Behind multiple trapdoors: A cryptographic commitment scheme for establishing secure communications

Borja Gomez

kub0x@elhacker.net

October 15, 2019

Abstract

This paper introduces a cryptographic commitment using multiple platform groups where the attacker must solve multiple instances of the Discrete Logarithm Problem to break the scheme. The goal is that Alice and Bob establish a secret communication after verifying a finite set of values that have been computed using multiple trapdoor-permutation functions. Moreover, applicable cryptanalytic techniques and procedures that entirely define this scheme are discussed.

1 Introduction

Before commenting out the construction of the scheme and its security, presentation of the background is necessary. The algebraic scheme relies mainly on using group elements of cyclic groups where the Discrete Logarithm is believed to be hard. Later an implementation is discussed using ${Z_n}^*$: the multiplicative group of units modulo n. Besides, a commutative permutation group P is introduced to permute the transformations made by the parties on the initial shared sequence and public key material. Alice and Bob aim to commit on a sequence of values and as seen in other asymmetric schemes, every part involved in the commitment process has its own pair of private-public key. The commitment is represented as a n-tuple, and both parties must agree on the image of every point in their derived secret material supporting on hash functions to perform verification on those values. For the security part, the reader should be aware of decomposition techniques based on the Chinese Remainder Theorem (CRT), general group theory, probability, combinatorics and other methods that may require numerical knowledge.

1.1 A brief remark

The goal of this work is to show that Alice and Bob can commit to a sequence of values where the attacker has no clue of which values have been selected by both parties. Generally, parties setup their public keys by picking up an initial sequence, transforming and permuting it. Second, to compute secret key material they permute the other's party public key with their own private private permutation. As a consequence of P being commutative, permutations commute and parties should obtain the same ordering on the initial values, however, in the public keys, these have been already transformed by the other's party private key, so if transformations commute parties end up having the same shared key material. This is not easy even for parties, as the scheme is built in a way that they cannot know which transformation to apply once they permute other's public key since parties need to probe each other using hash functions to check on what value they do agree to reveal the position of the image of the commitment value. In the end parties have an unique sequence of points-images that can be used to build symmetric key material for enciphering information. In the security part, the attacker must solve a finite number of DLP instances to be able to compute the commitment values. There is also the chance for the attacker to build a system of equivalence chains that permits him to succeed against the group membership problem that is encountered later.

1.2 Motivation

The whole document is a consequence of the author studying how to build a cryptographic scheme that uses multiple instances of a concrete problem[1]. The idea is to create a commitment scheme using multiple trapdoor-permutations functions. Moreover a n-tuple of distinct trapdor-permutation functions $X = (f_1, \cdots, f_n)$ where Alice calculates the public key by permuting X with $\sigma \in P$ and evaluating every $f_{\sigma(i)}$ on the $\sigma(i)$-th element of her private key.

$Pub_A = (f_{\sigma(1)}(priv_{A,\sigma(1)}), \cdots, f_{\sigma(n)}(priv_{A,\sigma(n)})$

The attacker must determine for a random element which trapdoor-permutation function generated it, therefore, distinguishing is crucial to gain information of the permutation used by Alice. The other relevant point is how Alice and Bob arrive at the same shared key material for establishing a secure communication. As permuting each other's public key with their private permutation gives the same ordering on the elements of X, they must probe each other by transforming a commitment point with their private key elements until they find a coincidence, revealing the right image of the commitment point.

2 Scheme construction

The construction of the scheme is given algebraically first. The approach on this paper is to use a description based on groups and their elements, instead of using general trapdoor functions as seen before in section 1.2. The main reason of using groups is to give an implementation of the scheme using known platform groups found in Cryptography. In the end, both descriptions are equal since the operation that group elements use is a transformation, that's a trapdoor-permutation function.

2.1 Group based construction

Let X be the set of generators of n distinct cyclic groups G_i.

- $X = (g_1, \cdots, g_n) \in G_1 \times \cdots \times G_n$ where $< g_i >= G_i$ and $|X| = n$

Let P be a commutative group of permutations generated by r permutations $\sigma_1 \cdots \sigma_r$

- $P =< \sigma_1 \cdots \sigma_r >$

Let $Deg(P)$ be the degree of the permutation group P. The degree is the number of symbols of a permutation and it cannot exceed n as it could not permute beyond that limit.

- $Deg(P) \leq |X|$

Let δ be the set of commitment points involved on the commitment process between Alice and Bob

- $\delta = (\delta_1, \cdots, \delta_k) \in [1, Deg(P)]^k$

Let $Priv_A$ be the private key of Alice where the i-th element belongs to G_i.

- $Priv_A = (\alpha_1, \cdots, \alpha_n) \in G_1 \times \cdots \times G_n$

Let $Priv_B$ be the private key of Bob where the i-th element belongs to G_i.

- $Priv_B = (\beta_1, \cdots, \beta_n) \in G_1 \times \cdots \times G_n$

Let Pub_A be the public key of Alice consisting on n residues where she swapped the i-th position with the j-th, this is $\sigma(i) = j$. Same for Bob but he uses π to permute the instances.

- $Pub_A = (g_{\sigma(1)}^{\alpha_{\sigma(1)}}, \cdots, g_{\sigma(n)}^{\alpha_{\sigma(n)}}) \in G_{\sigma(1)} \times \cdots \times G_{\sigma(n)}$
- $Pub_B = (g_{\pi(1)}^{\beta_{\pi(1)}}, \cdots, g_{\pi(n)}^{\beta_{\pi(n)}}\} \in G_{\pi(1)} \times \cdots \times G_{\pi(n)}$

Let $Shared_{A,B}$ be the shared key that Alice obtains when permuting Pub_B with σ. If $\sigma\pi = \pi\sigma$ the generators are placed in the same order in $Shared_{A,B}$ and $Shared_{B,A}$, then Alice and Bob need only one step more to obtain the same transformation.

- $Shared_{A,B} = (g_{\sigma(\pi(1))}^{\beta_{\sigma(\pi(1))}}, \cdots, g_{\sigma(\pi(n))}^{\beta_{\sigma(\pi(n))}}) \in G_{\sigma\pi(1)} \times \cdots \times G_{\sigma\pi(n)}$
- $Shared_{B,A} = (g_{\pi(\sigma(1))}^{\alpha_{\pi(\sigma(1))}}, \cdots, g_{\pi(\sigma(n))}^{\alpha_{\pi(\sigma(n))}}) \in G_{\pi\sigma(1)} \times \cdots \times G_{\pi\sigma(n)}$

Let $Probe_{A,\delta_i}$ be the set of digest values that Alice sends to Bob to verify the correct image in the point δ_i. Every digest is calculated using the generator $g \in G_{\sigma\pi(\delta_i)}$, thus, at the i-th position we have the digest or hash of the result of $g_{\sigma\pi(\delta_i)}^{\beta_{\sigma\pi(\delta_i)}\alpha_i}$.

- $Probe_{A,\delta_i} = (H(g_{\sigma(\pi(\delta_i))}^{\beta_{\sigma(\pi(\delta_i))}{}^{\alpha_1}}), \cdots, H(g_{\sigma(\pi(\delta_i))}^{\beta_{\sigma(\pi(\delta_i))}{}^{\alpha_n}}))$
- $Probe_{B,\delta_i} = (H(g_{\pi(\sigma(\delta_i))}^{\alpha_{\pi(\sigma(\delta_i))}{}^{\beta_1}}), \cdots, H(g_{\pi(\sigma(n))}^{\alpha_{\pi(\sigma(n))}{}^{\beta_n}}))$

Alice and Bob are able to agree on the same value on position $\pi\sigma(\delta_i) = \sigma\pi(\delta_i)$ since $H(g_{\sigma\pi(\delta_i)}^{\beta_{\sigma\pi(\delta_i)}\cdot\alpha_{\sigma\pi(\delta_i)}}) = H(g_{\pi\sigma(\delta_i)}^{\alpha_{\pi\sigma(\delta_i)}\cdot\beta_{\pi\sigma(\delta_i)}})$. But this proposition makes the verification vulnerable to an attack where the attacker figures out the commitment images since both probe tuples have an equal value on the aforementioned position. Two probe tuples are needed for both Alice and Bob, using two different salts. This way, the attacker cannot figure out the right value as he has no information of what digests are equal, being unable to reveal the right image of δ_i.

Eventually, taking the vulnerability into consideration, the new probing phase consits of Alice sending $Probe_{A,\delta_i,salt_1}$ to Bob and keeping $Probe_{A,\delta_i,salt_2}$ for herself. Now Alice is able to distinguish the right image of δ_i in $Probe_{B,\delta_i,salt_2}$ comparing it to $Probe_{A,\delta_i,salt_2}$. Bob is able to distinguish the image of δ_i in $ProbeA, \delta_i, salt_1$ comparing it to $Probe_{B,\delta_i,salt_1}$. Attacker knows $Probe_{A,\delta_i,salt_1}$ and $Probe_{B,\delta_i,salt_2}$ thus the position $\pi\sigma(\delta_i)$ cannot be determined as there's no equality or collision as salts are different for each probe tuple he has.

2.2 An implementation for $G = Z_n$

An interesting procedure to investigate further on the security of the scheme is to study the case when G is taken as the multiplicative group of integers modulo n. For the following case represent $a \equiv b \pmod{n}$ as $a \equiv_n b$:

- $X = (g_1, \cdots, g_n) \in {Z_1}^* \times \cdots \times Z_n^*$
- $\delta = (\delta_1, \cdots, \delta_k) \in [1, Deg(P)]^k$
- $P =< \sigma_1, \cdots, \sigma_r >, \quad Deg(P) = |X| = n$
- $|P| = lcm(c_1, \cdots, c_s) = q \iff \sum_{i=1}^{s} |c_i| = Deg(P) = n$

Alice and Bob select $\sigma, \pi \in P$ and their private key tuple named $priv$.

- $\sigma, \pi \in P$

- $priv_A = (\alpha_1, \cdots, \alpha_n) \in {Z_1}^* \times \cdots \times Z_n^*$
- $priv_B = (\beta_1, \cdots, \beta_n) \in {Z_1}^* \times \cdots \times Z_n^*$

Now both setup their public key by transforming and permuting X with their private keys and permutations. Public keys are exchanged and the shared tuple computed.

- $Pub_A = (g_{\sigma(1)}^{\alpha_{\sigma(1)}} \equiv_{n_{\sigma(1)}}, \cdots, g_{\sigma(n)}^{\alpha_{\sigma(n)}} \equiv_{n_{\sigma(n)}}) \in Z^*_{n_{\sigma(1)}} \times \cdots \times Z^*_{n_{\sigma(n)}}$
- $Pub_B = (g_{\pi(1)}^{\beta_{\pi(1)}} \equiv_{n_{\pi(1)}}, \cdots, g_{\pi(n)}^{\beta_{\pi(n)}} \equiv_{n_{\pi(n)}}) \in Z^*_{n_{\pi(1)}} \times \cdots \times Z^*_{n_{\pi(n)}}$
- $Shared_{A,B} = (g_{\sigma\pi(1)}^{\beta_{\sigma\pi(1)}} \equiv_{n_{\sigma\pi(1)}}, \cdots, g_{\sigma\pi(n)}^{\beta_{\sigma\pi(n)}} \equiv_{n_{\sigma\pi(n)}} \in Z^*_{n_{\sigma\pi(1)}} \times \cdots \times Z^*_{n_{\sigma\pi(n)}}$
- $Shared_{B,A} = (g_{\pi(\sigma(1))}^{\alpha_{\pi(\sigma(1))}}, \cdots, g_{\pi(\sigma(n))}^{\alpha_{\pi(\sigma(n))}}) \in G_{\pi\sigma(1)} \times \cdots \times G_{\pi\sigma(n)}$

For the last part, Alice and Bob must agree on every commitment point $\delta_i \in \delta$. They build the $Probe$ tuple for the point δ_i that consists of digests of the transforms of the δ_i -th value on the the $Shared$ tuple using every private value of their $priv$ tuple. This is stated as:

- $Probe_{A,\delta_i,salt_1} = (H({g_{\sigma(\pi(\delta_i))}^{\beta_{\sigma(\pi(\delta_i))}}}^{\alpha_1} \equiv_{n_1} ||salt_1), \cdots, H({g_{\sigma(\pi(\delta_i))}^{\beta_{\sigma(\pi(\delta_i))}}}^{\alpha_n}) \equiv_{n_n} ||salt_1) \in Z^{n^+}$
- $Probe_{A,\delta_i,salt_2} = (H({g_{\sigma(\pi(\delta_i))}^{\beta_{\sigma(\pi(\delta_i))}}}^{\alpha_1}||salt_2), \cdots, H({g_{\sigma(\pi(\delta_i))}^{\beta_{\sigma(\pi(\delta_i))}}}^{\alpha_n})||salt_2) \in Z^{n^+}$
- $Probe_{B,\delta_i,salt_1} = (H({g_{\pi\sigma(\delta_i)}^{\alpha_{\pi\sigma(\delta_i)}}}^{\beta_1}||salt_1), \cdots, H({g_{\pi\sigma(\delta_i)}^{\alpha_{\pi\sigma(\delta_i)}}}^{\beta_n})||salt_1) \in Z^{n^+}$
- $Probe_{B,\delta_i,salt_2} = (H({g_{\pi\sigma(\delta_i)}^{\alpha_{\pi\sigma(\delta_i)}}}^{\beta_1}||salt_2), \cdots, H({g_{\pi\sigma(\delta_i)}^{\alpha_{\pi\sigma(\delta_i)}}}^{\beta_n})||salt_2) \in Z^{n^+}$

Alice shares $Probe_{A,\delta_i,salt_1}$ and keeps $Probe_{A,\delta_i,salt_2}$ secret. Bob shares $Probe_{B,\delta_i,salt_2}$ and keeps $Probe_{B,\delta_i,salt_1}$ secret.

Since $Probe_{A,\delta_i,salt_2}$ and $Probe_{B,\delta_i,salt_2}$ have the same digest on the point $\sigma\pi(\delta_i)$, Alice is able to determine the transformation ${g_{\sigma\pi(\delta_i)}^{\beta_{\sigma\pi(\delta_i)}}}^{\alpha_{\sigma\pi(\delta_i)}} \equiv_{n_{\sigma\pi(\delta_i)}}$. In the same way does Bob, determining the transformation ${g_{\sigma\pi(\delta_i)}^{\beta_{\sigma\pi(\delta_i)}}}^{\alpha_{\sigma\pi(\delta_i)}} \equiv_{n_{\sigma\pi(\delta_i)}}$. This construction works as the underlying operation is commutative on P and on every $Z^*_{n_i}$. However, various issues arise when inspecting the scheme at a lower level as seen in the next section.

3 Security: Distinguishability and Membership of elements

The presented scheme is functional and applicable to real world purposes, however, its security must be analyzed to point out what are the requirements to achieve the best affordable security. In the other hand, security will tell what are the possible chances for an attacker to succeed and break into the communication. Furthermore, presentation of scenarios and environments is crucial to determine and understand what an attacker would attempt to breakthrough. To study such a thing, the chosen approach by the author is to address these concerns one by one.

We talk about distinguishability when the attacker wants to determine which properties does satisfy a random selected element. Distinguishing could lead him to succeed if his chances to guess are likely to occur. On this scheme distinguishability plays an important role specially in the situation where the attacker can distinguish with high probability which element of the public key is a member of a particular group G_i. Concretely in Pub_A the value in the i-th position depends on the permutation σ say $\sigma(i) = j$, so attacker sees an element in G_j in position i. He must find out that the value in position i was originally in position j of X, it was brought to position i, transformed by a trapdoor permutation function and inserted on the public key. This is hard for the attacker, but not impossible.

3.1 Analyzing the structure of the permutation group P

Permutation elements of P constitute one of the core features of this scheme: they are responsible of permuting every transformation applied to elements in X to generate public keys. In addition, they are involved on the shared key material generation as well, and this material is used to build the $Probe$ tuple that Alice and Bob use to commit on a sequence of values. As a result of these details, the attacker can partially succeed in his mission of breaking the scheme if he can retrieve or obtain private permutations selected by Alice or Bob. Let's proceed with an approach using group theory that attempts to represent what the attacker gains from the permutation used to permute any public key:

Let P be a commutative permutation group of order q and degree n where $q, n \in \mathbf{Z}$. Suppose that $Deg(P) = k < n$, as X contains n elements or generators, $\sigma \in P$ has less than n symbols and will inherently fix $n - k$ symbols on Pub_A. Consequently, a necessary condition is $Deg(P) = |X| = n$.

The attacker can sharpen and figure out what group elements can appear in a certain position of Pub_A. This is strongly related to P acting on S transitively. Let $S = \{1, \cdots, n\} \in Z^+$ be the set of integer symbols ranging from 1 to n and define the group action morphism of the G-set P_S as:

$$\phi : P \times S \mapsto S$$

$$\rho \cdot x \mapsto \phi(\rho, x) \quad \rho \in P, x \in S$$

Do not confuse P_S with P_P because if $S = P$ then P_P acts on itself, giving the permutation representation (Cayley's Theorem), thus the group action would be transitive as $\phi : P \times P \mapsto P$ gives the automorphism group. Focus the interest on P_S as it defines a bijection in the set of symbols S.

Eventually, define the group action of P_S as transitive $\iff \forall x, y \in S \quad \exists \rho \in \phi : \rho(x) = y$.

This is, if the action on P_S is transitive every symbol in S is related to every other symbol by the equivalence $x \sim y$ as permutation elements in P have a single orbit $\mathcal{O}$ (Burnside's Lemma)[2]. Otherwise, the action of P_S is intransitive and $\rho(x) \neq y$ in certain cases so position x in Pub is restricted to an orbit containing less than n elements[2]. The attacker can exploit this fact to reduce the number of group candidates for a certain element of Pub.

It results that the group action of P_S is intransitive when P is generated by disjoint permutations cycles. Let $P =< \rho >$ and $\rho = c_1, \cdots, c_r$ Then $Deg(P) = \sum_{i=1}^{k} n_{c_i}|c_i|$ where $|c_i|$ denotes the length of the cycle c_i and n_{c_i} how many cycles of length i are in ρ. P is clearly commutative, but as it's generated by a permutation generator consisting of disjoint cycles, then a symbol in X of c_i cannot appear in c_k, thus its group action is intransitive. This result is crucial as the attacker knows that in position i it can have $|\mathcal{O}_i|$ candidates, which is inherently less than n. The goal is to create a permutation group that is transitive where every symbol or position i has n candidates, thus there would exists an unique orbit of size n. Therefore $|\mathcal{O}| = Deg(P) = n$. This way, every group element in Pub_A could belong to any group G_i, as a consequence, the attacker cannot elaborate a reduced list of group candidates for the position i in Pub_A using the intransivity criteria. The final conclusion is that P must be a permutation group generated by a single ρ, thus $|P| = Deg(P)$

3.2 Ordering Chains

In the previous section the attacker could gain information if the action on P_S is intransitive. Now he's in the position to investigate further as in the scheme P acts on S transitively. He could be confused by the fact that every position of Pub_A can contain any element of any group G_i where $1 \leq i \leq n$. But he's clever enough to realize that logic comparison is used to classify permutations via ordering chains. A general method is presented that can be applied when P is either transitive or intransitive. Of course when P is intransitive, this method will have a bigger impact on the breakthrough since it allows the attacker to discard group candidates with more ease.

Return to the definition of the scheme when $G_i = Z_{n_i}^*$. Due the group action of P_S being transitive, let $\sigma(i) = j$ so the element on position i of Pub_A is the residue h_j such that $g_j^{\alpha_j} \equiv_{n_j} h_j$. The attacker knows that h_j may belong to any group when h_j satisfies $h_j \in G_k \iff h_j < n_k$. Thus he builds an ordering chain that defines the membership of the element h_j respect to all G_i.

$C_{h_j} = n_1 \leq n_2 \leq \cdots h_{\sigma(i)} < n_{\sigma(i)} < \cdots n_n$

The previous chain is the trivial chain for the residue $h_j \in G_j$. Note that $h_j < n_j$ and since $h_j > n_{j-1}$ you are done because h_j it's greater than any other

modulus less than n_j. But what if $h_j < n_{j-1}$? And what if $h_j < n_1$? Furthermore, there exists an abstract method to classify the number of possible chains for a residue $h_j \in G_j$. This method can be used to estimate the probability of a residue h_j having a particular chain.

$$\begin{cases} C_1 = h_1 < n_1 < \cdots < n_n \\ C_2 = \begin{cases} n_1 < h_2 < n_2 < \cdots < n_n \\ h_2 < n_1 < n_2 < \cdots < n_n \end{cases} \\ \cdots \\ C_n = \begin{cases} n_1 < \cdots < h_n < n_n \\ n_1 < \cdots < h_n < n_{n-1} < n_n \\ \cdots \\ h_n < n_1 < \cdots < n_n \end{cases} \end{cases}$$

An important conclusion is that there are j possible ordering chains for the element h_j. Obviously, when observing Pub_A the attacker selects one of these chains for every h_j, building the chain himself by the criteria exposed above.

The attacker realizes that C_{h_j} can be used to distinguish the membership of the congruence h_j respect to all groups G. His next step is to group these chains to detect possible arrangements using cycle detection or elimination techniques.

3.3 Elimination techinques

Let $A(Pub_A) \in F_2^{n \times n}$ be the adjacency matrix that results from the possible arrangements (permutations) involved in Pub_A. Every row i contains the possible group memberships of the element h_j recall that $\sigma(i) = j$.

$$A(Pub_A)_{i,k} = \begin{cases} 1 \iff h_{\sigma(i)} < n_k \\ 0 \iff h_{\sigma(i)} > n_k \end{cases}$$

The reader can check that the following property is always satisfied when P acts on S transitively:

$\forall i \in [1, n] \rightarrow A(Pub_A)_{i,\sigma(i)} = 1$

Thus if every row of $A(Pub_A)$ has distinct degree then a triangular matrix is obtained (after reordering/sorting). It is clear that the attacker can distinguish every element in Pub_A by elimination. This is the case when every C_i is the trivial chain. The scenario can be complex, as P is transitive, perfect indistinguishability is achieved when every $A(Pub_A)_{i,j} = 1$ masquerading every residue.

If P is intransitive, the attacker knows that the symbol i has an orbit of size $k < n$. Let $\mathcal{O}_i$ be the orbit associated with the symbol i. He obtains the chain for $h_{\sigma(i)=j}$ by the following rule:

$$A(Pub_A)_{i,k} = \begin{cases} 1 \iff k \in \mathcal{O}_i \wedge h_j < n_k \\ 0 \iff k \notin \mathcal{O} \end{cases}$$

This is why the author recommends that P acts on S transitively. Intransitivity weakens the security of the scheme by letting the attacker to distinguish residues or group elements by eliminating candidates taking $\mathcal{O}_i$ into account. The elimination process works because a membership candidate i.e $A(Pub_A)_{i,k} = 1$ turns into $A(Pub_A)_{i,k} = 0$ if k is not in the orbit $\mathcal{O}_i$. A naive idea is to think that making P_S intransitive would result in $Deg(P) = n$ but $|P| > n$. The ability of the attacker to sneak in the *Probe* tuple depends on breaking DLP instances found in Pub_A and for that he must distinguish the right group for h_j, thus action intransitivity reduces the candidates for every point/position.

3.4 Brief example

Let's start by solving the trivial case mentioned above where every chain C_{h_j} is trivial. Let $\sigma = (13452)$ so $Pub_A = (h_3, h_1, h_4, h_5, h_2) \in G_3 \times \cdots \times G_2$ where $h_i \equiv_{n_i} g_i^{\alpha_i}$. Attacker must write a chain for every h_i. As this example works with trivial chains he writes up:

$$\begin{cases} C_{\sigma(1)} = C_3 = n_1 < \cdots < h_3 < n_3 < \cdots < n_n \\ C_{\sigma(2)} = C_1 = h_1 < n_1 < \cdots < n_n \\ \\ \cdots \\ C_{\sigma(5)} = C_2 = n_1 < h_2 < n_2 < \cdots < n_n \end{cases}$$

Now he derives the adjacency matrix $A(Pub_A) \in F_2^{n \times n}$ from the system of chains:

$$A(Pub_A) = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

And he trivially recovers $\sigma = (13452)$. This method is the principle to distinguish residues and their memberships. Note that when attacker encounters non-trivial chains he could obtain more than one $\sigma \in P$, yet he's still winning as he can distinguish some residues h_i if $A(Pub_A)$ is not full of $1's$ (perfect indistinguishability). There is another crucial point regarding probabilities, as the attacker has different chances of seeing a particular ordering chain for an element h_i. As a consequence, it's a must to determine what are the chances for $A(Pub_A)$ to satisfy perfect indistinguishability, between others like the chances to obtain trivial chains.

3.5 Probability on ordering chains

First, how many adjacency matrices $A(Pub_A) \in F_2^{n \times n}$ are out there? Recall that h_i has i chains, so every row r_i has i possible candidates. Then $\prod_{i=1}^{n} i = n!$ matrices.

Let K be the probability that $A(Pub_A)$ results in an adjacency matrix where every row r_i represents the trivial chain of $h_{\sigma(i)}$. Then:

$$P[K] = \prod_{i=2}^{n} \frac{\varphi(n_i) - \varphi(n_{i-1})}{\varphi(n_i)}$$

$$= \prod_{i=2}^{n} (1 - \frac{\varphi(n_{i-1})}{\varphi(n_i)})$$

Therefore, the probability that a random h_j belongs to groups where their modulus is smaller than n_j turns to be bigger when the distance $n_j - n_k, \quad n_k < n_j$ is small.

The other relevant case is when every $A(Pub_A)_{i,k} = 1$ as it grants perfect indistiguishability due to the attacker not being able to distinguish h_j's membership because every element in Pub_A belongs to G_1, so immediately belongs to every other group as $n_1 < \cdots < n_n$.

Let K be the probability that every $A(Pub_A)_{i,k} = 1$. Then:

$$P[K] = \prod_{i=2}^{n} (1 - \frac{\varphi(n_i) - \varphi(n_1)}{\varphi(n_i)})$$

$$= \prod_{i=2}^{n} \frac{\varphi(n_1)}{\varphi(n_i)}$$

There are complex cases not belonging to trivial chains or all residues being in G_1. A formula is needed to obtain the probability that the residue h_j belongs to the group G_k but not to G_i, where $i = k - 1$. This is equal to the difference of the elements in G_k and G_i divided by the number of elements in G_j as the residue set where h_j belongs is on G_j.

$$P[h_j \in G_k \wedge h_k \notin G_i] = P[n_i < h_j < n_k < \cdots < n_j] = \frac{\varphi(n_k) - \varphi(n_i)}{\varphi(n_j)}$$

The aforementioned formula is useful when dealing with all the residues h_j that are in between modulus that are less than n_j.

3.6 Composition via Chinese Remainder Theorem

When all the groups include a prime modulus, the Chinese Remainder Theorem can be applied to solve a discrete logarithm in G_N that recovers all the exponents in $priv_A$. It is not a good attack when all p_i are big in bit-size but it can be useful when these are not big enough. Let $N = \prod_{i=1}^{n} p_i$, construct an unit in G_N that later defines the congruence to h_N via exponentiation:

First do CRT on all the generators in X:

$$g_1 \equiv_{p_1} g_1 \tag{1}$$
$$\cdots \tag{2}$$
$$g_n \equiv_{p_n} g_n \tag{3}$$

Then we have the representation of the base g_N that's used to recover the exponent α_N solving the discrete logarithm on the residue h_N. For that recover first h_N as:

$$g_1^{\alpha_1} \equiv_{p_1} h_1 \tag{4}$$
$$\cdots \tag{5}$$
$$g_n^{\alpha_n} \equiv_{p_n} h_n \tag{6}$$

From this system recover $h_N \equiv_N g_N^{\alpha_N}$. Then obtain each α_i as:

$$\alpha_N \equiv_{\varphi(p_1)} \alpha_1 \tag{7}$$
$$\cdots \tag{8}$$
$$\alpha_N \equiv_{\varphi(p_n)} \alpha_n \tag{9}$$

Requirements are that the attacker knows which residues belong to the right group, this is, he must find out the permutation σ for mounting this attack. He could be able to solve n discrete logarithms by computing just 1 dlog in the big field, but in complexity is better to separate a big coprime modulus into prime factors via CRT, this is why this attack is not recommendable.

4 Complexity

At this time the attacker has a potential method for distinguishing random elements depending on the structure and properties of the public key and the permutation group P. However, the underlying problem where this scheme is based must be analyzed as every presented element in Pub_A has been transformed by a DLP based trapdoor-permutation involving a private exponent.

From an arbitrary Pub_A and a commitment point $\delta_i \in \delta$ the attacker solves the discrete logarithm of the residue in a quantity of groups less than $\mathcal{O}$ this is:

$$g_{\sigma(\delta_i)^{\alpha_{\sigma(\delta_i)}}} = h_j \in G_i, \quad 1 \leq i \leq \mathcal{O}$$

It is obvius that when he finds the right G_j it will recover the right α_j. As this is the brute force approach, the whole attack needs solving $|\delta| \cdot \mathcal{O}$ discrete logarithms because for every commitment point there are $\mathcal{O} = n$ mappings of distinct residues when P_S is transitive.

When $A(Pub_A)_{i,j} = 1 \quad \forall i, j$, this is all $1's$ in the matrix this is the perfect indistinguishability case. In this case there are $n!$ permutations, but as the attacker knows that $\sigma \in P$ this is equal to having $|P|$ candidates. This scenario represents the best case for hiding the permutation σ as it conduces the attacker to the brute force approach.

In the case that $A(Pub_A)$ can be reduced via elimination or reduction techniques the formula for determining how many permutation cycles can be constructed via reduction is:

$$N(cycles) = \prod_{i=0}^{n-1} |m(G_{i+1}) - i|$$

where $|m(G_{i+1})|$ is the number of residues in Pub_A that belong to G_{i+1}. The proof of the previous argument is hard to give but is constructed taking into account the intersection of membership sets. Check that the perfect indistinguishability case is satisfied as $N(cycles) = n! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$ as $|m(G_1)| = \cdots |m(G_n)|$. The case where all the ordering chains are trivial, this is $n_{j-1} < h_j < n_j$ is equal to $N(cycles) = 1$ as $|m(G_1)| = 1, |m(G_2)| = 2, \cdots, |m(G_n) = n|$. As the formula seems to work with these two trivial case as non-trivial cases as well (see section 4), the author concludes that it should be taken into account when estimating the number of cycles involved in an adjacency matrix that represents the ordering chain of Pub_A. An elimination attack can cause relevant impact if recovering σ from the candidates: the complexity of determining the number of cycles is trivial, and the algorithm of testing-composing every permutation resulting from a mapping table is also trivial. This leads the attacker to solve $|\delta|$ discrete logarithms, one for every commitment point.

Complexity goes beyond when dealing with probabilities as in the real world there are public keys that have a specific ordering chain more often than others. Estimation on these chains must be done to the point that the system can be weak if the presented ordering chains are solvable or can be reduced into a new ordering system that turns these chains into solvable. The scheme must be constructed with security in mind, as the perfect indistinguishability is achieved when every residue is in G_1, this is equal to present multiple $Z^*_{p_i}$ where p_1 is big enough and the rest of primes bigger than p_1 are close to it. Then the outcomes of the probability formulas exposed in the previous section tend to satisfy that residues $h_j \in G_1$. To mount the scheme using these prime modulus think on intervals with consecutive primes where the first one is big enough in bit-size.

5 An elimination attack involving non-trivial chains

Every system of ordering chains where the solution depends on elimination or reduction techniques is called non-trivial. Our goal is to *hide* the permutation $\sigma \in P$ that has been used to permute Pub_A. If the attacker obtains σ he gains information about the membership of all the residues in Pub_A. As we have seen, every public key can be expressed as an adjacency matrix that express the system of ordering chains of the residues in Pub_A. Note that in this example residues are ignored, the resulting ordering chain has been arbitrarily chosen. In a real case scenario, the attacker must build the system of chains himself, it is not complicated as it's based on logic comparison. Moreover, we are going to represent Alice in this example.

Let $P = C_5$, then define $\sigma = (13524)$ as a permutation element from the cyclic group of five units. Transform the pubic set of generators $X = (g_1, \cdots, g_n)$ with our private key $priv_A = (\alpha_1, \cdots, \alpha_n)$ via modular exponentiation s.t $h_i \equiv_{n_i} g_i^{\alpha_i}$. Then our public key Pub_A is obtained by applying the permutation σ s.t $Pub_A = \sigma(X.priv)$. Call the ordering chain of $X.priv_A$ as $A(X.priv_A) \in F_2^{5\times 5}$:

$$A(X.priv_A) \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Apply σ to obtain the ordering chain on the permuted residues in the public key

$$A(Pub_A) = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

The attacker sees only $A(Pub_A)$ as the permutation σ remains secret. He knows about ordering chains and is capable of mounting an elimination attack to obtain a reduced subset of possible permutations. Call candidate to any of these permutations that are in C_5. First he writes up a mapping for each $i \leq Deg(C_5)$ based on $A(Pub_A)_{i,j} = 1 \iff h_i < n_j$:

$$1 \to 2, 3, 4, 5 \tag{10}$$
$$2 \to 4, 5 \tag{11}$$
$$3 \to 4, 5 \tag{12}$$
$$4 \to 1, 2, 3, 4, 5 \tag{13}$$
$$5 \to 2, 3, 4, 5 \tag{14}$$

He realizes that some mapping candidates can be eliminated and obtains a new mapping table:

$$1 \to 2, 3 \tag{15}$$
$$2 \to 4, 5 \tag{16}$$
$$3 \to 4, 5 \tag{17}$$
$$4 \to 1 \tag{18}$$
$$5 \to 2, 3 \tag{19}$$

This table is equal to the following adjacency matrix that results from eliminating mappings in the previous table:

$$M = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Eventually, he generates all the permutations built from the final symbol mapping table. These permutations are: $(124)(35), (134)(25), (12534), (13524)$. Note that attacker obtains σ as it's on the last position of the previous list of candidates. σ is the unique permutation that belongs to C_5 in those candidates, which attacker distinguishes with ease. Check that $N(cycles) = 4$, this is $|m(G_1)| = 1, |m(G_2)| = 3, |m(G_3)| = 3, |m(G_4)| = 5, |m(G_5)| = 5$ then $N(cycles) = 1 \cdot 2 \cdot 1 \cdot 2 \cdot 1 = 4$ possible permutations as seen in section 4.

6 A complete example with $n = 5$ groups

Let X be the public set of generator $X = (2, 3, 5, 3, 2) \in Z_{11} \times Z_{17} \times Z_{23} \times Z_{31} \times Z_{37}$. The private exponent tuples are $priv_A = (7, 15, 9, 11, 35), priv_B = (3, 11, 5, 19, 31)$,. The permutation group is the cyclic group of 5 elements thus $P = C_5$, the selected private permutations are $\sigma = (13524), \pi = (15432)$, and the commitment point set is $\delta = (1, 3, 5)$.

$X \cdot priv_A = (7, 6, 11, 13, 19)$ and $X. \cdot priv_B = (8, 7, 20, 12, 22)$ via modular exponentiation. Now construct public keys permuting with σ, π

$Pub_A = \sigma \cdot X \cdot priv_A = (11, 13, 19, 7, 6)$ and $Pub_B = \pi \cdot X \cdot priv_B = (22, 8, 7, 20, 12)$. Alice exchanges her public key with Bob as Bob does with Alice. Both parties compute the shared tuple as follows:

$Shared_{A,B} = \sigma \cdot Pub_B = (7, 20, 12, 22, 8)$

$Shared_{B,A} = \pi \cdot Pub_A = (6, 11, 13, 19, 7)$

For the commitment point $\delta = 1$ Alice computes the probe tuple as follows:

$Probe_{A,\delta_1} = (7^7 \equiv_{11} 6, 7^{15} \equiv_{17} 5, 7^9 \equiv_{23} 15, 7^{11} \equiv_{31} 20, 7^{35} \equiv_{37} 16)$

$Probe_{B,\delta_1} = (6^3 \equiv_{11} 7, 6^{11} \equiv_{17} 5, 6^5 \equiv_{23} 2, 6^{19} \equiv_{31} 6, 6^{31} \equiv_{37} 31)$

It's obvious that both users discover that the image in $\delta_1 = 1$ is 5, located on both probing tuples at position 2 since $\sigma\pi(1) = 2 = \pi\sigma(1)$. Probing tuples must use hash functions and two salts, thus two probing tuples are needed as seen in the Scheme construction section, but in this example this procedure is simplified. For the rest of δ points, the mechanic is the same, take the position δ_i on the shared tuples, then apply every private exponent, hash the residues + salt and send back to the other party to check the commitment value. Note that this is just an example that is easily broken but it demonstrates the insight behind this scheme.

References

[1] W. Diffie, M. Hellman New directions in cryptography Journal IEEE Transactions on Information Theory archive Volume 22 Issue 6, November 1976 Page 644-654 `https://ee.stanford.edu/~hellman/publications/24.pdf`

[2] Stanley, Richard P. Enumerative Combinatorics Volume 1 `http://www-math.mit.edu/~rstan/ec/ec1.pdf`