Reductionist Security Arguments for Public-Key Cryptographic Schemes Based on Group Action

Anton Stolbunov
Department of Telematics
Norwegian University of Science and Technology (NTNU)
Anton@item.ntnu.no

Abstract

We provide reductionist security arguments for a key agreement protocol $\mathcal{KA}$, which is the Diffie-Hellman key agreement protocol generalized to the context of a group action on a set, and for a public-key encryption scheme $\mathcal{PE}$, which is the “hashed” ElGamal scheme generalized for a group action on a set. For the $\mathcal{KA}$ protocol we use the notion of session key security in the authenticated links model, proposed by Canetti and Krawczyk. For the $\mathcal{PE}$ scheme we use a version of the semantic security notion proposed by Goldwasser and Micali. We prove that the security of the $\mathcal{KA}$ protocol and the $\mathcal{PE}$ scheme is based on the decisional Diffie-Hellman group action problem, defined later in this paper. The $\mathcal{PE}$ scheme security also depends on the entropy smoothing property of the hash function family used in the scheme.

Keywords: key establishment, public-key encryption, provable security, reduction, group action.

1 Introduction

The formulation of public-key cryptographic schemes in terms of a semigroup action on a set allows to use new mathematical structures for implementing the schemes. Several implementations have been proposed that use (semi)group actions different from the exponentiation in a cyclic group \[1, 2, 3, 4\]. A common feature of these cryptographic schemes is that their security goes beyond the conventional discrete logarithm problem and is based on some new computational problems. The new problems might prove themselves to be asymptotically harder than those used nowadays. For instance, some known discrete logarithm solvers are difficult to adapt for the computational problems discussed in Section \[4\].

To identify the computational problems constituting the basis for security of a cryptographic scheme, one should construct a reduction from a particular computational problem(s) to an attack against the cryptographic scheme. This

This paper was presented at the NISK-2009 conference.
reduction is often called a security proof, however Koblitz and Menezes have recently advocated a more accurate name: a reductionist security argument \[5, 6\].

Our paper provides a reductionist security argument for a key agreement protocol \(\mathcal{KA}\) based on a group action on a set. For this we firstly define the protocol \(\mathcal{KA}\) and the computational problem DDHAP (decisional Diffie-Hellman group action problem, defined in Section 4) that is used in the reduction. We then proceed with defining a security notion, i.e. a very general attack, and show that this attack, if successful, can be used to solve the DDHA problem. We thus conclude that attacking the \(\mathcal{KA}\) protocol is not easier than solving the DDHA problem.

Formalizing the security of cryptographic protocols turned out to be more complicated than that of cryptographic primitives. Among the main reasons for this is the interactive nature of the protocols in the presence of two or more participants. This opens the possibility for many different types of attacks that use concurrent sessions, participant collusions, corrupt communication channels and so on. Bellare and Rogaway presented the first formal security notion for key agreement protocols \[7\]. Despite of its later extensions by various authors, the model appears to be rather difficult to work with. In our paper we will use a key agreement protocol security model proposed by Canetti and Krawczyk \[8, 9\]. The security in this model is formalized via the infeasibility for an adversary to distinguish between the real value of the session key and an independent random value. The adversary is modelled to have essentially the same characteristics as those specified by the Dolev-Yao threat model \[10\]. There exist more recent works on the topic of key agreement protocol security, e.g. by Kudla and Paterson \[11\], but we consider the Canetti-Krawczyk model to be well-suited and sufficient for the scope of this paper.

Another important result of this paper is a reductionist security argument for a public-key encryption scheme \(\mathcal{PE}\) based on a group action on a set. We use a version of the semantic security notion \[12\] called indistinguishability of encryptions in a chosen-plaintext attack (IND-CPA). The equivalence of the semantic security and the IND-CPA security notions has been shown by Goldreich \[13\]. We prove that the \(\mathcal{PE}\) scheme is secure in the sense of IND-CPA if the DDHA problem is hard and the hash function family is entropy smoothing.

## 2 Notation

We start with the basic notation. Let \(G\) be a finite commutative semigroup. We will omit the (semi)group operation sign, writing \(gh\) for the product of elements \(g, h \in G\). For a set \(X\), a semigroup action of \(G\) on \(X\) is a map

\[
G \times X \rightarrow X
\]

\[
(g, x) \mapsto g * x
\]

which satisfies the associativity property \((gh) * x = g * (h * x)\) for all \(g, h \in G\) and \(x \in X\). When \(G\) is a group, the group action of \(G\) on \(X\) also satisfies \(e * x = x\) for the identity element \(e \in G\) and all \(x \in X\). An example of a group action is provided in Table 1. The orbit of a set element \(x \in X\) is the subset \(G * x = \{g * x \mid g \in G\}\). When \(G\) is a group, the orbits are equivalence classes on \(X\) \[14\, Proposition 4.1.2\].

By \(a \leftarrow b\) we denote the assignment of a value \(b\) to a variable \(a\). By \(a \leftarrow^R G\) we mean that \(a\) is sampled from the uniform distribution on the set of elements of \(G\). We write \# \(S\) for the number of elements in \(S\). By \(\oplus\) we denote the bitwise XOR operation.
Table 1: Action of $G = \mathbb{Z}_5^*$ on $X = \{3, 4, 5, 9\} \subset \mathbb{Z}_{11}^*$ by exponentiation.

<table>
<thead>
<tr>
<th>Elements of $G$</th>
<th>Permutations on $X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$(3)(4)(5)(9)$</td>
</tr>
<tr>
<td>2</td>
<td>$(3\ 9\ 4\ 5)$</td>
</tr>
<tr>
<td>3</td>
<td>$(3\ 5\ 4\ 9)$</td>
</tr>
<tr>
<td>4</td>
<td>$(3\ 4)(5\ 9)$</td>
</tr>
</tbody>
</table>

Figure 1: Key agreement protocol $\mathcal{K}A$.

As a general rule, we assume that all the discussed algorithms take descriptions of $G$ and $X$, and a fixed element $x \in X$, as part of implied system parameters. By descriptions of $G$ and $X$ we mean the information needed to implement the operations involved in the algorithms, i.e. the random sampling from $G$, the action of $G$ on $X$, the semigroup operation etc.

3 Cryptographic Schemes

Since the cryptographic schemes proposed in this section do not require inverses and the identity element in $G$, we let $G$ be a finite commutative semigroup acting on a set $X$.

Key Agreement Protocol Based on Semigroup Action

We define a key agreement protocol $\mathcal{K}A$ depicted on Fig. 1. Here $A$ and $B$ are identifiers of the participants Alice and Bob, and $i$ is a unique session identifier. In this protocol Alice is the initiator and Bob is the responder. By $\mathcal{A}$ and $\mathcal{B}$ we denote the algorithms run by Alice and Bob, respectively.

Due to the commutativity of $G$ and the associativity of the action, the following holds:

$$k_A = a \ast \beta = a \ast (b \ast x) = (ab) \ast x = (ba) \ast x = b \ast (a \ast x) = b \ast \alpha = k_B ,$$  \hspace{1cm} (1)
**Key generation**

<table>
<thead>
<tr>
<th>Input: -</th>
</tr>
</thead>
<tbody>
<tr>
<td>sk ← ( G )</td>
</tr>
<tr>
<td>( y \leftarrow \text{sk} \ast \text{x} )</td>
</tr>
<tr>
<td>( k \leftarrow \mathcal{K} )</td>
</tr>
<tr>
<td>pk ← (( y, k ))</td>
</tr>
<tr>
<td>Output: sk, pk</td>
</tr>
</tbody>
</table>

**Encryption**

<table>
<thead>
<tr>
<th>Input: pk, m</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a \leftarrow \mathcal{G} )</td>
</tr>
<tr>
<td>( u \leftarrow a \ast y )</td>
</tr>
<tr>
<td>( h \leftarrow \mathcal{H}_k(u) )</td>
</tr>
<tr>
<td>( z \leftarrow a \ast x )</td>
</tr>
<tr>
<td>( c \leftarrow h \oplus m )</td>
</tr>
<tr>
<td>( ct \leftarrow (c, z) )</td>
</tr>
<tr>
<td>Output: ct</td>
</tr>
</tbody>
</table>

**Decryption**

<table>
<thead>
<tr>
<th>Input: sk, pk, ct</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u \leftarrow \text{sk} \ast z )</td>
</tr>
<tr>
<td>( h \leftarrow \mathcal{H}_k(u) )</td>
</tr>
<tr>
<td>( m \leftarrow h \oplus c )</td>
</tr>
<tr>
<td>Output: m</td>
</tr>
</tbody>
</table>

**Figure 2: Public-Key Encryption Scheme \( \mathcal{PE} \).**

so \( A \) and \( B \) output the same session key.

If we choose \( G \) and \( X \) in a way as in Table [1] it becomes clear that the protocol \( \mathcal{KA} \) is a generalization of the key agreement protocol proposed by Diffie and Hellman [15]. A simplified version of the \( \mathcal{KA} \) protocol that did not contain session identifiers, party identifiers and “erase” statements, was proposed by Monico [2].

**Public-Key Encryption Based on Semigroup Action**

We now generalize the ElGamal public-key encryption scheme to the context of semigroup action. An approach proposed by Monico [2] requires the set \( X \) to be a group in order to mask a message \( m \in X \). In contrast with this, we use a ”hashed” version of the ElGamal encryption scheme, which eliminates these restrictions on \( X \) and \( m \). For a fixed message length \( w \), the message space is the set of bit strings \( \{0,1\}^w \), and thus we can write \( m \in \{0,1\}^w \). We use a hash function family \( \mathcal{H} = \{H_k: k \in K\} \) indexed by a finite set \( K \), such that each \( H_k \) is a function

\[
H_k: G \ast x \rightarrow \{0,1\}^w.
\]

We define a public-key encryption scheme \( \mathcal{PE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}) \) on Fig. 2. Some of notations we use are: \( \text{sk} \in G \) is a secret key, \( \text{pk} \in X \times K \) is a public key and \( \text{ct} \in \{0,1\}^w \times X \) is a ciphertext. We also have that \( x, y, z, u \in X \).

The encryption scheme \( \mathcal{PE} \) is sound, that is to say, for all pairs \( (\text{sk}, \text{pk}) \) which can be output by \( \mathcal{K} \) and for all \( m \in \{0,1\}^w \) we have that \( \mathcal{D}(\text{sk}, \text{pk}, \mathcal{E}(\text{pk}, m)) = m \). Indeed, we can write \( H_k(a \ast y) = H_k(\text{sk} \ast z) \) since

\[
a \ast y = a \ast (\text{sk} \ast x) = (a \text{sk}) \ast x = \text{sk} \ast (a \ast x) = \text{sk} \ast z.
\]

**4 Computational Problems**

In this section we define computational problems that will serve as the basis for the security of the cryptographic schemes defined above.

We note that a generic semigroup action has some disadvantages, as compared to a group action, when viewed from the security perspective. Without inverses in \( G \), an action by \( a \in G \) on \( X \) might not be a permutation, and so \( \alpha = a \ast x \) does not necessarily imply \( x \in G \ast \alpha \). In other words, the orbits \( G \ast x \) and \( G \ast \alpha \) might
be of different sizes. This may, for instance, reduce the session key space in the KA protocol. An easy example is the semigroup of the subsets of the set of two elements \{p, q\} with the join composition \(\cup\), acting on itself. The orbit of \{\}\ has four elements, whereas the orbit of \{p, q\} has only one element. Thus, semigroups which are not groups should be chosen carefully. On the other hand, a group action eliminates this kind of flaw, as the orbits are equivalence classes. We note that if \(G\) is a group acting on a set \(X\), then due to the orbit-stabilizer theorem \[14, Proposition 4.1.2\], we have that
\[
\# G \ast x = [G : G_x],
\]
where \(G_x = \{g \in G \mid g \ast x = x\}\) is the stabilizer of \(x\). That means that the random sampling \(a \leftarrow G\) followed by the group action \(u \leftarrow a \ast x\) produces the same result as the random sampling from the orbit \(u \leftarrow G \ast x\).

For the rest of this paper we let \(G\) be a finite abelian group acting on a set \(X\), and \(x\) a fixed element in \(X\).

**Problem 1.** *Group Action Inverse (GAI) Problem:* given a randomly chosen element \(y \in G \ast x\), find a group element \(g \in G\) such that \(g \ast x = y\).

Problem 1 is a generalization of the discrete logarithm problem. To show this, consider \(X\) to be a multiplicative group, \(x\) a generator of a cyclic subgroup and \(G = \mathbb{Z}_{\text{ord } x}\). However there exist instances of \(G\) and \(X\) which are outside of this trivial scope. In order to adapt conventional algorithms such as the Pollard rho, Pollard kangaroo, Pohlig-Hellman or index calculus for the generic GAI problem, one may try to define a group on \(G \ast x\) which is isomorphic to \(G/G_x\). This is achieved by choosing \(x\) to be the identity element and defining the product of any two elements \(y, z \in G \ast x\) as
\[
y \cdot z = (ab) \ast x, \tag{2}
\]
where \(a\) and \(b\) are such that \(y = a \ast x\) and \(z = b \ast x\). But, when both \(a\) and \(b\) are not known, the multiplication (2) is equivalent to Problem 2 and is hard for some instances of \(G\) and \(X\). Thus it is still a question how the named discrete logarithm solvers can be adapted for the GAI problem.

**Problem 2.** *Computational Diffie-Hellman Group Action (CDHA) Problem:* given elements \(y = a \ast x\) and \(z = b \ast x\), where \(a\) and \(b\) are chosen at random from \(G\), find \((ab) \ast x\).

**Problem 3.** *Decisional Diffie-Hellman Group Action (DDHA) Problem:* given a triple \((y, z, u) \in X^3\) sampled with probability 1/2 from one of the two following probability distributions:
- \((a \ast x, b \ast x, (ab) \ast x)\), where \(a\) and \(b\) are randomly chosen from \(G\),
- \((a \ast x, b \ast x, c \ast x)\), where \(a, b\) and \(c\) are randomly chosen from \(G\),

decide which distribution the triple is sampled from.

---

1. A good example is the action of the class group \(\mathcal{CL}(O_K)\) of an imaginary quadratic field \(K\) on the set \(\mathcal{EL}(\mathbb{F}_p, n)\) of isomorphism classes of elliptic curves over \(\mathbb{F}_p\) with \(n\) points and the endomorphism ring \(O_K\) \[3\].
Problems 2 and 3 are generalizations of the computational Diffie-Hellman problem and the decisional Diffie-Hellman problem, respectively. Using a GAI P solver it is straightforward to construct a solver for the CDHA problem, thus the CDHA problem is not harder than the GAI problem. Similarly, the DDHA problem is not harder than the CDHA problem.

For a DDHAP distinguisher $S$, its probability of returning the correct solution will be denoted by $Pr_{DDH}^S$. $Pr_{DDH}^S$ is a function of a security parameter $s = \log \# G \ast x$. Since the distinguisher $S$ can gain a success probability of $1/2$ by returning a random solution, the advantage of $S$ is defined to be

$$Adv_{DDH}^S = \left| Pr_{DDH}^S - \frac{1}{2} \right|.$$

We can now formulate the following assumption about the computational complexity of the DDHA problem:

**Assumption 1. DDHAP Assumption:** for any polynomial-time DDHAP distinguisher $S$, the advantage $Adv_{DDH}^S$ is a negligible function of $s$.

### 5 Reductionist Security Arguments

#### Session-Key Security of the KA Protocol

To model the security of a key agreement protocol we will use a notion of session-key (SK) security in the authenticated-links adversarial model (AM) proposed by Canetti and Krawczyk [9]. We refer to their paper for a formal definition of the SK security in the AM. Below we provide an outline of this security notion.

A protocol $\Pi$ is modelled as a collection of interactive probabilistic polynomial-time (PPT) algorithms run by the parties. These algorithms are triggered by arriving messages. A session is an instantiation of $\Pi$ run at a party. Note that there can be more parties than roles in $\Pi$, and any number of sessions can be run within each party.

The adversary $I$ is a PPT algorithm that has full control over the communication links. In addition to this, $I$ can:

- activate a session within some party by either sending it an action request message or a protocol message;
- corrupt a party, i.e. learn its current internal state;
- learn the current internal state of the specified session within a party;
- learn the session key output by the specified session;
- perform a test-session query (see below).

The only restriction the AM imposes on $I$ is that it cannot inject or modify messages (except for messages from corrupted parties) and that any message can be delivered at most once.

The notion of SK security captures the idea that such an adversary $I$ does not learn anything about the value of the key of an uncorrupted session. This is

---

2A function $\mu(x)$ is negligible, if for every positive integer $c$ there exists an $N_c > 0$ such that for all $x > N_c$, the following holds: $|\mu(x)| < 1/x^c$. 
formalized via the infeasibility for $I$ to distinguish between the real value of the 
session key and an independent random value. In particular, $I$ participates in the 
following $SK$ experiment. After some period of its regular actions described above, 
the adversary $I$ chooses a session in which it wants to be tested, by issuing a test- 
session query. We then toss a coin and provide $I$ with either the value of the session 
key for the queried session, or with a random value from the key space. $I$ is then 
allowed to continue with the regular actions, but not allowed to expose the test 
session. At the end, $I$ outputs its guess. We will denote by $Pr_{I}^{SK}$ the probability 
that $I$ guesses correctly, and by 

$$Adv_{I}^{SK} = \left| Pr_{I}^{SK} - \frac{1}{2} \right|$$

the $I$’s advantage.

**Definition 1.** A key exchange protocol $\Pi$ is called SK-secure if the following 
properties hold for any polynomial-time adversary $I$ in the AM:

1. If two uncorrupted parties complete matching sessions then they both output 
the same key.

2. $Adv_{I}^{SK}$ is a negligible function of the security parameter $s$.

**Theorem 1.** If the DDHAP assumption holds for the finite abelian group $G$ acting 
on the set $X$, then the $KA$ protocol is SK-secure in the AM.

**Proof.** Our proof will be a generalization of that proposed by Canetti and Krawczyk 
for the case of $X = Z_{q}^{*}$ [9, §5.1].

It has been shown in [1] that two uncorrupted parties in matching sessions output the same session key, what satisfies the first requirement of Definition [1]. To 
show that the second requirement holds for the $KA$ as well, let us assume there is a 
polynomial-time adversary $I$ with a non-negligible advantage $\epsilon$. We now construct

a DDHAP distinguisher $S$ that employs the adversary $I$ as depicted in Algorithm [1].

Consider the case when the $r$-th session is not chosen by $I$ as the test session. 
The distinguisher $S$ outputs a random guess, and thus $Adv_{S}^{DDH} = 0$. Now when 
the $r$-th session is the test session, the definition of Problem [3] ensures that the 
probability distributions observed by $I$ are the same as in the SK experiment, and 
thus $Adv_{S}^{DDH} = \epsilon$. Since this “lucky” case happens with $1/l$ probability, we have 
that in general

$$Adv_{S}^{DDH} = \frac{\epsilon}{l} ,$$

which is non-negligible. Since $S$ is polynomial-time, we have a contradiction with 
the DDHAP assumption. $\square$

**IND-CPA Security of the $PE$ Scheme**

The classical goal of encryption is to preserve the privacy of messages: an adversary 
should not be able to learn from a ciphertext information about its plaintext beyond 
the length of that plaintext. This idea is captured via the notion of semantic security 
of an encryption scheme, which asserts that any polynomial-time adversary cannot 
effectively distinguish between the encryption of two messages of his choosing [12]. 
We will use an equivalent notion, indistinguishability of encryptions in a chosen-
plaintext attack (IND-CPA) [16, §2.2].
**Algorithm 1** DDHAP distinguisher $S$

**Input:** $(y,z,u) \in X^3$

1: $r \leftarrow R \{1,\ldots,l\}$, where $l$ is an upper bound on the number of sessions activated by $I$ in any interaction
2: invoke $I$ and simulate the environment of the SK experiment in the AM, except for the $r$-th activated protocol session
3: upon the activation of the $r$-th session (say it is between Alice and Bob and has a session identifier $i$), let Alice send $(A,i,y)$ to Bob, and let Bob send $(B,i,z)$ to Alice
4: if the $r$-th session is chosen by $I$ as the test session then
5: provide $u$ as the answer to the test query
6: $d \leftarrow I$’s output
7: else
8: $d \leftarrow R \{0,1\}$
9: end if

**Output:** $d$

**Algorithm 2** IND-CPA experiment

1: $(pk,sk) \leftarrow K$
2: $(m_0,m_1,j) \leftarrow I_1(pk)$
3: $d \leftarrow R \{0,1\}$
4: $ct \leftarrow E(pk,m_d)$
5: $d' \leftarrow I_2(j,ct)$

---

For a public-key encryption scheme $\Pi = (K,E,D)$ the IND-CPA experiment is depicted in Algorithm 2. An adversary is viewed as a pair of algorithms $I = (I_1,I_2)$. Algorithm $I_1$ can output some state information $j$ that is then passed to $I_2$. The messages $m_0$ and $m_1$ output by $I_1$ are required to have the same length. By $Pr_I^{IND}$ we denote the probability that $d = d'$ in the IND-CPA experiment. The advantage of $I$ is

$$Adv_I^{IND} = \left| Pr_I^{IND} - \frac{1}{2} \right|.$$

**Definition 2.** A public-key encryption scheme $\Pi$ is said to be secure in the sense of IND-CPA if $I$ being polynomial-time implies that the advantage $Adv_I^{IND}$ is negligible.

---

In our security argument we will also use a property of a hash function family $H$ to be entropy smoothing (ES). The smooth entropy denotes the number of almost uniform random bits in a random variable [17]. The ES hash function should be able to produce almost uniformly distributed outputs by decreasing the output size, as compared to the size of the input. This is formalized via the requirement that any polynomial-time adversary cannot effectively distinguish between the values $(k,H_k(u))$ and $(k,h)$, where $k \in K$, $u \in U$ and $h \in \{0,1\}^w$ are chosen at random, and $U$ is the domain of the hash functions. The ES property is a reasonable assumption for modern ad hoc hash function families [18].
**Definition 3.** Let $\mathcal{H} = \{H_k : k \in K\}$ be an indexed family of hash functions, where each $H_k$ is a function $H_k : U \rightarrow \{0, 1\}^w$ for a fixed $w$. Consider two probability spaces over $K \times \{0, 1\}^w$, namely $P_0$ is the uniform probability space and $P_1$ is induced by the uniform distribution on $K \times U$ and a function $K \times U \rightarrow K \times \{0, 1\}^w (k, u) \mapsto (k, H_k(u))$.

$\mathcal{H}$ is said to be entropy smoothing, when the probability spaces $P_0$ and $P_1$ are computationally indistinguishable, i.e. the advantage $\text{Adv}^{ES}_S$ of any polynomial-time distinguisher $S$, that takes an element of $K \times \{0, 1\}^w$ and outputs a bit, is negligible.

In the following theorem, the hash function’s domain $U$ is the orbit $G \ast x$.

**Theorem 2.** If the DDHAP assumption holds for the finite abelian group $G$ acting on the set $X$, and the hash function family $\mathcal{H}$ is entropy smoothing, then the public-key encryption scheme $\text{PE}$ is secure in the sense of IND-CPA.

**Proof.** We use a sequence-of-games technique described by Shoup [18] to construct our proof. Let an adversary $\mathcal{I}$ participate in Game 0 (see Algorithm 3), which is exactly the standard IND-CPA experiment.

**Algorithm 3** Game 0

1: $sk \xleftarrow{R} G$, $y \leftarrow sk \ast x$, $k \xleftarrow{R} K$
2: $(m_0, m_1, j) \leftarrow \mathcal{I}_1(y, k)$
3: $d \xleftarrow{R} \{0, 1\}$
4: $a \xleftarrow{R} G, u \leftarrow a \ast y, h \leftarrow H_k(u), z \leftarrow a \ast x, c \leftarrow h \oplus m_d$
5: $d' \leftarrow \mathcal{I}_2(j, c, z)$

Algorithm 4 defines Game 1. The only difference from Game 0 is that $u$ is now chosen at random from the orbit $G \ast x$.

**Algorithm 4** Game 1

1: $sk \xleftarrow{R} G$, $y \leftarrow sk \ast x$, $k \xleftarrow{R} K$
2: $(m_0, m_1, j) \leftarrow \mathcal{I}_1(y, k)$
3: $d \xleftarrow{R} \{0, 1\}$
4: $a \xleftarrow{R} G, \leftline{\begin{array}{l} u \xleftarrow{R} G \ast x \end{array}} h \leftarrow H_k(u), z \leftarrow a \ast x, c \leftarrow h \oplus m_d$
5: $d' \leftarrow \mathcal{I}_2(j, c, z)$

Let us define $E_i$ to be the event when $d = d'$ in Game $i$.

We construct a DDHAP distinguisher $S_{01}$ as shown in Algorithm 5. $S_{01}$ outputs 0 when $\mathcal{I}$ guesses correctly, and 1 otherwise. When $S_{01}$ receives a “right” triple $(y, z, u)$ on input, i.e. when $y = a \ast x$, $z = b \ast x$ and $u = ab \ast x$, the probability
Algorithm 5 DDHAP distinguisher $S_{01}$

**Input:** $(y, z, u) \in X^3$

1. $k \xleftarrow{\$} K$
2. $(m_0, m_1, j) \leftarrow \mathcal{I}_1(y, k)$
3. $d \xleftarrow{\$} \{0, 1\}$
4. $h \leftarrow H_k(u)$, $c \leftarrow h \oplus m_d$
5. $d' \leftarrow \mathcal{I}_2(j, c, z)$

**Output:** $d \oplus d'$

distributions observed by $\mathcal{I}$ when run in $S_{01}$ are equivalent to those in Game 0, and we have that

$$\Pr\left[ S_{01}(a \ast x, b \ast x, ab \ast x) = 0 \mid (a, b) \xleftarrow{\$} G^2 \right] = \Pr[\mathcal{E}_0] .$$

Similarly we observe that

$$\Pr\left[ S_{01}(a \ast x, b \ast x, c \ast x) = 0 \mid (a, b, c) \xleftarrow{\$} G^3 \right] = \Pr[\mathcal{E}_1] .$$

As a result,

$$\text{Adv}^\text{DDH}_{S_{01}} = \left| \Pr_{S_{01}}^\text{DDH} - \frac{1}{2} \right| = \left| \frac{\Pr[\mathcal{E}_0] + (1 - \Pr[\mathcal{E}_1])}{2} - \frac{1}{2} \right| = \frac{1}{2} |\Pr[\mathcal{E}_0] - \Pr[\mathcal{E}_1]| .$$

(3)

It may seem that in Game 1 the adversary $\mathcal{I}$ already has no information about the bit $d$. But this is not exactly right. For instance, $\mathcal{I}$ can compute $h_0 \leftarrow c \oplus m_0$ and $h_1 \leftarrow c \oplus m_1$. If there is a way to check that one of the $h_0, h_1$ is not a hash image of an element from $G \ast x$, then $\mathcal{I}$ can conclude on the value of $d$. We see that the ability of the hash function to hide preimage is important, what is expressed by the entropy smoothing property.

We proceed with Game 2, where $h$ is now a random bit string (see Algorithm 6).

Algorithm 6 Game 2

1. $sk \xleftarrow{\$} G$, $y \leftarrow sk \ast x$, $k \xleftarrow{\$} K$
2. $(m_0, m_1, j) \leftarrow \mathcal{I}_1(y, k)$
3. $d \xleftarrow{\$} \{0, 1\}$
4. $a \xleftarrow{\$} G$, $u \xleftarrow{\$} G \ast x$, $h \xleftarrow{\$} \{0, 1\}^w$, $z \leftarrow a \ast x$, $c \leftarrow h \oplus m_d$
5. $d' \leftarrow \mathcal{I}_2(j, c, z)$

Algorithm 7 illustrates an ES distinguisher $S_{12}$.

When a tuple $(k, H_k(u))$ with random $k \in K$ and $u \in G \ast x$ is supplied to $S_{12}$, the adversary $\mathcal{I}$ observes the same probability distributions as in Game 1. On the other hand, when $(k, h)$ is supplied such that $k \in K$ and $h \in \{0, 1\}^w$ are random, we get the setting of Game 2. Hence

$$\text{Adv}^\text{ES}_{S_{12}} = \frac{1}{2} |\Pr[\mathcal{E}_1] - \Pr[\mathcal{E}_2]| .$$

(4)
Algorithm 7 ES distinguisher $S_{12}$

**Input:** $(k, h) \in K \times \{0, 1\}^w$

1. $sk \xleftarrow{R} G$, $y \leftarrow sk \cdot x$
2. $(m_0, m_1, j) \leftarrow I_1(y, k)$
3. $d \xleftarrow{R} \{0, 1\}$
4. $a \xleftarrow{R} G$, $z \leftarrow a \cdot x$, $c \leftarrow h \oplus m_d$
5. $d' \leftarrow I_2(j, c, z)$

**Output:** $d \oplus d'$

Since in Game 2 $h$ is chosen at random, the encryption $c = h \oplus m_d$ is equivalent to the one-time pad. So $I$’s output is independent of the bit $d$, meaning that

$$\Pr[E_2] = \frac{1}{2}.$$  \hspace{1cm} (5)

Getting together (3), (4) and (5) and applying the triangle inequality, we have

$$|\text{Adv}^{\text{IND}}_I| = |\Pr[E_0] - \Pr[E_1] + \Pr[E_1] - \Pr[E_2] + \Pr[E_2] - \frac{1}{2}| \leq \leq 2|\text{Adv}^{\text{DDH}}_{S_{01}}| + 2|\text{Adv}^{\text{ES}}_{S_{12}}| .$$

Note that when $I$ is polynomially bounded, so are the distinguishers $S_{01}$ and $S_{12}$. Hence it follows from the DDHAP assumption and the ES assumption that the advantage $\text{Adv}^{\text{IND}}_I$ is negligible.

6 Conclusive Remarks

As it has been noticed by Koblitz and Menezes \cite{6}, reductionist security arguments should be taken with care. To illustrate, the following considerations can be expressed about the security arguments provided in our paper.

To begin with, the computational problem used in the reductions is somewhat artificial. The DDHA problem is not harder than the corresponding CDHA problem, which in turn is not harder than the corresponding GAI problem. In practice, more research is often devoted to the latter two problems, leaving the former problem unexamined. However only well-studied problems can provide some sort of confidence about their hardness to the scientific community. Without a careful analysis of the DDHA problem complexity, nothing credible can be said about the security of cryptographic schemes based on this problem, and the security of the $KA$ protocol and the $PE$ scheme in particular.

Secondly, the reduction provided in Theorem 1 is not tight. Namely, having an adversary with an advantage $\epsilon$, the advantage of the DDHAP distinguisher constructed during the proof is $\epsilon/l$. Although $l$ cannot be exponentially large, it might still be so big that the attack is not practical for solving the DDHAP instance, i.e. the advantage is too small to effectively obtain the solution. This inconsistency between asymptotic and practical results can damage one’s assurance in security.

It should be also noted that our security argument for the $KA$ protocol only concerns (concurrent sessions of) the $KA$ protocol in isolation. We have not considered the question of composing the $KA$ protocol with other cryptographic schemes, such as authentication protocols or encryption schemes. A trivial example
of such a bad composition could be a weak symmetric cipher with the session key established by the KA protocol. If the cipher allows some kind of attack, then the whole composition is obviously insecure. Cremers has shown that many cryptographic protocols, proven to be correct in isolation, are vulnerable to multi-protocol attacks [19]. To prevent these attacks one may, for example, separate key material between different protocols, or tag protocol messages according to their context.

To summarize, in case when the DDHA problem is hard for a finite abelian group $G$ acting on a set $X$, the result of this paper assures that the key agreement protocol KA based on $G$ and $X$ is secure when used over a channel providing authenticity, and that the public-key encryption scheme PE based on $G$ and $X$ is secure when the adversary does not have access to a decryption oracle and when the used hash function family has good pseudorandom generation capabilities.

References


