Assumptions Related to Discrete Logarithms: Why Subtleties Make a Real Difference

Ahmad-Reza Sadeghi and Michael Steiner

Fachrichtung Informatik, Universität des Saarlandes, D-66123 Saarbrücken, Germany,
{sadeghi,steiner}@cs.uni-sb.de

Abstract. The security of many cryptographic constructions relies on assumptions related to Discrete Logarithms (DL), e.g., the Diffie-Hellman, Square Exponent, Inverse Exponent or Representation Problem assumptions. In the concrete formalizations of these assumptions one has some degrees of freedom offered by parameters such as computational model, the problem type (computational, decisional) or success probability of adversary. However, these parameters and their impact are often not properly considered or are simply overlooked in the existing literature.
In this paper we identify parameters relevant to cryptographic applications and describe a formal framework for defining DL-related assumptions. This enables us to precisely and systematically classify these assumptions.
In particular, we identify a parameter, termed granularity, which describes the underlying probability space in an assumption. Varying granularity we discover the following surprising result: We prove that two DL-related assumptions can be reduced to each other for medium granularity but we also show that they are provably not reducible with generic algorithms for high granularity. Further we show that reductions for medium granularity can achieve much better concrete security than equivalent high-granularity reductions.

Keywords: Complexity Theory, Cryptographic Assumptions, Generic Algorithms, Discrete Logarithms, Diffie-Hellman, Square Exponent, Inverse Exponent.

1 Introduction

Most modern cryptographic algorithms rely on assumptions on the computational difficulty of some particular number-theoretic problem. One well-known class of assumptions is related to the difficulty of computing discrete logarithms in cyclic groups [1]. In this class a number of variants exists. The most prominent ones besides Discrete Logarithm (DL) itself are the computational and decisional Diffie-Hellman (DH) assumptions [2, 3, 4] and their generalization [5, 6]. Less known assumptions are Matching Diffie-Hellman [7, 8], Square Exponent[1](SE) [10, 11], and the Inverse Exponent (IE) [12], an assumption also

[1] This problem is called Squaring Diffie-Hellman in [9]

implicitly required for the security of [13, 14]. Several papers have studied relations among these assumptions, e.g., [15, 16, 17, 18, 9].

In the concrete formalizations of these assumptions one has various degrees of freedom offered by parameters such as computational model, problem type (computational, decisional or matching) or success probability of adversary. However, such aspects are often not precisely considered in the literature and related consequences are simply overlooked. In this paper, we address these aspects by identifying the parameters relevant to cryptographic assumptions. Based on this, we present an understandable formal framework and a notation for defining DL-related assumptions. This enables us to precisely and systematically classify these assumptions.

Among the specified parameters, we focus on a parameter we call *granularity* of the probability space which underlies an assumption. Granularity defines what part of the underlying algebraic structure (i.e., algebraic group and generator) is part of the probability space and what is fixed in advance: For high granularity an assumption has to hold for all groups and generators; for medium granularity the choice of the generator is included in the probability space and for low granularity the probability is taken over both the choice of the group and the generator. Assumptions with lower granularity are weaker than those with higher granularity. Yet not all cryptographic settings can rely on the weaker variants: Only when the choice of the system parameters is guaranteed to be random one can rely on a low-granularity assumption. Consider an anonymous payment system where the bank chooses the system parameters. To base the security of such a system a-priori on a low-granularity assumption would not be appropriate. A cheating bank might try to choose a weak group with trapdoors (easy problem instances) [19] to violate the anonymity of the customer. An average-case low-granular assumption would not rule out that infinitely many weak groups exist even though the number of easy problem instances is asymptotically negligible. However, if we choose the system parameters of the payment system through a random yet verifiable process we can resort to a weaker assumption with lower granularity. Note that to our knowledge no paper on anonymous payment systems does address this issue properly. Granularity was also overlooked in different contexts, e.g., [3] ignores that low-granular assumptions are not known to be random self-reducible which leads to a wrong conclusion.

In this paper we show that varying granularity can lead to surprising results. We extend the results of [9] to the problem class IE, i.e., we prove statements on relations between IE, DH and SE for both computational and decisional variants in the setting of [9] which corresponds to the high-granular case. We then consider medium granularity (with other parameters unchanged) and show the impact: We prove that the decisional IE and SE assumptions are equivalent for medium granularity whereas this is provably not possible for their high-granular variants, at least not in the generic model [15]. We also show that reductions between computational IE, SE and DH can offer much better concrete security for medium granularity than their high-granular analogues.

2 Terminology

2.1 Algebraic Structures

The following terms are related to the algebraic structures underlying an assumption.

Group G**:** All considered assumptions are based on cyclic finite groups. For brevity, however, we will omit the "cyclic finite" in the sequel and refer to them simply as "groups". The order of a group is associated with a security parameter k which classifies the group according to the difficulty of certain problems (e.g., DL).
Group family $\mathcal{G}$**:** A set of groups with the "same" structure/nature. An example is the family of the groups used in DSS [20], i.e., unique subgroups of $\mathbb{Z}_p^*$ of order q with p and q prime, $|q| \approx 2k$ and $p = rq + 1$ for an integer r sufficiently large to make DL hard to compute in security parameter k. Other examples are non-singular elliptic curves or composite groups $\mathbb{Z}_n^*$ with n a product of two safe primes.
Generator g**:** In the DL settings, we also need a generator g which generates the group G, i.e., $\forall y \in G\ \exists x \in \mathbb{Z}_{|G|} :\ y = g^x$.
Structure instance SI**:** The structure underlying the particular problem. In our case this means a group G together with a non-empty tuple of generators g_i. As a convention we abbreviate g_1 to g if there is only a single generator associated with a given structure instance.

2.2 Problem Families

The following two definitions characterize a particular problem underlying an assumption.

Problem family $\mathcal{P}$**:** A family of abstract and supposedly difficult relations. Examples are Discrete Logarithm (DL), Diffie-Hellman (DH), or the Representation Problem (RP). Note that the problem family ignores underlying algebraic groups and how parameters are chosen. Further, note that in the definition of problem families we don't distinguish between decisional or computational variants of a problem.
Problem instance PI**:** A list of concrete parameters fully describing a particular instance of a problem family, i.e., a structure instance SI and a tuple $(priv, publ, sol)$ where $priv$ is the tuple of secret values used to instantiate that problem, $publ$ is the tuple of information publicly known on that problem and sol is the solution of that problem instance. This presentation achieves a certain uniformity of description and allows a generic definition of problem types. For convenience, we define PI^{SI}, $PI^{\mathcal{P}}$, PI^{publ}, PI^{priv} and PI^{sol} to be the projection of a problem instance PI to its structure instance, problem family and public, private and solution part, respectively. When not explicitly stated, we can assume that $priv$ consists always of elements from $\mathbb{Z}_{|G|}$ and $publl$ and sol consists

of elements from G. Furthermore, the structure instance SI is assumed to be publicly known.
If we take the DH problem for integers modulo a prime p as an example, PI_{DH} is defined by the tuple $(((\mathbb{Z}_p^*, p), (g)), ((x, y), (g^x, g^y), (g^{xy})))$ with ${PI_{DH}}^{\mathcal{P}} := DH$, ${PI_{DH}}^{SI}{:=}((\mathbb{Z}_p^*, p), (g))$, ${PI_{DH}}^{priv}{:=}(x, y)$, ${PI_{DH}}^{publ}{:=}(g^x, g^y)$, ${PI_{DH}}^{sol}{:=}g^{xy}$, respectively.

3 Parameters of DL-based Assumptions

In formulating intractability assumptions for problems related to DL we identified the following orthogonal parameters which suffice to describe assumptions relevant to cryptographic applications.[2]

Note that the labels of the following sublists (e.g., "u" and "n" for the first parameter) are used later in Section 4 to identify values corresponding to a given parameter (e.g., "Computational capability of adversary" for above example).

1. **Computational capability of adversary:** Potential algorithms solving a problem have to be computationally limited for number-theoretic assumptions to be meaningful (otherwise we could never assume their nonexistence). Here, we only consider algorithms (called adversary in the following) with running times bounded by a polynomial. The adversary can be of
   u (Uniform complexity): There is a single probabilistic Turing machine (TM) $\mathcal{A}$ which for any given problem instance from the proper domain returns a (not necessarily correct) answer in expected polynomial time in the security parameter k.
   n (Non-uniform complexity): There is an (infinite) family of TMs $\{\mathcal{A}_i\}$ with description size and running time of $\mathcal{A}_i$ bounded by a polynomial in the security parameter k.
   To make the definition of the probability spaces more explicit we model probabilistic TMs always as deterministic machines with the random coins given as explicit input $\mathcal{C}$ chosen from the uniform distribution $\mathcal{U}$.
   Finally, a note on notation: In the case a machine $\mathcal{A}$ has access to some oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$ we denote that as $\mathcal{A}^{\mathcal{O}_1,\ldots,\mathcal{O}_n}$.
2. **"Algebraic knowledge":** A second parameter describing the adversary's computational capabilities relates to the adversary's knowledge on the group family. It can be one of the following:
   $\boldsymbol{\sigma}$ (Generic): This means that the adversary doesn't know anything about the structure (representation) of the underlying algebraic group. More precisely this means that all group elements are represented using a random bijective encoding function $\sigma(\cdot) : \mathbb{Z}_{|G|} \rightarrow G$ and group operations can only be performed via the addition and inversion oracles $\sigma(x+y) \leftarrow \sigma_+(\sigma(x), \sigma(y))$ and $\sigma(-x) \leftarrow \sigma_-(x)$ respectively, which the adversary receives as a black box [15, 22, 23].

[2] For this paper we slightly simplified the classification. Further parameters and values and more details can be found in the full paper [21].

If we use σ in the following we always mean the (not further specified) random encoding used for generic algorithms with a group G and generator g implicitly implied in the context. In particular, by $\mathcal{A}^\sigma$ we refer to a generic algorithm.

(marked by absence of σ) (Specific): In this case the adversary can also exploit special properties (e.g., the encoding) of the underlying group.

3. **Success probability:** The adversary's success probability in solving problem instances (for a given security parameter k and probability distribution $\mathcal{D}$) can either be

   1 (Perfect): The algorithm $\mathcal{A}$ must solve all problem instances from $\mathcal{D}$.

   $1-1/\mathsf{poly}(k)$ (Strong): The algorithm $\mathcal{A}$ must be successful with overwhelming probability, i.e., at most a negligible (in k) amount of instances in $\mathcal{D}$ can remain unsolved.

   ϵ (Invariant): The algorithm $\mathcal{A}$ must answer at least a constant fraction ϵ of the queries from $\mathcal{D}$ successfully.

   $1/\mathsf{poly}(k)$ (Weak): The algorithm $\mathcal{A}$ must be successful with at least a non-negligible amount of queries from $\mathcal{D}$.

   An assumption requiring the inexistence of perfect adversaries corresponds to worst-case complexity, i.e., if the assumption holds then there are at least a few hard instances. However, what is a-priori required in most cases in cryptography is an assumption requiring even the inexistence of weak adversaries, i.e., if the assumption holds then most instances are hard.

4. **"Granularity of probability space":** Depending on what part of the structure instance is a-priori fixed (i.e., the assumption has to hold for all such parameters) or not (i.e., the parameters are part of the probability space underlying an assumption) we distinguish among following situations:

   l (Low-granular): The group family (e.g., prime order subgroups of $\mathbb{Z}_p^*$) is fixed but not the specific structure instance (e.g., parameters p, q and generators g_i for the example group family given above).

   m (Medium-granular): The group (e.g., p and q) but not the generators g_i are fixed.

   h (High-granular): The group as well as the generators g_i are fixed.

5. **Problem family**: Following problem families are useful (and often used) for cryptographic applications. As mentioned in Section 2.2 we describe the problem family (or more precisely their problem instances) by an (abstract) structure instance SI $(G, g_1, g_2, \dots)$ and an (explicit) tuple $(priv, publ, sol)$:

   DL (Discrete Logarithm): $PI_{DL} := (SI, ((x), (g^x), (x)))$.

   DH (Diffie-Hellman): $PI_{DH} := (SI, ((x, y), (g^x, g^y), (g^{xy})))$.

   GDH (Generalized Diffie-Hellman): $PI_{GDH} := (SI, ((x_i | 1 \le i \le n \wedge n \ge 2), (g^{\prod_{i \in I} x_i} | \forall\, I \subset \{1, \dots, n\}),\ (g^{\prod_{i=1}^{n} x_i})))$.

   SE (Square-Exponent): $PI_{SE} := (SI, ((x), (g^x), (g^{x^2})))$.

   IE (Inverse-Exponent): $PI_{IE} := (SI, ((x), (g^x), (g^{x^{-1}})))$.
   Note that $priv(x)$ has to be an element of $\mathbb{Z}_{|G|}^*$ here, contrary to the other problem families mentioned where $priv$ contains elements of $\mathbb{Z}_{|G|}$.

   RP (Representation Problem): $PI_{RP} := (SI,\ ((x_i \mid 1 \le i \le n\ \wedge\ n \ge 2),\ (\prod_{i=1}^{n} g_i^{x_i}),\ (x_i \mid 1 \le i \le n)))$.

6. **Problem type**: Each problem can be formulated in three variants.

   C (Computational): For a given problem instance PI the algorithm $\mathcal{A}$ succeeds if and only if it can solve PI, i.e., $\mathcal{A}(PI^{publ}, \dots) = PI^{sol}$.

   D (Decisional): For a given problem instance PI, a random problem instance $PI_{\mathcal{R}}$ and a random bit b the algorithm $\mathcal{A}$ succeeds if and only if it can decide whether a given solution matches the given problem instance, i.e., $\mathcal{A}(PI^{publ}, b * PI^{sol} + \bar{b} * {PI_{\mathcal{R}}}^{sol}), \dots) = b$.

   M (Matching): For two given problem instances PI_0 and PI_1, and a random bit b the algorithm $\mathcal{A}$ succeeds if and only if it can correctly associate the solutions to their corresponding problem instances, i.e., $\mathcal{A}({PI_0}^{publ}, {PI_1}^{publ}, {PI_b}^{sol}, {PI_{\bar{b}}}^{sol}, \dots) = b$.
7. **Group family**: We distinguish between group families with the following generic properties. The factorization of the group order contains

   lprim large prime factors (at least one)

   nsprim no small prime factor

   prim only a single and large prime factor

4 Defining Assumptions

Using the parameters and corresponding values defined in the previous section we can define intractability assumptions in a compact and precise way. The used notation is composed out of the labels corresponding to the parameter values of a given assumption. This is best illustrated in following example:[3] The term

$$1/\mathsf{poly}(k)\text{-DDH}^{\sigma}(\text{c:u; g:h; f:prim})$$

denotes the decisional (D) Diffie-Hellman (DH) assumption in prime-order groups (f:prim) with weak success probability ($1/\mathsf{poly}(k)$), limited to generic algorithms (σ) of uniform complexity (c:u), and with high granularity (g:h).

The formal assumption statement automatically follows from the parameter values implied by an assumption term. For space reasons we restrict ourselves again to an example as explanation: To assume that above-mentioned assumption $1/\mathsf{poly}(k)\text{-DDH}^{\sigma}(\text{c:u; g:h; f:prim})$ holds informally means that there are no generic algorithms of uniform complexity which are asymptotically able to distinguish a non-negligible amount of DH tuples from random ones in prime-order subgroups where the probability space is defined according to high granularity. Formally this assumption is given below. To give the reader a better feel for the newly introduced parameter granularity we specify also the corresponding assumptions with medium and low granularity.

A few explanations to the statements: S_G, S_g and S_{PI} are the probabilistic algorithms selecting groups, generators and problem instances, respectively; ExpectRunTime gives a bound on the expected run time of the algorithm and $\mathbf{Prob}[\mathcal{S} :: \mathcal{PS}]$ gives the probability of statement $\mathcal{S}$ with the probability taken over a probability space defined by $\mathcal{PS}$. Furthermore, remember that PI_{DH} is $(SI, ((x, y), (g^x, g^y), (g^{xy})))$, ${PI_{DH}}^{publ}$ is (g^x, g^y) and ${PI_{DH}}^{sol}$ is (g^{xy}).

[3] A more thorough treatment is omitted here due to space reasons and will appear in [21].

1. Assumption $1/\mathsf{poly}(k)$-DDH$^\sigma$(c:u; g:h; f:prim), i.e., with high granularity:

   $\forall p_1, p_2 > 0;\ \forall \mathcal{A}^\sigma \in \text{TM};\ \exists k_0;\ \forall k > k_0;$
   $\forall G \leftarrow S_G(\text{“prime-order groups”}, 1^k);\ \forall g \leftarrow S_g(G);\ \mathit{SI} \leftarrow (G, g);$
   $\text{ExpectRunTime}(\mathcal{A}^\sigma(\mathcal{C}, G, g, \mathit{PI}_{DH})) < k^{p_2};$
   $(|\,\mathbf{Prob}[\mathcal{A}^\sigma(\mathcal{C}, G, g, {\mathit{PI}_{DH}}^{publ}, b * {\mathit{PI}_{DH}}^{sol} + \bar{b} * {\mathit{PI}_{\mathcal{R}}}^{sol}) = b ::$
   $\quad b \overset{\mathcal{R}}{\leftarrow} \{0,1\};\ \mathcal{C} \overset{\mathcal{R}}{\leftarrow} \mathcal{U};$
   $\quad \mathit{PI}_{DH} \leftarrow S_{PI}(DH, \mathit{SI});\ \mathit{PI}_{\mathcal{R}} \leftarrow S_{PI}({\mathit{PI}_{DH}}^{\mathcal{P}}, {\mathit{PI}_{DH}}^{SI});$
   $\ ]\text{- } 1/2\,| \cdot 2)\ < 1/k^{p_1}$
2. As above except now with medium granularity ($1/\mathsf{poly}(k)$-DDH$^\sigma$(c:u; g:m; f:prim)):

   $\forall p_1, p_2 > 0;\ \forall \mathcal{A}^\sigma \in \text{TM};\ \exists k_0;\ \forall k > k_0;$
   $\forall G \leftarrow S_G(\text{“prime-order groups”}, 1^k);$
   $\text{ExpectRunTime}(\mathcal{A}^\sigma(\mathcal{C}, G, g, \mathit{PI}_{DH})) < k^{p_2};$
   $(|\,\mathbf{Prob}[\mathcal{A}^\sigma(\mathcal{C}, G, g, {\mathit{PI}_{DH}}^{publ}, b * {\mathit{PI}_{DH}}^{sol} + \bar{b} * {\mathit{PI}_{\mathcal{R}}}^{sol}) = b ::$
   $\quad b \overset{\mathcal{R}}{\leftarrow} \{0,1\};\ \mathcal{C} \overset{\mathcal{R}}{\leftarrow} \mathcal{U};$
   $\quad g \leftarrow S_g(G);\ \mathit{SI} \leftarrow (G, g);$
   $\quad \mathit{PI}_{DH} \leftarrow S_{PI}(DH, \mathit{SI});\ \mathit{PI}_{\mathcal{R}} \leftarrow S_{PI}({\mathit{PI}_{DH}}^{\mathcal{P}}, {\mathit{PI}_{DH}}^{SI});$
   $\ ]\text{- } 1/2\,| \cdot 2)\ < 1/k^{p_1}$
3. As above except now with low granularity ($1/\mathsf{poly}(k)$-DDH$^\sigma$(c:u; g:l; f:prim)):

   $\forall p_1, p_2 > 0;\ \forall \mathcal{A}^\sigma \in \text{TM};\ \exists k_0;\ \forall k > k_0;$
   $\text{ExpectRunTime}(\mathcal{A}^\sigma(\mathcal{C}, G, g, \mathit{PI}_{DH})) < k^{p_2};$
   $(|\,\mathbf{Prob}[\mathcal{A}^\sigma(\mathcal{C}, G, g, {\mathit{PI}_{DH}}^{publ}, b * {\mathit{PI}_{DH}}^{sol} + \bar{b} * {\mathit{PI}_{\mathcal{R}}}^{sol}) = b ::$
   $\quad b \overset{\mathcal{R}}{\leftarrow} \{0,1\};\ \mathcal{C} \overset{\mathcal{R}}{\leftarrow} \mathcal{U};$
   $\quad G \leftarrow S_G(\text{“prime-order groups”}, 1^k);\ g \leftarrow S_g(G);\ \mathit{SI} \leftarrow (G, g);$
   $\quad \mathit{PI}_{DH} \leftarrow S_{PI}(DH, \mathit{SI});\ \mathit{PI}_{\mathcal{R}} \leftarrow S_{PI}({\mathit{PI}_{DH}}^{\mathcal{P}}, {\mathit{PI}_{DH}}^{SI});$
   $\ ]\text{- } 1/2\,| \cdot 2)\ < 1/k^{p_1}$

To express relations among assumptions we will use following notation:

$A \implies B$ means that if assumption A holds, so does assumption B, i.e., A (B) is a weaker (stronger) assumption than B (A). Vice-versa, it also means that if there is a (polynomially-bounded) algorithm $\mathcal{A}_B$ breaking assumption B then we can build another (polynomially-bounded) algorithm $\mathcal{A}_A^{\mathcal{A}_B}$ with (oracle) access to $\mathcal{A}_B$ which breaks assumption A.

$A \iff B$ means that $A \implies B$ and $B \implies A$, i.e., A and B are assumptions of the same (polynomial) complexity.

Furthermore, if we are referring to oracle-assumption, i.e., assumptions where we give adversaries access to auxiliary oracles, we indicate it by listing the oracles at the end of the list in the assumption term. For example, the assumption $1/\mathsf{poly}(k)$-DDH$^\sigma$(c:u; g:h; f:prim; $\mathcal{O}_{\text{1-DSE(c:u; g:h; f:prim)}}$) corresponds to the first assumption statement given above except that now the adversary also gets access to an oracle breaking the assumption 1-DSE(c:u; g:h; f:prim). Finally, if we use $*$ for a particular parameter in an assumption term we mean the class of assumptions where this parameter is varied over all possible values.

5 The Impact of Granularity

It would go beyond the scope (and space) of this paper to discuss all previously identified parameters and we will focus only on granularity. Before stating the actual results, let us first briefly repeat the practical relevance of granularity as alluded in the introduction. Assumptions with lower granularity are weaker and are so more desirable in principle. However, which of the granularity variants is appropriate in cryptographic protocols depends on how and by whom the parameters are chosen. A priori we have to use a high-granular assumption. Yet in following situations we can resort to a weaker less granular assumption: The security requirements of the cryptographic system guarantee that it's in the best (and only) interest of the chooser of the system parameters to choose them properly; the system parameters are chosen by a mutually trusted third party; or the system parameters are chosen in a verifiable random process.[4] Furthermore, care has to be taken for DL-related high and medium granular assumptions in $\mathbb{Z}_p^*$ and its subgroups. Unless we further constrain the set of valid groups with (expensive) tests as outlined by [19], we require, for a given security parameter, considerably larger groups than for the low granular counterpart of the assumptions.

6 Computational DH, SE and IE

Maurer and Wolf [10] prove the equivalence between the computational SE and DH assumption in their uniform and high-granular variant for both perfect and invariant success probabilities.

We briefly review their results, extend their results to medium granularity and prove similar relations between IE and DH.

6.1 CSE versus CDH

Theorem 1 **([10]).**
ϵ-CSE(c:u; g:h; f:$*$) $\iff$ ϵ-CDH(c:u; g:h; f:$*$) □

More concretely, they show the following: Let $0 < \alpha_1 < 1$, $0 < \alpha_2 < 1$ be arbitrary constants and let G be a cyclic group with known order $|G|$. Then

(a) given an oracle $\mathcal{O}_{CDH}$ which breaks ϵ-CDH(c:u; g:h; f:$*$) in G with success probability at least $\epsilon = \alpha_1$, there exists an algorithm $\mathcal{A}^{\mathcal{O}_{CDH}}$ that breaks ϵ-CSE(c:u; g:h; f:$*$) in G with success probability at least $\epsilon = \alpha_1$.

(b) given an oracle $\mathcal{O}_{CSE}$ which breaks ϵ-CSE(c:u; g:h; f:$*$) in G with success probability at least $\epsilon = \alpha_2$, there exists an algorithm $\mathcal{A}^{\mathcal{O}_{CSE}}$ that breaks ϵ-CDH(c:u; g:h; f:$*$) in G with success probability at least $\epsilon = {\alpha_2}^3$.

From these reductions the theorem immediately follows.

[4] This can be done either through a joint generation using random coins [24] or using heuristics such as the one used for DSS key generation [20].

Remark 1. Maurer and Wolf showed the reduction for invariant success probability. However, the results easily extend also to all other variants related to success probability, i.e., weak, strong and perfect. ○

Above relation also holds for medium granularity as we show next.

Theorem 2.
$1/\mathsf{poly}(k)$-CSE(c:u; g:m; f:∗) $\iff$ $1/\mathsf{poly}(k)$-CDH(c:u; g:m; f:∗). □

Proof (sketch). The proof idea of Theorem 1 can also be applied in this case. The only thing we have to show is that the necessary randomization in the reduction steps can be extended to the medium granularity variants of CDH and CSE. This can be done using standard techniques and is shown in the full version of this paper [21]. The rest of the proof remains then the same as the proof of Theorem 1. ■

Remark 2. Reduction proofs of a certain granularity can in general be easily applied to the lower granularity variant of the involved assumptions. The necessary condition is only that all involved randomizations extend to the wider probability space associated with the lower granularity parameter.

Remark 3. In all the mentioned problem families the necessary random self-reducibility exists for medium granularity and above remark always holds, i.e., we can transform proofs from a high-granular variant to the corresponding medium-granular variant. However, it does not seem to extend to low-granular variants. This would require to randomize not only over the public part of the problem instance PI and the generator g but also over the groups G with the same associated security parameter k; this seems impossible to do in the general case and is easily overlooked and can lead to wrong conclusions, e.g., the random self-reducibility as stated in [3] doesn't hold as the assumptions are (implicitly) given in their low-granular form. ○

6.2 CDH versus CIE

In the following we prove that similar relations as above also exist for CIE. In the high-granular case following theorem holds. Here as well as in following results related to IE we will restrict ourselves to groups of prime order. The results also extend to more general groups but the general case is more involved[5] and omitted in this version of the paper for space reasons.

Theorem 3.
$1/\mathsf{poly}(k)$-CDH(c:u; g:h; f:prim) $\iff$ $1/\mathsf{poly}(k)$-CIE(c:u; g:h; f:prim) □

More concretely, we show the following: Let G be a cyclic group with known prime order. Then

[5] Due to the difference in input domains between IE and other assumptions we have to deal with the distribution of $\mathbb{Z}^*_{|G|}$ over $\mathbb{Z}_{|G|}$. This results, e.g., in the success probability being reduced by a factor of $\varphi(|G|)/|G|$.

(a) given an oracle $\mathcal{O}_{CDH}$ which breaks $*$-CDH(c:u; g:h; f:prim) in G with success probability at least $* = \alpha_1(k)$, there exists an algorithm $\mathcal{A}^{\mathcal{O}_{CDH}}$ that solves CIE(c:u; g:h; f:prim) in G with success probability at least $\alpha_1(k)^{O(\log|G|)}$.[6]

(b) given an oracle $\mathcal{O}_{CIE}$ which breaks $*$-CIE(c:u; g:h; f:prim) in G with success probability at least $* = \alpha_2(k)$, there exists an algorithm $\mathcal{A}^{\mathcal{O}_{CIE}}$ that solves CSE(c:u; g:h; f:prim) in G with success probability at least $\alpha_2(k)^3$.

(c) following (b), there exists also an algorithm $\mathcal{A}^{\mathcal{O}_{CIE}}$ that, with success probability at least $\alpha_2(k)^9$, breaks $1/\mathsf{poly}(k)$-CDH(c:u; g:h; f:prim) in G.

From these reductions and Remark 4 the theorem immediately follows. The complete proof can be found in [21].

Remark 4. For strong and perfect success probabilities, i.e., $\alpha_1(k)$ is either 1 or 1–$1/\mathsf{poly}(k)$, the resulting success probability in case (a) can always be polynomially bounded because $O(\log|G|) = O(\mathsf{poly}(k))$ and there always exist constants c and c' such that for large enough k it holds that $(1 - 1/k^{c'})^{O(\mathsf{poly}(k))} \geq 1/k^c$. However, for the weak and invariant success probability, i.e., $\alpha_1(k)$ is either ϵ or $1/\mathsf{poly}(k)$, the resulting error cannot be bounded polynomially. This implies that above reduction in (a) does not work directly in the case where the oracle $\mathcal{O}_{CDH}$ is only of the weak or invariant success probability flavor! The success probability of $\mathcal{O}_{CDH}$ has first to be improved by self-correction [15] to strong success probability, a task expensive both in terms of oracle calls and group operations. ○

Next, we prove above equivalence also for medium granularity. Similar to Theorem 2 we could argue that due to the existence of a randomization the result immediately follows also for the medium granularity case. However, we will show below that the reduction can be performed much more efficiently in the medium granular case than in the case above; thereby we improve the concrete security considerably.

Theorem 4.

$$1/\mathsf{poly}(k)\text{-CSE(c:u; g:m; f:prim)} \iff 1/\mathsf{poly}(k)\text{-CIE(c:u; g:m; f:prim)}. \qquad \square$$

Proof. "⇒": We construct $\mathcal{A}^{\mathcal{O}_{CIE}}$ as follows: Assume we are given a CSE instance g^x with respect to generator g. We set $h := g^x$ and $t := x^{-1}$, pass $g^x(= h)$ and $g(= h^t)$ to $\mathcal{O}_{CIE}$. Assuming the oracle answered correctly we get the desired solution to the CSE problem: $h^{t^{-1}} = (g^x)^{(x^{-1})^{-1}} = g^{x^2}$.
"⇐": Conversely we can exploit the identity $((g^x)^{(x^{-1})^2} = (g^x)^{x^{-2}} = g^{xx^{-2}} = g^{x^{-1}}$ to construct $\mathcal{A}^{\mathcal{O}_{CSE}}$ solving CIE with a single call to $\mathcal{O}_{CSE}$. ■

Remark 5. For each direction we need now only a single oracle call. If we take also into account that with a single oracle call $1/\mathsf{poly}(k)$-CSE(c:u; g:m; f:prim) can be reduced to $1/\mathsf{poly}(k)$-CDH(c:u; g:m; f:prim) we achieve a reduction from $1/\mathsf{poly}(k)$-CIE(c:u; g:m; f:prim) to $1/\mathsf{poly}(k)$-CDH(c:u; g:m; f:prim) while retaining the success probability of the oracle.

[6] The exponent $O(\log|G|)$ stems from a square and multiply used in the reduction.

Remark 6. Above observation also implies that, contrary to the high-granular (Remark 4) case, this reduction directly applies to the invariant and weak success probability variant of the assumptions, i.e., no self-correction is required. ○

In particular the Remark 5 is of high significance. The reduction we get in the medium-granular case is much more efficient than the corresponding reduction in the high-granular case: With a single instead of $\log(|G|)$ (very expensive) oracle calls and $O(\log(|G|))$ instead of $O(\log(|G|)^2)$ group operations we achieve a success probability which is higher by a power of $O(\log(|G|))$!

7 Decisional DH, SE and IE

7.1 Difficulty in the Generic Model

We state first a Lemma which plays an important role for later proofs in the context of generic algorithms:

Lemma 1 ([25, 15]). *Let $P(X_1, X_2, \cdots, X_n)$ be a non-zero polynomial in $\mathbb{Z}_{p^e}[X]$ of total degree $d \geq 0$ $(p \in \mathbb{P}; e \in \mathbb{N})$. Then the probability that $P(x_1, x_2, \cdots, x_n) \equiv 0$ is at most d/p for a random tuple $(x_1, x_2, \cdots, x_n) \in_{\mathcal{R}} \mathbb{Z}^n_{p^e}$.* □

Using Lemma 1, Wolf [9] shows that there exists no generic algorithm that can solve DSE (and consequently also DDH) in polynomial time if the order of the multiplicative group is not divisible by small primes, as summarized in following theorem:

Theorem 5 ([9]).
true $\implies$ $1/\mathsf{poly}(k)$-$\text{DSE}^{\sigma}(\text{c:u; g:h; f:nsprim})$ □

Remark 7. More precisely, Wolf shows that the probability that any $\mathcal{A}^{\sigma}$ can correctly distinguish correct DSE inputs from incorrect ones is at most $\frac{(T+4)(T+3)}{2p'}$ where p' is the smallest prime factor of $|G|$ and T is an upper bound on the algorithm's runtime.

Remark 8. It might look surprising that $1/\mathsf{poly}(k)$-$\text{DSE}^{\sigma}(\text{c:u; g:h; f:nsprim})$ always holds, i.e., it's a fact, not an assumption. Of course, the crucial aspect is the rather restricted adversary model (the σ in the assumption statement) which limits adversaries to generic algorithms. However, note that this fact means that to break DSE one has to exploit deeper knowledge on the actual structure of the used algebraic groups. In particular, for appropriately chosen prime-order subgroups of $\mathbb{Z}^*_p$ and elliptic or hyper-elliptic curves no such exploitable knowledge could yet be found and all of currently known efficient and relevant algorithms in these groups are generic algorithms, e.g., Pohlig-Hellman [26] or Pollard-ρ [27]. Nevertheless, care has to be applied when proving systems secure in the generic model [28]. ○

In the following theorem we show that also DIE cannot be solved by generic algorithms if the order of the multiplicative group is not divisible by small primes.

Theorem 6.

$$\textit{true} \implies 1/\mathsf{poly}(k)\text{-DIE}^{\sigma}(\text{c:u; g:h; f:nsprim}) \qquad \square$$

The proof is similar to the proof of Theorem 5 and can be found in [21].

7.2 DSE versus DDH

Wolf [9] shows following two results on the relation of DSE and DDH: DSE can easily be reduced to DDH but the converse doesn't hold; in fact, Theorem 8 shows that a DSE oracle, even when perfect, is of no help in breaking DDH assumptions.

Theorem 7 ([9]).

$$1/\mathsf{poly}(k)\text{-DSE}(\text{c:u; g:h; f:}*) \implies 1/\mathsf{poly}(k)\text{-DDH}(\text{c:u; g:h; f:}*) \qquad \square$$

Remark 9. Following Remark 2, this result easily extends also to the medium-granular variant. ○

Theorem 8 ([9]).

$$\textit{true} \implies 1/\mathsf{poly}(k)\text{-DDH}^{\sigma}(\text{c:u; g:h; f:nsprim}; \mathcal{O}_{1\text{-DSE}(\text{c:u; g:h; f:nsprim})}) \qquad \square$$

Remark 10. More precisely, Wolf shows that the probability that any $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ can correctly distinguish correct DDH inputs from incorrect ones is at most $\frac{(T+5)(T+4)}{2p'}$ where p' is the smallest prime factor of $|G|$ and T is an upper bound on the algorithm's runtime. ○

7.3 DIE versus DDH

In the following we prove that similar relations also hold among DDH and DIE: We show a reduction from DIE to DDH and prove that a DIE oracle, even when perfect, is of no help in breaking DDH assumptions.

Theorem 9.

$$1/\mathsf{poly}(k)\text{-DIE}(\text{c:u; g:h; f:prim}) \implies 1/\mathsf{poly}(k)\text{-DDH}(\text{c:u; g:h; f:prim}) \qquad \square$$

Theorem 10.

$$\textit{true} \implies 1/\mathsf{poly}(k)\text{-DDH}^{\sigma}(\text{c:u; g:h; f:nsprim}; \mathcal{O}_{1\text{-DIE}(\text{c:u; g:h; f:nsprim})}) \qquad \square$$

Both proofs follow similar strategies than the proofs of Theorem 7 and 8 and can be found in [21]. One twist is that the input domains between IE and DH are different and the DIE-oracle cannot answer correctly to the queries not from its domain. However, since this limits the use of a DIE-oracle in solving DDH even further, this does not affect the desired result.

7.4 DSE versus DIE

In the next theorem we prove that an oracle breaking 1-DSE(c:u; g:h; f:$*$) is of no help in breaking $1/\mathsf{poly}(k)$-DIE$^\sigma$(c:u; g:h; f:$*$).

Theorem 11.
true $\Longrightarrow$ $1/\mathsf{poly}(k)$-DIE$^\sigma$(c:u; g:h; f:nsprim; $\mathcal{O}_{\text{1-DSE(c:u; g:h; f:nsprim)}}$) □

Proof. Similar to the proofs of Theorem 6 and 10 we define a Lemma which associates the minimal generic complexity of solving DIE directly to the smallest prime factor of the order of the underlying group G. Theorem 11 immediately follows from Lemma 2 and Remark 11. ■

Remark 11. In the classical formulation of decision problems the adversary gets, depending on the challenge b, either the correct element or a random element as input, i.e., in the case of DIE the adversary gets g^x together with $g^{x^{-1}}$ if $b = 0$ and g^c if $b = 1$. The formulation used in the Lemma considers a slightly different variant of the decisional problem type: We consider here an adversary which receives, in random order, both the correct and a random element and the adversary has to decide on the order of the elements, i.e., the adversary gets g^x and $(g^{x^{-1}}, g^c)$ for $b = 0$ and $(g^c, g^{x^{-1}})$ for $b = 1$.

This formulation makes the proofs easier to understand. However, note that both variants can be shown equivalent. ○

Lemma 2. *Let G be a cyclic group and g a corresponding generator, let p' be the smallest prime factor of $n = |G|$. Let $\mathcal{O}_{DSE}$ be a given oracle solving DSE tuples in G and let $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ be any generic algorithm for groups G with maximum run time T and oracle access to $\mathcal{O}_{DSE}$. Further let x_0, x_1 be random elements of $\mathbb{Z}^*_{|G|}$, $b \in_{\mathcal{R}} \{0,1\}$ a randomly and uniformly chosen bit and $\mathcal{C} \overset{\mathcal{R}}{\leftarrow} \mathcal{U}$. Then it always holds that*

$$\boldsymbol{Prob}[\mathcal{A}^\sigma(\mathcal{C}, (G,g), g^{x_0}, g^{{x_b}^{-1}}, g^{{x_{\bar{b}}}^{-1}}) = b] \le \frac{(T+4)(T+3)}{2p'}$$ □

Proof. For given $\sigma(1)$, $\sigma(x)$, $\{\sigma(x^{-1}), \sigma(c)\}$, assume that the algorithm $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ computes at most $T_1 + 4$ (images of) distinct linear combinations P_i of the elements $1, x, x^{-1}$, c with $P_i(1, x, x^{-1}, c) = a_{i1} + a_{i2}x + a_{i3}x^{-1} + a_{i4}c$ such that

$$\sigma(P_i(1, x, x^{-1}, c)) = \sigma(a_{i1} + a_{i2}x + a_{i3}x^{-1} + a_{i4}c),$$

where a_{ij} are constant coefficients. Furthermore, it is not a-priori known to $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ which of the (known) values in $\{a_{i3}, a_{i4}\}$ is the coefficient for x^{-1} and which one corresponds to c. Assume that $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ makes T_2 calls to $\mathcal{O}_{DSE}$. $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ may be able to distinguish the coefficient by obtaining information from either of the following events:

E_a: $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ finds a collision between two distinct linear equations (P_i, P_j) with $i \neq j$, i.e.,

$$\sigma(P_i(1, x, x^{-1}, c)) = \sigma(P_j(1, x, x^{-1}, c)) \Rightarrow P_i(1, x, x^{-1}, c) = P_j(1, x, x^{-1}, c)$$

E_b: $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ gets at least one positive answer from $\mathcal{O}_{DSE}$ for a non-trivial query with $i \neq j$, i.e.,

$$\sigma(P_i(1, x, x^{-1}, c)) = \sigma((P_j(1, x, x^{-1}, c)^2).$$

Let E be the union of the events E_a and E_b. Now we compute an upper bound for the probability that either of these events occurs.

Case E_a: Consider P_i and P_j as polynomials. There are $\binom{T_1+4}{2} = \frac{(T_1+4)(T_1+3)}{2}$ possible distinct pairs. The probability of a collision for two linear combinations P_i, P_j is the probability of (randomly) finding the root of polynomial $x(P_i - P_j) \equiv 0 \bmod p^e$ for any prime factor p of $|G|$ with $p^e||G|$. Due to Lemma 1 this probability is at most $2/p^e$ $(\leq 2/p')$, because the degree of $x(P_i - P_j)$ is at most two.[7] It follows that $\mathbf{Prob}[E_a] \leq \frac{(T_1+4)(T_1+3)}{2}\frac{2}{p'} = \frac{(T_1+4)(T_1+3)}{p'}$.
Case E_b: For $i \neq j$ it is not possible to derive a relation $P_i = {P_j}^2$ except that P_i and P_j are both constant polynomials ($\neq 0$), meaning that the polynomial $x^2(P_i - {P_j}^2) \equiv 0 \bmod p^e$ for $x \neq 0$. The total degree of the polynomial $x^2(P_i - {P_j}^2)$ is at most 4 and the probability for E_b is at most $\frac{4}{p'}T_2$.

In total we have

$$\mathbf{Prob}[E] \leq \mathbf{Prob}[E_a] + \mathbf{Prob}[E_b] = \frac{(T_1+4)(T_1+3)}{p'} + T_2\frac{4}{p'} \leq \frac{(T+4)(T+3)}{p'},$$

with $T_1 + T_2 \leq T$. The success probability of $\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}$ therefore is:

$$\mathbf{Prob}[\mathcal{A}^{\sigma,\mathcal{O}_{DSE}}(..) = b] = \mathbf{Prob}[E] + \frac{1}{2}\mathbf{Prob}[\bar{E}] =$$

$$\mathbf{Prob}[E] + \frac{1 - \mathbf{Prob}[E]}{2} = \frac{1}{2} + \frac{\mathbf{Prob}[E]}{2} \leq \frac{1}{2} + \frac{(T+4)(T+3)}{2p'}.$$

■

In sharp contrast to the above mentioned high granular case, we prove in the following theorem that these assumptions are equivalent for their medium granular version (other parameters remain unchanged).

Theorem 12.
$1/\mathsf{poly}(k)$-DSE(c:u; g:m; f:prim) $\iff$ $1/\mathsf{poly}(k)$-DIE(c:u; g:m; f:prim). □

Proof. "$\Leftarrow$": Assume we are given a DIE tuple $I_{DIE} = (g, g^x, g^z)$ where g^z is either $g^{x^{-1}}$ or a random element of group G. Set $h{:=}g^z$ then $g = h^t$ and $g^x = h^{tx}$ for some (unknown) $t \in \mathbb{Z}^*_{|G|}$. After reordering the components we obtain the tuple (h, h^t, h^{xt}).

[7] Note that P_i, P_j are also functions of $x^{-1}, x \neq 0$ and thus one can virtually think of the polynomial $x(P_i - P_j)$ by multiplying both sides of the equation $P_i = P_j$ with x. Furthermore, uniformly distributed random values mod$n$ are also randomly and uniformly distributed modp^e.

If $z = x^{-1}$ then $t = x$ and the tuple (h, h^t, h^{xt}) will have the form (h, h^t, h^{t^2}) which represents a DSE tuple and can be solved by the given DSE oracle. The probability distribution is correct, since h is a group generator and h^t is a random element of G.
If $z \neq x^{-1}$ then $t \neq x$ and h^{xt} is a random group element (x is a random element of $\mathbb{Z}^*_{|G|}$) and the elements h, h^t, h^{tx} are independent.

"$\Rightarrow$": Assume, we are given a DSE tuple (g, g^x, g^z) where g^z is either g^{x^2} or a random group element. Set $h := g^x$ then $g = h^t$ and $g^z = h^{tz}$ for some (unknown) $t \in \mathbb{Z}^*_{|G|}$. After reordering the components we obtain the tuple (h, h^t, h^{tz}).
If $z = x^2$ then we have[8] $x = t^{-1}$ and $z = t^{-2}$ meaning that the tuple (h, h^t, h^{xt}) has the form $(h, h^t, h^{t^{-1}})$ representing a DIE tuple. Its probability distribution is correct because h is a group generator, h^t is a random element of G and the last element $h^{t^{-1}}$ has the correct form.
If $z \neq x^2$ then h^{zt} is a random group element, since t is a random element of $\mathbb{Z}^*_{|G|}$, and further the elements h, h^t and h^{tz} are independent. ■

8 Conclusions

In this paper, we identify the parameters relevant to cryptographic assumptions. Based on this we present a framework and notation for defining assumptions related to Discrete Logarithms. Using this framework these assumptions can be precisely and systematically classified. Wider adoption of such a terminology would ease the study and comparison of results in the literature, e.g., the danger of ambiguity and mistakes in lengthly stated textual assumptions and theorems would be minimized. Furthermore, clearly stating and considering these parameters opens an avenue to generalize results regarding the relation of different assumptions and to get a better understanding of them. This is the focus of our ongoing research and is covered to a larger extent in the full version of the paper [21].

A parameter in defining assumptions previously ignored in the literature is granularity. We show (as summarized in Figure 1) that varying this parameter leads to surprising results: we prove that some DL-related assumptions are equivalent in one case (medium granular) and provably not equivalent, at least not in a generic sense, in another case (high granular). Furthermore, we also show that some reductions for medium granularity are much more efficient than their high-granular version leading to considerably improved concrete security, in particular as medium granularity results in weaker assumptions than high-granular ones. However, we note that medium- or low-granular assumption apply in cryptographic settings only when the choice of system parameters is guaranteed to be truly random.

In this paper we only scratched the topic of granularity and interesting open questions remain to be answered: While for both CDL and CDH it can be shown

[8] This is because $h^x = g^{x^2} = h^{tz}$ which implies $h^x = h^{tx^2}$, $x = tx^2$ and $t = x^{-1}$.

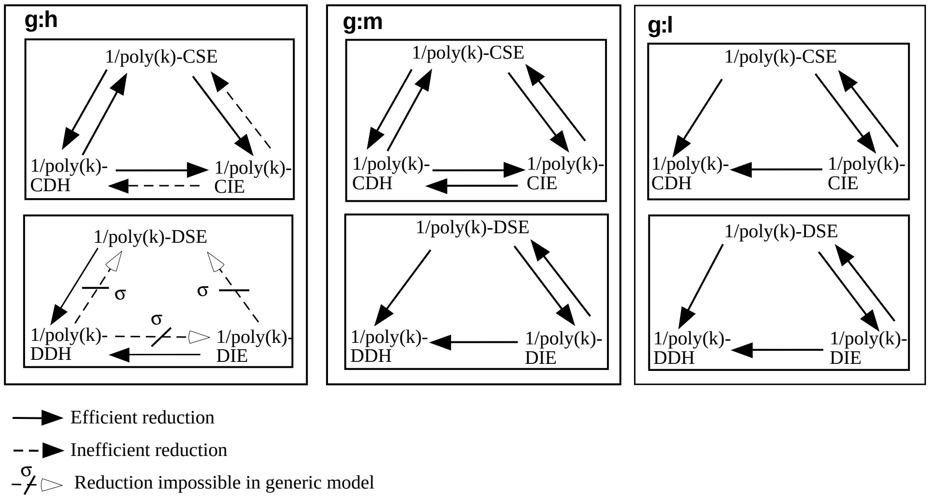

Fig. 1. Summary of our results

that their high- and medium-granular assumptions are equivalent, this is not yet known for DDH (also briefly mentioned as an open problem in [29]). Only few relations can be shown for low-granular assumption as no random self-reducibility is yet known. However, achieving such "full" random self-reducibility seems very difficult in general (if not impossible) in number-theoretic settings [30] contrary to, e.g., lattice settings used in [31].

Acknowledgements

We thank Birgit Pfitzmann, Matthias Schunter, and the anonymous reviewers for their helpful comments.

References

[1] Kevin S. McCurley. The discrete logarithm problem. In Carl Pomerance, editor, *Cryptology and Computational Number Theory*, volume 42 of *Proceedings of Symposia in Applied Mathematics*, pages 49–74, Providence, 1990. American Mathematical Society. 101

[2] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, November 1976. 101

[3] Dan Boneh. The Decision Diffie-Hellman problem. In *Third Algorithmic Number Theory Symposium*, number 1423 in Lecture Notes in Computer Science, pages 48–63. Springer-Verlag, Berlin Germany, 1998. 101, 102, 109

[4] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, number 1462 in Lecture Notes in Computer Science, pages 13–25. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1998. 101

[5] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th Symposium on Foundations of Computer Science (FOCS)*, pages 458–467. IEEE Computer Society Press, 1997. 101

[6] Michael Steiner, Gene Tsudik, and Michael Waidner. Key agreement in dynamic peer groups. *IEEE Transactions on Parallel and Distributed Systems*, 11(8):769–780, August 2000. 101

[7] Yair Frankel, Yiannis Tsiounis, and Moti Yung. "Indirect discourse proofs": Achieving fair off-line cash (FOLC). In K. Kim and T. Matsumoto, editors, *Advances in Cryptology – ASIACRYPT '96*, number 1163 in Lecture Notes in Computer Science, pages 286–300. Springer-Verlag, Berlin Germany, 1996. 101

[8] Helena Handschuh, Yiannis Tsiounis, and Moti Yung. Decision oracles are equivalent to matching oracles. In *International Workshop on Practice and Theory in Public Key Cryptography '99 (PKC '99)*, number 1560 in Lecture Notes in Computer Science, Kamakura, Japan, March 1999. Springer-Verlag, Berlin Germany. 101

[9] Stefan Wolf. *Information-theoretically and Computionally Secure Key Agreement in Cryptography.* PhD thesis, ETH Zürich, 1999. 101, 102, 111, 112

[10] Ueli M. Maurer and Stefan Wolf. Diffie-Hellman oracles. In Koblitz [32], pages 268–282. 101, 108

[11] Mike Burmester, Yvo Desmedt, and Jennifer Seberry. Equitable key escrow with limited time span (or, how to enforce time expiration cryptographically). In K. Ohta and D. Pei, editors, *Advances in Cryptology – ASIACRYPT '98*, number 1514 in Lecture Notes in Computer Science, pages 380–391. Springer-Verlag, Berlin Germany, 1998. 101

[12] Birgit Pfitzmann and Ahmad-Reza Sadeghi. Anonymous fingerprinting with direct non-repudiation. In Okamoto [33], pages 401–414. 101

[13] Jan Camenisch, Ueli Maurer, and Markus Stadler. Digital payment systems with passive anonymity-revoking trustees. In E. Bertino, H. Kurth, G. Martella, and E. Montolivo, editors, *Proceedings of the Fourth European Symposium on Research in Computer Security (ESORICS)*, number 1146 in Lecture Notes in Computer Science, pages 33–43, Rome, Italy, September 1996. Springer-Verlag, Berlin Germany. 102

[14] George Davida, Yair Frankel, Yiannis Tsiounis, and Moti Yung. Anonymity control in e-cash systems. In *Proceedings of the First Conference on Financial Cryptography (FC '97)*, number 1318 in Lecture Notes in Computer Science, pages 1–16, Anguilla, British West Indies, February 1997. International Financial Cryptography Association (IFCA), Springer-Verlag, Berlin Germany. 102

[15] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, number 1233 in Lecture Notes in Computer Science, pages 256–266. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1997. 102, 104, 110, 111

[16] Ueli M. Maurer and Stefan Wolf. Diffie-Hellman, Decision Diffie-Hellman, and discrete logarithms. In *IEEE Symposium on Information Theory*, page 327, Cambridge, USA, August 1998. 102

[17] Ueli M. Maurer and Stefan Wolf. Lower bounds on generic algorithms in groups. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT '98*, number 1403 in Lecture Notes in Computer Science, pages 72–84. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1998. 102

[18] Eli Biham, Dan Boneh, and Omer Reingold. Breaking generalized Diffie-Hellman modulo a composite is no easier than factoring. *Information Processing Letters*, 70:83–87, 1999. Also appeares in Theory of Cryptography Library, Record 97-14, 1997. 102

[19] Daniel M. Gordon. Designing and detecting trapdoors for discrete log cryptosystems. In E.F. Brickell, editor, *Advances in Cryptology – CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 66–75. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1993. 102, 108

[20] National Institute of Standards and Technology (NIST). The Digital Signature Standard (DSS). FIPS PUB 186-2, January 2000. 103, 108

[21] Ahmad-Reza Sadeghi and Michael Steiner. Assumptions related to discrete logarithms: Why subtleties make a real difference. Full version of paper, available from `http://www.semper.org/sirene/lit/abstrA1.html`. 104, 106, 109, 110, 112, 115

[22] Dan Boneh and Richard J. Lipton. Algorithms for black box fields and their application to cryptography. In Koblitz [32], pages 283–297. 104

[23] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55(2):165–172, 1994. Translated from Matematicheskie Zametki, 55(2):91–101, 1994. 104

[24] Christian Cachin, Klaus Kursawe, and Victor Shoup. Random oracles in Constantinople: Practical asynchronous Byzantine agreement using cryptography. In *Proceedings of the 19th Annual ACM Symposium on Principles of Distributed Computing*, Portland, Oregon, July 2000. ACM. Full version appeared as Cryptology ePrint Archive Report 2000/034 (2000/7/7). 108

[25] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, October 1980. 111

[26] S.C. Pohlig and M. E. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Transactions on Information Theory*, 24:106–110, 1978. 111

[27] J. M. Pollard. Monte carlo methods for index computation mod p. *Mathematics of Computation*, 32:918–924, 1978. 111

[28] Marc Fischlin. A note on security proofs in the generic model. In Okamoto [33], pages 458–469. 111

[29] Victor Shoup. On formal models for secure key exchange. Research Report RZ 3120 (#93166), IBM Research, April 1999. A revised version 4, dated November 15, 1999, is available from `http://www.shoup.net/papers/`. 116

[30] Dan Boneh. Personal Communication, October 2000. 116

[31] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *29th Annual Symposium on Theory Of Computing (STOC)*, pages 284–293, El Paso, TX, USA, May 1997. ACM Press. 116

[32] Neal Koblitz, editor. *Advances in Cryptology – CRYPTO '96*, number 1109 in Lecture Notes in Computer Science. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1996. 117, 118

[33] T. Okamoto, editor. *Advances in Cryptology – ASIACRYPT '2000*, number 1976 in Lecture Notes in Computer Science, Kyoto, Japan, 2000. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany. 117, 118