (GitM) [8], where members collect inconsistent lists. Another challenge is that one of the group members may be malicious. In addition to the capabilities of an outsider adversary, the malicious insider has the privilege of being able to participate in the protocol with the other members. The malicious insider may be interested in injecting Sybil identities [9] that help her outvote legitimate members in case the exchanged information is used for voting.

Previous work [5,16,17,29–31] on authentic key agreement or exchange requires each group member to have a certificate issued by a shared trusted CA. Such a requirement is too restrictive and severely limits the applicability of the protocol. Other previous work [1, 2, 32] assumes the group members to have a group password. Establishing a group password in an open space is not easy. An attacker may hear them talking and see what they write and subsequently perform a GitM attack. Moreover, these protocols are not secure against a malicious insider, as the password only provides secrecy, but not authenticity. GAnGS [8] and SPATE [21] constitute a new class of security protocols that leverage physical constraints on human motion to defend against attacks. In trying to scale to groups of up to 50 individuals, GAnGS only achieves probabilistic detection of outsider and

insider attacks, and poor usability. As members are likely to miscount the size of a large group [8], GAnGS uses a cumbersome divide-and-conquer approach where members are randomly assigned to small groups where they can accurately count. On the other hand, SPATE is a simple group exchange protocol, but only applies to small groups of fewer than 8 people. Moreover, in SPATE members may skip the visual hash comparison (we explain a visual hash in Section 3) and therefore not detect attacks [26]. Although GAnGS and SPATE were implemented, their implementation is not publicly available.

This paper presents Ho-Po Key[1], a protocol for the authenticated exchange of information among a group that leverages physical constraints on human motion. Ho-Po Key is simple and scales to large groups. Ho-Po Key provides perfect detection of outsider attacks and probabilistic detection of insider attacks. Ho-Po Key can be extended to provide perfect insider attack detection at the cost of requiring users to perform slightly more tasks. Ho-Po Key security properties are achieved thanks to a novel verification technique. The verification technique runs after members collect each other's information. As depicted in Figure 1, group members physically form a ring based on randomly assigned positions and compare word lists with their neighbor in the ring.

The successful physical ring formation ensures that the number of collected items equals the number of group members, providing an *implicit* count of group members. An outsider adversary that injects its information will also be assigned a position in the ring. As the outsider is unable to occupy its position undetected, the ring will not be formed and the attack is detected. Similarly, when a malicious insider injects Sybil identities, these Sybil identities will also be assigned positions in the ring. In case these positions are not adjacent, the insider will be unable to simultaneously occupy all these positions. As the ring will not form, the group members will detect the attack.

If the ring does form, each member's device displays three word lists. Each of these word lists consists of 3 words from the PGP word list [15]. One of these word lists encodes the first 3 bytes of the hash of the collected information, while the two other lists are randomly generated. The three lists are displayed in random order and two neighboring users need to indicate the matching list. Displaying two decoy lists and asking users to find the matching list is intended to force users to carefully compare the word lists instead of skipping the comparison by clicking "yes" [26]. Finally, the transitivity of the word list comparison in the ring ensures that all group members have the same collected information.

We perform user-studies to evaluate how easy it is for group members to physically form a ring based on randomly assigned positions. The ring formation is surprisingly fast (10–30 seconds) even for groups of up to 20 individuals. Members announce their positions loudly and form small ordered segments. These segments move together to form the complete ring. After forming the ring, members compare word lists with their two neighbors in the ring. As each member only performs two comparisons independent of the group size and different members can perform the comparisons simultaneously, this step is also very fast.

Our contributions in this paper are as follows:

- A novel group public key exchange that is easy, scalable, and provides perfect detection of outsider attacks and probabilistic detection of insider attacks. Achieving perfect detection of insider attacks is possible at the cost of slightly more user actions.
- An iPhone application of the Ho-Po Key exchange that is waiting for approval on the iPhone application store and an implementation for Android smart phones that is already publicly available on the Android market store.
- User studies which demonstrate that the tasks in Ho-Po Key can be easily performed by users.

[1] `http://en.wikipedia.org/wiki/HokeyCokey`

2. PROBLEM DEFINITION

We consider a group of people $\mathbb{X}_1, \cdots, \mathbb{X}_n$ who physically meet, where each member $\mathbb{X}_i$ has a piece of information $I_{\mathbb{X}_i}$. The goal of Ho-Po Key is to let each member $\mathbb{X}_i$ collect an information list $\Lambda_{\mathbb{X}_i}$ that contains exactly one piece of information $I_{\mathbb{X}_j}$ from each member $\mathbb{X}_j$. More formally, Ho-Po Key should satisfy the following properties [8]:

- **Security:**

  Each member $\mathbb{X}_i$ collects a list $\Lambda_{\mathbb{X}_i} = \{I_{\mathbb{X}_1}, \cdots, I_{\mathbb{X}_n}\}$ that meets the following requirements or detects an error.

  1. Consistency: All group members obtain the same data list $\Lambda_{\mathbb{X}_i}$.

  2. Exclusivity. The data list contains information from group members, but no other information.

  3. Uniqueness. The data list contains exactly *one* piece of information $I_{\mathbb{X}_j}$ from each member $\mathbb{X}_j$.

- **Usability.** The protocol should be easy to use to avoid burdening users. It should also complete quickly and require minimal user actions. The protocol should remain usable even in the case of a large group of up to 50 individuals.

- **Resilience to Human Error.** Users can make errors when running the protocol. For example, users may click on the wrong button. The chance that some user makes an error grows with the group size. In case of human error, the protocol should fail safely and users should discard the collected information. Resistance to human error should hold even for large groups.

2.1 Assumptions

Group members are collocated in the same physical space. Each group member can distinguish legitimate group members from outsiders. Each member has a mobile device that can access the Internet either through the cellular network or a WiFi access point.

2.2 Adversary Model

We distinguish between an *outsider adversary* and an *insider adversary*. The outsider adversary is not a legitimate group member, but controls the wireless channel by injecting, suppressing and modifying messages. The adversary may be physically present with the group members. It can overhear all their verbal and non-verbal communication. For example, it can see what members do and hear what they say. An insider adversary is a legitimate group member and

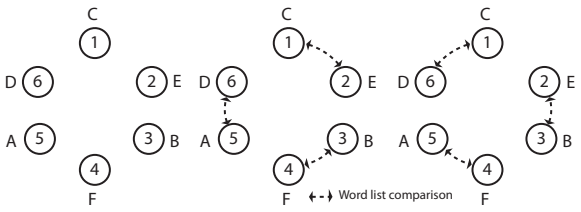


Figure 1: Verification Step. Members physically form a ring based on *randomly* assigned positions. Each member compares her word lists with each of her neighbors'.

has the same capabilities as the outsider adversary. The goal of both adversaries is to break any of the security properties. For example, an outsider adversary may be interested in violating exclusivity by injecting its information item into the list. An insider adversary may be interested in violating uniqueness by injecting Sybil identities that help her outvote legitimate members in case the exchanged information is later used for voting. The insider can freely move in the space and communicate with other members. However, when group members physically form a ring, we assume that the insider cannot stand in two non-adjacent places simultaneously. Finally, we assume that the adversary cannot break cryptographic primitives.

3. BACKGROUND

Visual Hashes. It is common practice to compare two pieces of data by comparing their hashes. The match of the hashes implies the match of the data pieces. However, while comparing a sequence of 20 bytes is easy for computers, this comparison is difficult and error-prone for humans. In order to alleviate this problem, the use of visual hashes was suggested in the literature [11] . A visual hash is a visual encoding of a hash value that can easily be compared by humans. Examples include random art [27], dictionary words [11], or T-Flags [21]. As humans should be able to easily compare visual hashes, these hashes typically contain an entropy on the order of a few bytes. Because of this limited entropy, finding collisions or second preimages is possible in seconds via simple brute force. This issue is typically alleviated in protocols by the use of commitments. Participants commit to values to be used as part of the input of the visual hash *before* knowing the values from the other participants. This limits the probability of causing a collision to 2^{-e}, for an entropy of e bits.

PGP word lists. In this paper, we use word lists as a visual hash. A word list consists of three words, each of which is chosen from the PGP word list [15]. The PGP word list consists of 256 short, easily pronounceable and distinguishable words. Each of these words represents one byte.

4. PROTOCOL OVERVIEW

Ho-Po Key is a secure protocol for exchanging authentic information among a group of people who physically meet. After running the protocol, the exchanged information satisfies the security properties of consistency, exclusivity and uniqueness. The protocol runs in two steps:

Collection. The goal of the collection step is for each device $M_{\mathbb{X}_i}$ to collect a pre-authenticated list of group members' information $\Pi_{\mathbb{X}_i}$. $\Pi_{\mathbb{X}_i}$ may contain information from outsiders or Sybil identities. Before the exchange of information I, devices exchange commitments over their information so that during the verification step group members only need to compare visual hashes instead of 20 byte hash values.

Verification. The goal of the verification step is to ensure that $\Pi_{\mathbb{X}_i}$ satisfies the security properties of consistency, exclusivity and uniqueness (recall Section 2). These properties hold if 1) the number of information items in $\Pi_{\mathbb{X}_i}$ equals the number of group members and 2) all members have matching word lists.

In order to verify the first property, the protocol assigns a random unique position between 1 and $|\Pi|$ to each group member. The random position assignment is based on data collected from all members. Group members need to physically form a ring based on these positions and verify that there is no missing or duplicate position in the ring. A missing or duplicate position indicates the presence of an outsider or Sybil identity.

In order to verify the second property, members verify that they have matching word lists with their two neighbors in the ring. Each member's device displays three word lists. One of these word lists encodes the first three bytes of the hash of the collected information, whereas the other two are randomly generated decoys. Neighboring users need to indicate the matching word list. The comparison with the other neighboring user is similar except that one word list encodes the fourth, fifth and sixth bytes of the hash of Π, instead of the first three bytes. The transitivity of the pairwise word list comparison ensures that all users have the same word list. The case where two neighboring users do not have any matching word list indicates that consistency does not hold and the protocol fails for all members. When all members find matching word lists with their neighbors, this ensures consistency by transitivity. Figure 1 depicts the verification step. Members $A, \ldots, F$ are assigned random unique positions between 1 and 6 and physically form a ring based on these positions. Subsequently, they perform the prescribed word list comparisons. For example, E compares word lists with C and B.

5. PROTOCOL DESCRIPTION

In this section, we describe our protocol in more detail. Section 5.1 presents a basic version of our protocol. This version is simple, fast and provides perfect detection of attacks by outsiders, but only provides probabilistic detection of attacks by malicious insiders. Section 5.2 presents an extension to provide perfect detection of attacks by insiders at the cost of slightly more actions from users. Table 1 describes our notation.

5.1 Basic Ho-Po Key

Ho-Po Key runs in two steps as described in Figure 3.

Collection. The goal of this step is for each mobile device $M_{\mathbb{X}_i}$ to collect $\Pi_{\mathbb{X}_i}$, a pre-authenticated list of members' information. The messages exchanged in the collection step are similar to prior work [21]. We review it here for com-

Table 1: Notation

$\mathbb{X}_i$	A group member
$I_{\mathbb{X}_i}$	Information of $\mathbb{X}_i$
$M_{\mathbb{X}_i}$	Mobile device of $\mathbb{X}_i$
$UI_{\mathbb{X}_i}$	User interface of $\mathbb{X}_i$
$\Pi_{\mathbb{X}_i}$	Pre-authenticated list of group members' information collected by $\mathbb{X}_i$
$\Lambda_{\mathbb{X}_i}$	Authenticated list of group members' information collected by $\mathbb{X}_i$
n	Number of group members
$P_{\mathbb{X}_i}$	Position of $\mathbb{X}_i$ in the ring, $1 \leq P_{\mathbb{X}_i} \leq \lvert\Pi_{\mathbb{X}_i}\rvert$
$\lvert S\rvert$	size of set S

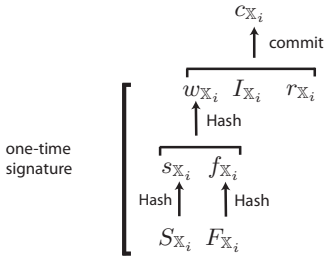


Figure 2: Generation of the one-time signature and the commitment by a mobile device $M_{\mathbb{X}_i}$.

pleteness. The information item $I_{\mathbb{X}_i}$ is set to a vCard [14] that contains a public key by default. At the end of this step, $\Pi_{\mathbb{X}_i}$ is not guaranteed to satisfy the security properties of consistency, exclusivity and uniqueness. Before exchanging the information items, group members exchange commitments in order to prevent an adversary from influencing the word list computation in a predictable manner as explained in Section 3.

Figure 2 depicts the generation of different values by a mobile device $M_{\mathbb{X}_i}$. $M_{\mathbb{X}_i}$ generates two random nonces: a success nonce $S_{\mathbb{X}_i}$ and a failure nonce $F_{\mathbb{X}_i}$. $M_{\mathbb{X}_i}$ applies a secure hash function (H) to each of the nonces to obtain $s_{\mathbb{X}_i} = H(S_{\mathbb{X}_i})$ and $f_{\mathbb{X}_i} = H(F_{\mathbb{X}_i})$, and subsequently computes $w_{\mathbb{X}_i} = H(s_{\mathbb{X}_i}||f_{\mathbb{X}_i})$. $M_{\mathbb{X}_i}$ generates a random number $r_{\mathbb{X}_i}$ and computes a commitment $c_{\mathbb{X}_i} \leftarrow commit(w_{\mathbb{X}_i}||I_{\mathbb{X}_i}||r_{\mathbb{X}_i})$. We now provide the intuition behind this construction. The use of commitments is related to the use of word lists as explained in Section 3. $r_{\mathbb{X}_i}$ is a source of randomness to be used for the random assignment of positions to group members. The construction $S_{\mathbb{X}_i}, F_{\mathbb{X}_i}, s_{\mathbb{X}_i}, f_{\mathbb{X}_i}, w_{\mathbb{X}_i}$ serves as a one-time signature with public value $w_{\mathbb{X}_i}$, and private values $S_{\mathbb{X}_i}, F_{\mathbb{X}_i}, s_{\mathbb{X}_i}, f_{\mathbb{X}_i}$. Towards the end of the protocol, $M_{\mathbb{X}_i}$ will release $(S_{\mathbb{X}_i}, f_{\mathbb{X}_i})$ if the user indicates that all protocol actions completed successfully. Alternatively, $M_{\mathbb{X}_i}$ releases $(s_{\mathbb{X}_i}, F_{\mathbb{X}_i})$ if at any point the user indicates the failure of the protocol.

When users start the exchange, their devices exchange the commitments c. After the collection of the commitments, devices exchange the decommitments which consist of the information to be exchanged, the list of public values of the one-time signatures and the random numbers. If $M_{\mathbb{X}_i}$ does not collect decommitments corresponding to all previously received commitments, the protocol times out and needs to be rerun.

Verification. The verification step of Ho-Po Key is novel. The goal of the verification step is to verify the security prop-

Collection step

0. $M_{\mathbb{X}_i}$: $F_{\mathbb{X}_i} \xleftarrow{r} \{0,1\}^\ell$, $f_{\mathbb{X}_i} \leftarrow H(F_{\mathbb{X}_i})$
: $S_{\mathbb{X}_i} \xleftarrow{r} \{0,1\}^\ell$, $s_{\mathbb{X}_i} \leftarrow H(S_{\mathbb{X}_i})$
: $w_{\mathbb{X}_i} \leftarrow H(s_{\mathbb{X}_i}||f_{\mathbb{X}_i})$, $r_{\mathbb{X}_i} \xleftarrow{r} \{0,1\}^\ell$
: $c_{\mathbb{X}_i} \leftarrow commit(w_{\mathbb{X}_i}||I_{\mathbb{X}_i}||r_{\mathbb{X}_i})$
: $\Pi_{\mathbb{X}_i} = \{I_{\mathbb{X}_i}\}$, $LC_{\mathbb{X}_i} = \{c_{\mathbb{X}_i}\}$, $LW_{\mathbb{X}_i} = \{w_{\mathbb{X}_i}\}$
1. $M_{\mathbb{X}_i} \rightarrow *$: $c_{\mathbb{X}_i}$
$M_{\mathbb{X}_i}$: $LC = LC \cup \{\text{collected commitments } c_{\mathbb{X}_j}\}$
: if $LC = \{c_{\mathbb{X}_i}\}$ quit
2. $M_{\mathbb{X}_i} \rightarrow *$: Open$(c_{\mathbb{X}_i})$
3. $M_{\mathbb{X}_i}$: $\Pi_{\mathbb{X}_i} = \Pi_{\mathbb{X}_i} \cup \{\text{collected } I_{\mathbb{X}_j}\}$
: $LW_{\mathbb{X}_i} = LW_{\mathbb{X}_i} \cup \{\text{collected } w_{\mathbb{X}_j}\}$
: $R_{\mathbb{X}_i} = R_{\mathbb{X}_i} \oplus r_{\mathbb{X}_j}$
: where $(w_{\mathbb{X}_j}||I_{\mathbb{X}_j}||r_{\mathbb{X}_j})$ is
: valid decommitment for $c_{\mathbb{X}_j}$
: If (timeout AND $\lvert\Pi_{\mathbb{X}_i}\rvert < \lvert LC\rvert$) quit

Verification step

Ring formation

4. $M_{\mathbb{X}_i}$: $P_{\mathbb{X}_i} \leftarrow AssignPos(R_{\mathbb{X}_i}, LW_{\mathbb{X}_i}, \Pi_{\mathbb{X}_i})$
: word list 1 $\leftarrow wordlist(H(\Pi_{\mathbb{X}_i}||LW_{\mathbb{X}_i}), 1)$
: word list 2 $\leftarrow wordlist(H(\Pi_{\mathbb{X}_i}||LW_{\mathbb{X}_i}), 2)$
5. $M_{\mathbb{X}_i} \xrightarrow{UI} \mathbb{X}_i$: $P_{\mathbb{X}_i}$,
: "Form a ring with other members by
: standing between members p, q"
: where $p = P_{\mathbb{X}_i} - 1 \, mod \, \lvert\Pi_{\mathbb{X}_i}\rvert$
: and $q = P_{\mathbb{X}_i} + 1 \, mod \, \lvert\Pi_{\mathbb{X}_i}\rvert$
:"Your left p", "Your right q"
: "Ring Formed", "Ring Cannot be Formed"
6. if ($\mathbb{X}_i \xrightarrow{UI} M_{\mathbb{X}_i}$: "Ring Cannot be Formed"){
$M_{\mathbb{X}} \rightarrow *$: $I_{\mathbb{X}_i}, w_{\mathbb{X}_i}, F_{\mathbb{X}_i}, s_{\mathbb{X}_i}$
$M_{\mathbb{X}}$: Quit }
7. else if ($\mathbb{X}_i \xrightarrow{UI} M_{\mathbb{X}_i}$: "Ring Formed") {

Word list comparison

$M_{\mathbb{X}_i} \xrightarrow{UI} \mathbb{X}_i$: "word list 1" // The 3 lists
: "decoy list 1" // are displayed
: "decoy list 2" // in random order
: "No word list matches"

If ($\mathbb{X}_i \xrightarrow{UI} M_{\mathbb{X}_i}$: "No word list matches" or "a decoy list")
: $z_{\mathbb{X}_i} \leftarrow S_{\mathbb{X}_i}, f_{\mathbb{X}_i}$
else if ($\mathbb{X}_i \xrightarrow{UI} M_{\mathbb{X}_i}$: "Word list 1") {
$M_{\mathbb{X}_i} \xrightarrow{UI} \mathbb{X}_i$: "word list 2" // The 3 lists
: "decoy list 3" // are displayed
: "decoy list 4" // in random order
: "No word list matches"
If ($\mathbb{X}_i \xrightarrow{UI} M_{\mathbb{X}_i}$: "No word list matches" or "a decoy list")
$M_{\mathbb{X}_i}$: $z_{\mathbb{X}_i} \leftarrow S_{\mathbb{X}_i}, f_{\mathbb{X}_i}$
else if ($\mathbb{X}_i \xrightarrow{UI} M_{\mathbb{X}_i}$: "word list 2")
$M_{\mathbb{X}_i}$: $z_{\mathbb{X}_i} \leftarrow s_{\mathbb{X}_i}, F_{\mathbb{X}_i}$
$M_{\mathbb{X}_i} \rightarrow *$: $z_{\mathbb{X}_i}$
$M_{\mathbb{X}_i}$: $LZ = LZ \cup$ collected $z_{\mathbb{X}_j}$
: where $w_{\mathbb{X}_j} = H(H(S_{\mathbb{X}_j})||f_{\mathbb{X}_j})$ or
: $w_{\mathbb{X}_j} = H(s_{\mathbb{X}_j}||H(H(F_{\mathbb{X}_j})))$
$M_{\mathbb{X}_i}$: if (LZ contains $z_{\mathbb{X}_j}$ such that
: $w_{\mathbb{X}_j} = H(s_{\mathbb{X}_j}||H(H(F_{\mathbb{X}_j})))||I_{\mathbb{X}_j}))$
: or $\lvert LZ\rvert < \lvert LC\rvert$)
: quit
$M_{\mathbb{X}}$: else $\Lambda_{\mathbb{X}_i} \leftarrow \Pi_{\mathbb{X}_i}$ }

Figure 3: Steps for user $\mathbb{X}_i$ to exchange information $I_{\mathbb{X}_i}$ with the other group members $\mathbb{X}_j$ via mobile devices. $\mathbb{X}_i \xrightarrow{UI} M_{\mathbb{X}_i}$ indicates inputs over the user interface from user $\mathbb{X}_i$ to their mobile device $M_{\mathbb{X}_j}$.

erties of the collected information list. If the verification is successful, the device saves the list as authentic. Otherwise, the device discards the list.

At the beginning of the verification step, the device computes a random position $P_{\mathbb{X}_i} \in \{1, ..., |\Pi_{\mathbb{X}_i}|\}$ as described in Algorithm 1. The computation is based on $R_{\mathbb{X}_i} = r_{\mathbb{X}_1} \oplus \ldots \oplus r_{\mathbb{X}_{|\Pi_{\mathbb{X}_i}|}}$: the xor of all the received random values, $LW_{\mathbb{X}_i} = \{w_{\mathbb{X}_1} \ldots w_{\mathbb{X}_{|\Pi_{\mathbb{X}_i}|}}\}$: the list of public values of the one-time signatures and $\Pi_{\mathbb{X}_i} = \{I_{\mathbb{X}_1} \ldots I_{\mathbb{X}_{|\Pi_{\mathbb{X}_i}|}}\}$: $\Pi_{\mathbb{X}_i}$ and $LW_{\mathbb{X}_i}$ are used after being sorted. A pseudo-random function seeded with R is used to generate a key k. A randomized message authentication code (MAC), e.g., CBC-MAC keyed with key k is applied to each of the public values of the one-time signatures. The computed MAC values are sorted and $P_{\mathbb{X}_i}$ is determined as the position of the MAC of $w_{\mathbb{X}_i}$ in the list of sorted MACs. Algorithm 1 is an easy technique to assign to $P_{\mathbb{X}_i}$ a random unique position in $1, \ldots, |\Pi_{\mathbb{X}_i}|$.

Algorithm 1 Random Position Assignment to member $\mathbb{X}_i$

Input: $R_{\mathbb{X}_i} = r_1 \oplus \cdots \oplus r_{|\Pi_{\mathbb{X}_i}|}$, $LW_{\mathbb{X}_i}$, $\Pi_{\mathbb{X}_i}$
Output: $P_{\mathbb{X}_i} \in \{1, \cdots, |\Pi_{\mathbb{X}_i}|\}$

$LW_{\mathbb{X}_i} \leftarrow sort(LW_{\mathbb{X}_i}))$, $\Pi_{\mathbb{X}_i} \leftarrow sort(\Pi_{\mathbb{X}_i})$
$k \leftarrow PRNG_R()$
for $j = 1$ to $|LW_{\mathbb{X}_i}|$ **do**
  $MC[j] \leftarrow MAC_k(LW_{\mathbb{X}_i}[j])$
end for
$MC_{sort} \leftarrow sort(MC)$
$MC_{\mathbb{X}_i} \leftarrow MAC_k(w_{\mathbb{X}_i})$
$P_{\mathbb{X}_i} \leftarrow$ position of $MC_{\mathbb{X}_i}$ in MC_{sort}

$M_{\mathbb{X}_i}$ prompts the user to physically form a ring with the other group members based on the assigned positions. Figure 4(a) depicts the screenshots of the mobile devices assigned positions 2 and 3. More specifically, $M_{\mathbb{X}_i}$ displays a message "Form a ring with other members by standing between members i and i", where $p = P_{\mathbb{X}_i} - 1 \, mod \, |\Pi_{\mathbb{X}_i}|$ and $q = P_{\mathbb{X}_i} + 1 \, mod \, |\Pi_{\mathbb{X}_i}|$. $M_{\mathbb{X}_i}$ also indicates that the member with position p should be on the left side of $\mathbb{X}_i$ and the member with position q should be on the right of $\mathbb{X}_i$. Finally, two buttons "Ring Formed" and "Ring Cannot be Formed" are displayed for the user to indicate whether the ring formation was successful or not. In case $\mathbb{X}$ indicates that the ring cannot be formed, the protocol fails and $M_{\mathbb{X}_i}$ releases the one-time signature private values $(s_{\mathbb{X}_i}, F_{\mathbb{X}_i})$ to indicate failure.

If $\mathbb{X}_i$ indicates "Ring Formed", the protocol continues with the word list comparisons. $M_{\mathbb{X}_i}$ displays a word list based on the first three bytes of $H(\Pi_{\mathbb{X}_i}, LW_{\mathbb{X}_i})$ in addition to two randomly generated decoy lists. The three lists are randomly ordered on the screen. $M_{\mathbb{X}_i}$ prompts $\mathbb{X}_i$ to indicate the matching word list with her neighbor assigned position $p = P_{\mathbb{X}_i} - 1 \, mod \, |\Pi_{\mathbb{X}_i}|$ when $P_{\mathbb{X}_i}$ is even. When $P_{\mathbb{X}_i}$ is odd, $M_{\mathbb{X}_i}$ prompts the user about the other user at position $q = P_{\mathbb{X}_i} + 1 \, mod \, |\Pi_{\mathbb{X}_i}|$. Figure 4 depicts the screenshots of members assigned positions 2 and 3 when performing the word list comparison. The matching word list is "inertia clamshell exodus", whereas the other word lists are decoys. Subsequently, $M_{\mathbb{X}_i}$ displays three new word lists, where one is computed based on the fourth, fifth and sixth bytes of $H(\Pi_{\mathbb{X}_i}, LW_{\mathbb{X}_i})$ and the two others are randomly generated decoys. The comparison with the other neighbor is similar to the first comparison.

The goal of indicating to users the order of neighbors with whom they should perform the comparison is intended to minimize the total number of non-parallelizable steps, thus minimizing the total protocol runtime. For example, for a group size of 6, all the pairwise word list comparisons $(1, 2)$, $(3, 4)$ and $(5, 6)$ can be performed concurrently. Subsequently, all the pairwise word list comparisons $(2, 3)$, $(4, 5)$ and $(6, 1)$ can be performed concurrently [2].

After the user clicks on the matching word lists during the two comparisons, $M_{\mathbb{X}_i}$ releases the one-time signature private values $(S_{\mathbb{X}_i}, f_{\mathbb{X}_i})$ to indicate success. Otherwise, $M_{\mathbb{X}_i}$ exchanges the private values $(s_{\mathbb{X}_i}, F_{\mathbb{X}_i})$ to indicate failure.

In the case where all group members' devices release the private values for success, the protocol is complete and the exchanged information list is saved as authentic. Otherwise, the protocol stops and the exchanged information is discarded.

5.2 Extensions

Basic Ho-Po Key ensures exclusivity, but only provides probabilistic guarantees about uniqueness and consistency. In case the malicious insider and her Sybil identities are assigned subsequent positions in the ring, the malicious insider can claim different positions to her two neighbors, thus evading detection. Similarly, two colluding insiders have a small probability of successfully launching a GitM attack. Detailed security analysis of Basic Ho-Po Key can be found in Section 6.2.3. In this section, we present an extension to Basic Ho-Po Key scheme that ensures perfect uniqueness and consistency. Providing perfect uniqueness and consistency comes at the cost of slightly more actions.

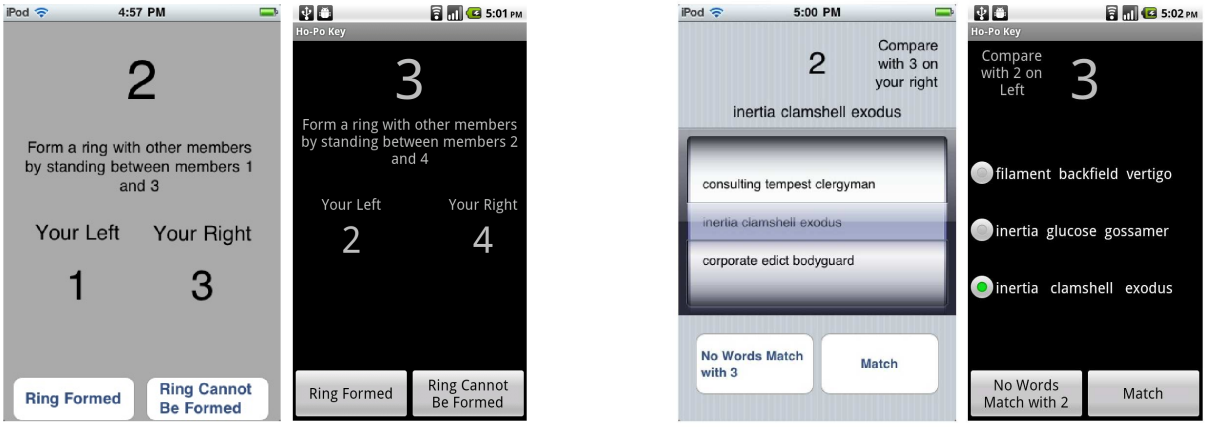
During the verification step, members' assigned positions are loudly announced. These announcements occur during the verification step, after members physically form a ring and before they perform the word list comparison. More specifically, after all members indicate "Ring Formed", their devices audibly announce their positions from 1 to $|\Pi|$. The sound should be loud enough to be heard by all members. Subsequently, $M_{\mathbb{X}_i}$ prompts $\mathbb{X}_i$ with a question: "Are group members standing in consecutive positions $1, \cdots, |\Pi_{\mathbb{X}_i}|$?". If $\mathbb{X}_i$ indicates "yes", then the protocol continues with the word list comparison similar to Basic Ho-Po Key. Otherwise, the protocol stops and the collected data is discarded.

Another extension that only applies to small groups is also applicable. At the start of the protocol, users need to indicate the number of group users as "1, 2, 3, 4, 5, 6, 7, 8, 8+". In case users enter a number ≤ 8, a version similar to prior work [21] runs. In case, users indicate "8+", the Ho-Po Key exchange runs.

5.3 Untrusted Server-Based Data Collection

The collection of devices' data is performed based on an untrusted server. To avoid collisions between multiple groups running the protocol simultaneously we use a Group ID number to identify members of a group to the server. Devices communicate with the untrusted server via HTTP. Using HTTP over the Internet for communication between devices is meant to allow scalability for multiple devices given the most openly available method common to the greatest number of smart phones. We observe that many smart

[2]When the group size is odd, the protocol requires 3 non parallelizable comparisons because both positions 1 and $|\Pi_{\mathbb{X}_i}|$ are odd



(a) Ring formation.

(b) Word list comparison.

Figure 4: Verification step. Screesnhots of the iPhone of member assigned position 2 and the Android of the member assigned position 3.

phones have at a minimum GSM cellular or WiFi Access Point communication modules. Bluetooth is another viable option. However, some phones (like the iPhone) are not interoperable with non-iPhone devices using Bluetooth, and do not scale well in a piconet topology. Ad-hoc WiFi, available on the Nokia E51 and Palm Pixie, could be viable as well, when more smartphones support tethering or a soft access point.

Each device posts its data to the server, and periodically queries the server for other devices' data. In order to allow the server to distinguish between data from different groups, a group identifier is used. We assign a pseudo-random unique user number to each member, preferably assigning small numbers. All members are asked to enter the lowest common user number among the group into their devices, promoting agreement between members on a common group identifier when starting Ho-Po Key.

6. SECURITY ANALYSIS

In this section, we analyze the security properties of Ho-Po Key. Section 6.1 considers an outsider adversary, while Section 6.2 considers a malicious insider. Unless explicitly stated, we analyze the Basic Ho-Po Key. Basic Ho-Po Key provides exclusivity, but only provides probabilistic guarantees about uniqueness and consistency. Uniqueness and consistency are enforced in the extension explained in Section 5.2. In this section we assume that honest users do not make errors. These errors are discussed in Section 8.2.

6.1 Outsider Adversary

In this section, we consider the case of an adversary that controls the wireless channel and can overhear all users' communications and observe all their actions.

6.1.1 *Consistency*

Consistency holds for the basic Ho-Po Key in the case of an outsider adversary. If all neighboring members have a matching word list, then all all members have a matching word list. The fact that all members have the same word list indicates that all members have the same information list.

6.1.2 *Exclusivity*

Exclusivity holds for Basic Ho-Po Key in the case of an outsider adversary. In the previous paragraph, we explained that if the verification step is successful, Λ is a superset of group members' information. In this paragraph, we explain that the number of information items in Λ equals the group size, which implies exclusivity.

The equality between the number of information items in Λ and the number of group members is ensured by the successful formation of the ring by group members. When forming the ring, a group member $\mathbb{X}_i$ stands between users assigned positions $P_{\mathbb{X}_i} - 1 \, mod \, |\Lambda_{\mathbb{X}_i}|$ and $P_{\mathbb{X}_i} + 1 \, mod \, |\Lambda_{\mathbb{X}_i}|$. All group members can find their two neighbors only if all positions $1, \cdots, |\Lambda_{\mathbb{X}_i}|$ are assigned to group members. This in turn is only possible if $|\Lambda_{\mathbb{X}_i}| = n$. In other words, if $|\Lambda_{\mathbb{X}_i}| > n$, there is at least one position P that is not assigned to any group member, and there is a hole in the ring.

6.1.3 *Uniqueness*

When all group members are honest, each group member injects exactly one piece of information. Therefore, uniqueness holds.

6.2 Insider Adversary

6.2.1 *Consistency*

Consistency holds for Basic Ho-Po Key in the case of a single malicious insider. If the word lists are similar for every word list comparison performed by a honest member, then all honest users have the same word list. This indicates that they all have the same data list Λ. Assume that two honest members with positions p, q , $p < q$ have different word lists. These two members are connected by two arcs of members: the ones with positions $p + 1, \cdots, q - 1$, and the ones with positions $q + 1, \cdots, p - 1 \, mod \, |\Lambda|$. The malicious insider is in only one of these arcs. This is because she can only be in

one physical location in the ring. Therefore one of the arcs only contains honest members. In this arc, two neighboring users have different word lists and detect the problem.

Colluding malicious insiders have a small chance of launching a successful GitM attack. We now compute the probability that two colluding insiders launch a GitM attack that is undetected. Without loss of generality, assume that the two colluding insiders are $\mathbb{X}_{n-1}$ and $\mathbb{X}_n$ and that members $sg_1 = \{\mathbb{X}_1, \ldots, \mathbb{X}_{n_1}\}$, where $1 \leq n_1 \leq n-2$, collect information list $\Pi_1 = \{I_{\mathbb{X}_1}, \ldots, I_{\mathbb{X}_{n_1}}, I'_{\mathbb{X}_{n_1+1}}, \cdots, I'_{\mathbb{X}_n}\}$, whereas the members $sg_2 = \mathbb{X}_{n_1+1}, \cdots, \mathbb{X}_{n-2}$ collect information list $\Pi_2 = \{I'_{\mathbb{X}_1}, \ldots, I'_{\mathbb{X}_{n_1}}, I_{\mathbb{X}_{n_1+1}}, \cdots, I_{\mathbb{X}_n}\}$, where $\Pi_1 \neq \Pi_2$. Members from sg_1 have no matching word lists. The GitM is undetected if the ring formation is successful if no member from sg_1 is assigned a subsequent position to a member from sg_2 in the ring. In other words, if all members from sg_1 are assigned subsequent positions, members in sg_2 are assigned subsequent positions and that $\mathbb{X}_{n-1}$ and $\mathbb{X}_n$ are assigned the two positions that separate sg_1 from sg_2. This condition needs hold in the random position assignment perceived by *both* sg_1 and sg_2. In the random position perceived by sg_1, the above position assignment occurs with probability $\frac{2n_1!(n-n_1-2)!}{n!}$. Therefore, the probability that the GitM attack is undetected is equal to $(\frac{2n_1!(n-n_1-2)!}{n!})^2$. By computing the derivative of the logarithm of this probability, we find that probability is maximized for $n_1 = 1$ and $n_1 = n-3$. In other words, the GitM attack has the highest probability of being undetected in case exactly one honest member has a different collected information from all the other honest group members. In this case, the probability of the GitM being undetected is $\frac{4}{(n-1)^2(n-2)^2}$.

6.2.2 Exclusivity

Exclusivity stands for the data list containing information from intended group members, but no other information. In other words, exclusivity is not violated with data introduced by insiders.

6.2.3 Uniqueness

Basic Ho-Po Key only provides probabilistic guarantees about uniqueness. In other words, there is a chance that Sybil identities injected by a malicious insider are undetected. Consider a malicious insider $\mathbb{M}$ that injects s Sybil identities during the collection step. For these Sybil identities to be undetected during the verification step, $\mathbb{M}$ needs to perform all the interactions that she and the Sybil identities are required to perform with the honest members. This is only possible if $\mathbb{M}$ and all the Sybil identities are assigned subsequent positions $i, i+1, \cdots, i+s \, mod \, |\Lambda|$ in the ring. In this case, $\mathbb{M}$ can interact with user $i-1 \, mod \, |\Lambda|$ as being $i \, mod \, |\Lambda|$ and user at $i+s+1 \, mod \, |\Lambda|$ as being $i+s \, mod \, |\Lambda|$. In case at least one of the Sybil identities is assigned a position not subsequent to the positions of $\mathbb{M}$ and the other Sybil identities, the ring cannot be formed as this would require $\mathbb{M}$ to stand at the two distinct physical locations at the same time, which is not possible. Therefore, the probability of detecting the Sybil attack is the probability that $\mathbb{M}$ and the Sybil identities are not assigned subsequent positions. As we show below, this probability is equal to $1 - \frac{(s+1)!}{(n+s-1)\cdots n}$ This probability is increasing in terms of s. The smallest detection probability is when $\mathbb{M}$ injects exactly one identity and is equal to $1 - \frac{2}{n}$. Let $\mathbb{S}_1, \cdots, \mathbb{S}_s$ be the s Sybil identities injected by $\mathbb{M}$. The probability that $\mathbb{M}$ and the s identities are assigned subsequent positions is $\sum_{i=1}^{n+s}(s+1)! \; P((P_{\mathbb{M}} \equiv i) \cap (P_{\mathbb{S}_1} \equiv i+1) \cdots (P_{\mathbb{S}_s} \equiv i+s))$ which is equal to $\sum_{i=1}^{n+s} \frac{(s+1)!}{(n+s)(n+s-1)\cdots n}$.

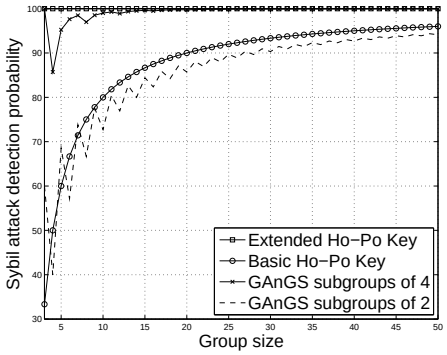


Figure 5: Sybil attack detection probability vs. The number of group members given that the number of injected Sybil identities is optimized to evade detection.

Loud Position Announcement These announcements prevent members from claiming different positions. More specifically, this extension forces members to claim exactly one position, which ensures $|\Lambda| = n$. Uniqueness follows because we have that for each $\mathbb{X}$, $I_{\mathbb{X}} \in \Lambda$ as explained in Sections 6.1 and 6.2.

Comparison between Sybil attack detection of Ho-Po Key and GAnGS. Figure 5 depicts the Sybil attack detection probability for different versions of Ho-Po Key and GAnGS vs. the group size. The figure considers the case when $\mathbb{M}$ injects a number of Sybil identities that is optimal to evade detection. For Basic Ho-Po Key, this optimal number is equal to 1, whereas for GAnGS this number is equal to the subgroup size in most cases.

In Extended Ho-Po Key, Sybil attacks are always detected. For Basic Ho-Po Key, the Sybil attack probability detection is small for small groups. For a group size of 3, it is less than 40%. This probability grows fast as the group size increases. It is equal to 80% for a group of 10 and 90% for a group of 20. The Sybil attack detection in GAnGS with subgroups of 2 follows a similar trend as in Ho-Po Key. For GAnGS with subgroups of 4, this probability is higher, but not equal to 1.

Figure 6 depicts the Sybil attack detection probability vs. the number of injected Sybil identities for both Ho-Po Key and GAnGS in the case of a group of 8 users. For Ho-Po Key, the detection probability is 80% for one Sybil identity. As the number of Sybil identities increases, the detection probability grows very fast. For 2 Sybil identities, this probability is already close to 95%. When more Sybil identities are injected, Ho-Po Key offers similar detection probability as the most secure versions of GAnGS with subgroups of 3 and 4. From the above, we see that for Ho-Po Key, Sybil attack detection is relatively small only when the number of Sybil identities is small. In general, the impact of Sybil attacks grows with the number of Sybil identities injected. For example, if the Sybil identities are intended to outvote legitimate members, then introducing one Sybil identity is

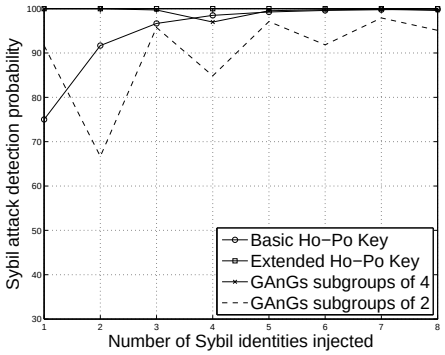


Figure 6: Sybil attack detection probability vs. The number of Sybil identities injected. Group size = 8.

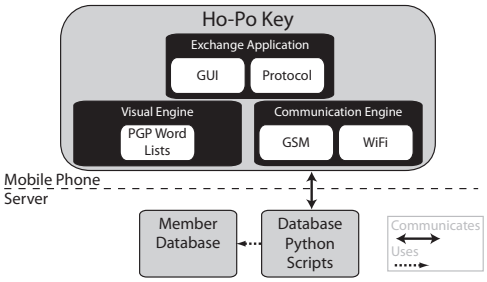


Figure 7: Ho-Po Key system overview.

a limited threat. When even one Sybil identity is an issue, then one can use the extension presented in Section 5.2.

We note that the computation of the Sybil attack detection probability in Ho-Po Key makes a pessimistic assumption that a Sybil attack is never detected in case $\mathbb{M}$ and the Sybil identities are assigned subsequent positions in the ring. In reality, there is a chance that this attack will be detected. The neighbors of $\mathbb{M}$ may hear her claiming two different positions to them. When this attack is detected, $\mathbb{M}$ is identified as being the attacker. The risk of identification represents a deterrent against launching this attack. Such deterrence is non-existent in GAnGS where a malicious insider is not required to perform any suspicious actions to cause the Sybil identities to be undetected. More specifically, in GAnGS, it is never possible to identify the malicious insider.

7. IMPLEMENTATION

We have implemented the Ho-Po Key system on Motorola A855 Droid and Apple iPhone 3GS smartphones as clients, and Google App Engine[3] as our server. The protocol contains client and server components, communicating over HTTP as a method of scaling the system to smartphones that have more restrictive communication protocols. The client supports secure key exchange using a server that can be untrusted. In the following sections, we describe the implementation details of this protocol.

The Ho-Po Key Mobile Client is implemented in Java for the Android 2.1 operating system and Objective-C for the iOS 4.2 operating system. The Android client application is 158 kb in size, while the iOS client is 228 kb. The communication between devices was built on an App Engine application to manage users and protocol sessions. The server application uses Python scripts to interact between client requests and the database. Clients use HTTP Post calls to communicate with the server and access the database.

Figure 7 shows the architecture of our Ho-Po Key Mobile Client and Server: it includes a library of commonly used client functions. The Ho-Po Key library includes communication and visual engines. The communication engine is responsible for data transmission and contains the web engine to make HTTP POST calls. The visual engine is used to generate word lists. Since Ho-Po Key is designed to scale past 8 devices, use of a Bluetooth *piconet* is not possible. We employ the web server to form a star network, with the server at the middle, acting as coordinator during a Ho-Po Key exchange.

The PGP Word Lists module is used at the end of authenticated data exchange; it displays the word lists on devices' screens Figure 4.

[3]http://code.google.com/appengine/

8. RESILIENCE TO HUMAN ERROR

Ho-Po Key makes an explicit effort to minimize user error. For example, the collection step runs completely transparently to users. Moreover, members are never required to count group members, and only need to compare word lists. However, members may still make errors. Negligent members may simply skip some actions. Ideally, in case of such behavior, Ho-Po Key should fail safely and the exchanged information should be discarded. In this section, we analyze the extent to which Ho-Po Key achieves this property. In our analysis, we consider that errors are due to neglect and not malicious behavior. We do not claim to exhaustively cover all possible user errors.

8.1 Errors During the Collection Step

The collection step runs transparently to group members. However, in case group members start the exchange at distant times, they may have different information lists at the end of the collection step. This should be detected during the verification step.

8.2 Errors During the Verification Step

Verification Step Skipping. Group members cannot skip the entire verification step. This is because during the verification step, members need to indicate the word list that matches one of the word lists on their neighbor's device. The protocol fails if one member fails to indicate the right word list.

No Physical Ring Formation. In case members do not physically form the ring and all of them indicate "Ring Formed", then the uniqueness and exclusivity properties may not hold. However, if at least one member indicate "Ring Cannot be Formed", then the protocol fails.

Careless Position Reading. Users may not pay careful attention to the assigned positions. For example, when forming the ring, a member may conclude that she is standing in the correct location after comparing her position to the member on her right, but not to the member on her left. Alternatively, members may completely discard the assigned positions and form a ring based on their positioning during the collection step. These errors may allow outsider or Sybil identities to be undetected. In Ho-Po Key, we make

Table 2: Comparative chart between Ho-Po Key, SPATE and GAnGS. n is the group size.

	Basic Ho-Po Key	Extended Ho-Po Key	SPATE	GAnGS-P + GAnGS-R	GAnGS-T + GAnGS-R
Exclusivity	Yes Yes	Yes Yes	Yes	**Probabilistic**	Yes
Uniqueness	Probabilistic	Yes	Yes	Probabilistic	Probabilistic
Total interactions	$O(n)$ position inquiries $O(n)$ word list comparisons	$O(n)$ position inquiries $O(n)$ word list comparisons	$O(n)$ (Count) $O(n)$ pictures $O(n^2)$ word list comparisons	$O(n)$ with projector $O(n)$ word list comparisons	$O(n)$ SiB $O(n)$ word list comparisons
Sequential interactions	$O(1)$ position inquiries $O(1)$ word list comparisons	$O(1)$ position inquiries $O(1)$ word list comparisons	$O(n)$ **(Count)** $O(n)$ **pictures** $O(1)$ word list comparisons	$O(n)$ **with projector** $O(1)$ pictures $O(1)$ word list comparisons	$O(\log_2 n)$ **SiB** $O(1)$ word list comparisons

an explicit effort to discourage these errors. For example, the positions are displayed in large font, and members are shown the positions of the members on their left and right. Moreover, a member $\mathbb{X}_i$ is asked to compare word lists with members with positions i and j, where $i = P_{\mathbb{X}_i} - 1 \, mod \, |\Pi_{\mathbb{X}_i}|$ and $j = P_{\mathbb{X}_i} + 1 \, mod \, |\Pi_{\mathbb{X}_i}|$, instead of simply asking them to compare word lists with members on their left and right. The loud position announcement extension minimizes the chances of this error since positions are loudly announced.

Careless Word List comparison. The use of the decoy word lists makes the protocol fail safe. Users need to carefully perform the visual hash comparison in order to make the protocol succeed. In case two users have no matching word lists, they are not provided with the option of clicking "yes" and skipping this comparison.

9. USABILITY

In order to evaluate the Ho-Po Key run time, we conducted user studies involving attendees at a research workshop. The studies mimic the situation when a group of researchers meet at a conference or workshop and need to authentically exchange information items. At the beginning of a workshop break, the user study was announced and attendees were invited to join. Attendees were asked to refrain from participating in multiple instances. User study participants consisted of students, faculty members and industry representatives.

We performed 4 user study instances involving groups of 9, 12, 19 and 22. Each study started by explaining the purpose of Ho-Po Key and providing each participant with a Motorola Droid A855 with the Ho-Po Key application installed. A user study instance consisted of two phases: a training phase and an evaluation phase. The training phase took 5-10 minutes. One of the organizers participated as a group member and guided users through the Ho-Po Key application. During the evaluation phase, the users were asked to run the Ho-Po Key for multiple rounds. The organizers provided the group identifier to be entered, but otherwise stayed aside from the group. During the four instances, the groups performed 4, 7, 5 and 3 rounds respectively. The number of rounds depended on the time remaining from the workshop break after recruiting users and training them. Information about users' actions was collected by examining the server's logs. Figure 8 depicts the protocol stages duration in seconds for each of the rounds performed during the evaluation phase. The rounds are reported in the order in which they were performed, with the x-axis representing the group size for that round.

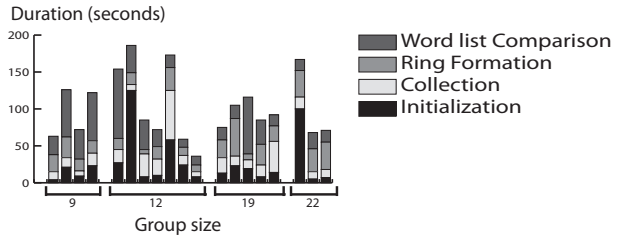


Figure 8: User studies results. Protocol Stages Duration.

As an overall observation, Ho-Po Key runs in 1-3 minutes. The runtime is not significantly affected by the group size, which is strong support for the scalability of Ho-Po Key. We now discuss the different protocol stages. Initilization represents the stage when members enter the group identifier. In most cases, members entered the group identifier within 10-20 seconds of each other. Collection represents the stage when devices exchange the actual information items. The exchange of these items only occurs after all members start the protocol and their devices exchange their commitments. Ring formation is the stage where members physically form the ring. Surprisingly, the ring formation took 10-20 seconds in most cases. The formation was highly parallelized. Members were announcing their positions and asking others for theirs. Small ordered segments were formed and these segments moved together to form the entire ring. Some users reported that forming the ring is playful and can be an icebreaker between a group of people. Word list comparison represents the stage where members compare their word lists with their neighbors' and indicate the comparison result. This stage is also highly parallelizable and scalable. Each member performs 2 word list comparisons independent of the group size.

The user studies indicate that users who are relatively familiar with the software can complete the protocol easily and fast. Members perform a small number of operations that are highly parallizable among group members. It should be noted however that the user studies do not necessarily evaluate the design of the user interface. This is due to the short training provided to members at the beginning of the study. Instead, the user studies demonstrate that the protocol actions are easy and fast to complete. As a future work, we would like to perform more user study instances and experimentally evaluate the robusteness of the protocol

to simulated attacks.

10. COMPARING Ho-Po Key with GAnGS and SPATE

Table 2 presents a comparison between Ho-Po Key, SPATE and GAnGS according to desired properties described in Section 2. The extended Ho-Po Key provides the exclusivity and uniqueness security properties. The operations in Ho-Po Key are easy and parallelizable. A member needs to inquire about other members' positions and position herself accordingly in the ring. In terms of complexity, the physical ring formation is equivalent to sorting a set of elements from 1 to n. This sorting takes constant time because once the position of element 1 is determined, the other elements' positions can be immediately determined. Subsequently, each member performs 2 word list comparisons independent of the group size.

In comparison, SPATE also provides the exclusivity and uniqueness security properties. In terms of operations performed, one of the members (the leader) needs to take a picture of the display of each of the other members. As these pictures cannot be parallelized, they take a time linear in the group size. Moreover, members need to count the number of group members. This count is error prone, especially for large groups. For these reasons, SPATE only applies to small groups of up to 8 individuals. Finally, SPATE was only evaluated by expert users. This evaluation provides useful insight concerning stages like the collection time. However, it provides very limited information concerning the performance of SPATE when run by other users.

In GAnGS, the collection step is based on either GAnGS-T or GAnGS-P, while the verification is based on GAnGS-R. When GAnGS-P is used for the collection, the protocol provides probabilistic exclusivity and uniqueness. When GAnGS-T is used instead, the protocol provides exclusivity but only probabilistic uniqueness. The number of sequential operations in GAnGS-P is linear in the group size as each member consecutively interacts with a projector. In GAnGS-T, the number of sequential operations is a logarithmic number of SiB [24] operations. A SiB operation takes approximately 15 seconds when run by users very familiar with the prototype [21]. Finally, the evaluation of GAnGS was not based on user studies.

11. PREVIOUS WORK

Researchers proposed several approaches for the authentic information exchange between two devices. Device pairing protocols include: comparison of a visual [10, 27], audio [12] or textual [18, 22, 23, 33] encoding of exchanged data, common entropy generation based on shaking devices [7, 13, 20], authentic information establishment based on properties of the wireless channel [6], and location limited channels [3, 25, 28].

The closest previous works are GAnGS [8] and SPATE [21]. GAnGS is also a protocol for the authentic information exchange within a group. GAnGS scales to large groups, at the cost of poor usability and probabilistic attack detection. SPATE is usable and provides better attack detection. However, SPATE only applies to groups of up to 8 individuals.

During PGP key signing parties [4], attendees sign PGP certificates that associate users' names and public keys. These parties can be adapted for the authentic information exchange within a group. For example, an organizer can create a list of attendees' names and public keys in advance to be verified at the party. However, this manual screening of the list is cumbersome and error-prone.

Key agreement protocols for groups aim at the establishment of a single group key. Many of these approaches [5, 16, 17, 29–31] require each user to have a certificate issued by a trusted Certification Authority (CA). These certificates are expensive and most people do not have them, which severely limits the applicability of these protocols. Other work [1, 2, 32] on key agreement relies on a shared password or string comparison. Securely agreeing on a group password in an open environment where an adversary may be present is challenging. Moreover Ho-Po Key solves a more general problem than key agreement protocols in the sense that any information can be exchanged. In the case where this information consists of public keys, those can be used to establish a group key. Group key protocols [19, 32] based on the comparison of a Short Authentication String were proposed in the litterature. However, those mainly focus on the cryptographic messages exchanged and pay little attention to the tasks performed by users.

Groupthink [26] studies usability aspects of existing group key protocols. More specifically, the paper considers protocols that require group members to verify Short Authentication Strings (SAS) and the group size. The paper compares multiple approaches for the verification of the SAS and group size, considering groups of size 4 and 6. More specifically, the paper compares methods that involve a centralized group member, to methods that are peer-based. The paper also compares methods that require members to input or copy the SAS and group size to methods that require members to verify these values. The paper comes to the conclusion that peer-based approaches where members input the group size and verify the SAS are best. While the paper provides interesting usability insight into group protocols, the techniques considered do not necessarily apply to large groups. For example, users cannot accurately count the number of people in a large group. Moreover, some approaches considered like requiring each member to compare the SAS with the member on the right and where members can choose their positions are vulnerable to attacks.

Other schemes [19, 32]

12. CONCLUSION

Ho-Po Key is a new protocol for authentic information exchange among a physically collocated group of people. Ho-Po Key achieves security properties by leveraging physical constraints on human mobility. More specifically, group members verify the properties of the collected information by physically forming a ring based on randomly assigned positions. The ring is a very simple and compact structure that group members can easily form. This verification technique provides perfect detection of outsider attackers and probabilistic detection of insider attackers. Perfect detection of insider attackers is achieved in Extended Ho-Po Key at the cost of slightly more actions by group members. Ho-Po Key is implemented and freely available on the Android market store. User-studies demonstrate that users can easily perform the tasks of Ho-Po Key. Forming the ring is surprisingly easy and fast. Some users reported that this activity is playful and can be an ice breaker between a group of people.

13. 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).

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

[3] Balfanz, D., Smetters, D., Stewart, P., and Wong, H. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS)* (2002).

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

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

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

[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] Chen, C.-H. O., Chen, C.-W., Kuo, C., Lai, Y.-H., McCune, J. M., Studer, A., Perrig, A., Yang, B.-Y., and Wu, T.-C. GAnGS: Gather Authenticate 'n Group Securely. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking* (Sept. 2008).

[9] Douceur, J. The sybil attack. *First International Workshop on Peer-to-Peer Systems (IPTPS)* (Mar. 2002).

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

[11] Ford, B., Strauss, J., Lesniewski-Laas, C., Rhea, S., Kaashoek, F., and Morris, R. Persistent personal names for globally connected mobile devices. In *Proceedings of USENIX Symposium on Operating Systems Design and Implementation (OSDI)* (Nov. 2006).

[12] 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).

[13] 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).

[14] Howes, T., and Smith, M. RFC 2425: A MIME content-type for directory information., Sept. 1998.

[15] Juola Patrick, P. Z. Whole-word phonetic distances and the pgpfone alphabet. In *Proccedings of the International Conference of Spoken Language Processing* (1996).

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

[17] 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).

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

[19] Laur, S., and Pasini, S. Sas-based group authentication and key agreement protocols. In *Public Key Cryptography* (2008).

[20] 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).

[21] Lin, Y.-H., Studer, A., Hsiao, H.-C., McCune, J. M., Wang, K.-H., Krohn, M., Lin, P.-L., Perrig, A., Sun, H.-M., and Yang, B.-Y. SPATE: Small-group PKI-less authenticated trust establishment. In *Proceedings of ACM/Usenix Mobisys* (2009).

[22] 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.

[23] 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.

[24] 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).

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

[26] Nithyanand, R., Saxena, N., Tsudik, G., and Uzun, E. Groupthink: Usability of secure group association for wireless devices. In *ACM Conference on Ubiquitous Computing (UbiComp)* (2010).

[27] Perrig, A., and Song, D. Hash visualization: A new technique to improve real-world security. In *International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC)* (1999).

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

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

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

[31] Tzeng, W.-G., and Tzeng, Z. Round-efficient conference-key agreement protocols with provable security. In *Advances in Cryptology – (ASIACRYPT)*

(2000), vol. 1976.

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

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