# The Decision Diffie-Hellman Problem

Dan Boneh
dabo@cs.stanford.edu
Stanford University

**Abstract**

The Decision Diffie-Hellman assumption (DDH) is a gold mine. It enables one to construct efficient cryptographic systems with strong security properties. In this paper we survey the recent applications of DDH as well as known results regarding its security. We describe some open problems in this area.

## 1 Introduction

An important goal of cryptography is to pin down the exact complexity assumptions used by cryptographic protocols. Consider the Diffie-Hellman key exchange protocol [12]: Alice and Bob fix a finite cyclic group $G$ and a generator $g$. They respectively pick random $a, b \in [1, |G|]$ and exchange $g^a, g^b$. The secret key is $g^{ab}$. To totally break the protocol a passive eavesdropper, Eve, must compute the Diffie-Hellman function defined as: $\text{DH}_g(g^a, g^b) = g^{ab}$. We say that the group $G$ satisfies the *Computational Diffie-Hellman assumption* (CDH) if no efficient algorithm can compute the function $\text{DH}_g(x, y)$ in $G$. Precise definitions are given in the next section. Recent results provide some limited reductions from computing discrete log to computing the Diffie-Hellman function [20, 3, 21].

Unfortunately, CDH by itself is not sufficient to prove that the Diffie-Hellman protocol is useful for practical cryptographic purposes. Even though Eve may be unable to recover the entire secret, she may still be able to recover valuable information about it. For instance, even if CDH is true, Eve may still be able to predict %80 of the bits of $g^{ab}$ with some confidence. With our current state of knowledge we are are unable to prove that, assuming CDH, no such attack exists (although we discuss some results along this line in Section 3.3). Consequently, based on CDH, one cannot simply use the bits of $g^{ab}$ as a shared key – CDH does not guarantee that Eve cannot predict these bits.

If $g^{ab}$ is to be the basis of a shared secret key, one must bound the amount of information Eve is able to deduce about it, given $g^a, g^b$. This is formally captured by the, much stronger, *Decision Diffie-Hellman assumption* (DDH) (defined in the next section). Loosely speaking, the DDH assumption states that no efficient algorithm can distinguish between the two distributions $\langle g^a, g^b, g^{ab} \rangle$ and $\langle g^a, g^b, g^c \rangle$ where $a, b, c$ are chosen at random in $[1, |G|]$. As we shall see in Section 3.1, the DDH assumption is equivalent to the (conceptually simpler) assumption saying there is no efficient probabilistic algorithm that given *any* triplet $\langle g^a, g^b, g^c \rangle$ in $G^3$ outputs "true" if $a = bc$ and "false" otherwise.

To illustrate the importance of DDH we show how it applies to secret key exchange. We observed above that, with our present knowledge, CDH alone does not enable one to securely use bits of $g^{ab}$ as a shared secret — based on CDH we cannot prove that Eve cannot predict some of these bits. Nevertheless, based on CDH alone Alice and Bob can derive one unpredictable bit (known as a hard core bit [16]) from $g^{ab}$. If, given $g^a, g^b$, Eve could predict the hard core bit of $g^{ab}$, she could also compute all of $g^{ab}$. Hence, based on CDH alone, to exchange a $k$ bit secret, Alice and Bob would have to run the Diffie-Hellman protocol $k$ times. Each time they extract one hard core bit which is provably unpredictable by Eve. This is clearly inefficient and undesirable[1]. In contrast, using DDH one can do much better. Suppose $|G| > 2^n$. One can prove that based on DDH it is possible to extract from a *single application* of the Diffie-Hellman protocol, $n/2$ bits which Eve cannot distinguish from a true random string. This is done by hashing $g^{ab}$ to an $n/2$ bit string using an application of the leftover hash lemma as explained in Section 4.1. This is an example of how DDH can be used to significantly increase the efficiency of a cryptographic protocol. We point out that implemented cryptographic systems that derive multiple bits from the Diffie-Hellman secret are implicitly relying on DDH, not CDH. Over the past several years DDH has been successfully used to simplify many cryptographic schemes. We discuss some of these in Section 4.

### 1.1 DDH in various group families

The DDH assumption is very attractive. However, one must keep in mind that it is a very strong assumption (far stronger than CDH). We note that in some groups the CDH assumption is believed to be true, yet the DDH assumption is trivially false. For example, consider the group $\mathbb{Z}_p^*$ for a prime $p$ and generator $g$. The Computational Diffie-Hellman problem is believed to be hard in this group. Yet, given $g^a, g^b$ one can easily deduce the Legendre symbol of $g^{ab}$. This observation gives an immediate method for distinguishing $\langle g^a, g^b, g^{ab}\rangle$ from $\langle g^a, g^b, g^c\rangle$ for random $a, b, c$. This simple attack explains why most group families in which DDH is believed to be intractable have prime order. We note that to foil the attack it suffices to ensure the group order not have any small prime divisors.

We give some examples of groups in which DDH is believed to be intractable. It is remarkable (and surprising) that in all these groups, the best known algorithm for DDH is a full discrete log algorithm.

1. Let $p = 2p_1 + 1$ where both $p$ and $p_1$ are prime. Let $Q_p$ be the subgroup of quadratic residues in $\mathbb{Z}_p^*$. It is a cyclic group of prime order. This family of groups is parameterized by $p$.
2. More generally, let $p = aq + 1$ where both $p$ and $q$ are prime and $q > p^{1/10}$. Let $Q_{p,q}$ be the subgroup of $\mathbb{Z}_p^*$ of order $q$. This family of groups is parameterized by both $p$ and $q$.
3. Let $N = pq$ where $p, q, \frac{p-1}{2}, \frac{q-1}{2}$ are prime. Let $T$ be the cyclic subgroup of order $(p-1)(q-1)$. Although $T$ does not have prime order, DDH is believed to be intractable. The group is parameterized by $N$.
4. Let $p$ be a prime and $E_{a,b}/\mathbb{F}_p$ be an elliptic curve where $|E_{a,b}|$ is prime. The group is parameterized by $p, a, b$.
5. Let $p$ be a prime and $J$ be a Jacobian of a hyper elliptic curve over $\mathbb{F}_p$ with a prime number of reduced divisors. The group is parameterized by $p$ and the coefficients of the defining equation.

[1] We note that if one assumes that $\mathrm{DH}_g(x, y)$ cannot be computed by any algorithm running in time $t$ then one can securely derive $\log t$ bits out of each invocation of the Diffie-Hellman protocol. This is only a minor improvement over the single bit extraction process described above. We also note that these hard core bits are not bits of $g^{ab}$. Rather, they are *derived* from $g^{ab}$ by viewing it as a bit string over $\mathbb{Z}_2$ and computing its inner product with a public random vector over $\mathbb{Z}_2$ of the same length. To apply the Goldreich-Levin theorem [16] to the Diffie-Hellman function one must make use of tricks described in [30, Sect. 5].

## 2 Definitions

We formally define the notion of indistinguishable distributions and the Decision Diffie-Hellman problem. Throughout the paper we use the term *efficient* as short hand for probabilistic polynomial time. We use the term *negligible* to refer to a function $\epsilon(n)$ which is smaller than $1/n^\alpha$ for all $\alpha > 0$ and sufficiently large $n$.

**Group families.** A group family $\mathbb{G}$ is a set of finite cyclic groups $\mathbb{G} = \{G_\mathfrak{p}\}$ where $\mathfrak{p}$ ranges over an infinite index set. We denote by $|\mathfrak{p}|$ the size of binary representation of $\mathfrak{p}$. We assume there is a polynomial time (in $|\mathfrak{p}|$) algorithm that given $\mathfrak{p}$ and two elements in $G_\mathfrak{p}$ outputs their sum.

**Instance generator.** An Instance Generator, $\mathcal{IG}$, for $\mathbb{G}$ is a randomized algorithm that given an integer $n$ (in unary), runs in polynomial time in $n$ and outputs some random index $\mathfrak{p}$ and a generator $g$ of $G_\mathfrak{p}$. Note that for each $n$, the Instance Generator induces a distribution on the set of indices $\mathfrak{p}$.

Examples of group families were given in the previous section. The index $\mathfrak{p}$ encodes the group parameters. For instance, for the group of points on an elliptic curve we may let $\mathfrak{p} = \langle p, a, b \rangle$ denote the curve $E_{a,b}/\mathbb{F}_p$. The instance generator is used to select a random member of $\mathbb{G}$ of the appropriate size. For instance, when $\mathbb{G}$ is the family of prime order subgroups of $\mathbb{Z}_p^*$ the instance generator, on input $n$, may generate a random $n$-bit prime $p$ such that $(p-1)/2$ is also prime. In some cases it may make sense to generate distributions other than uniform. For instance, one may wish to avoid primes of the form $2^k + 1$.

**Definition 2.1** *Let $\mathbb{G} = \{G_\mathfrak{p}\}$ be a group family.*

- *A* CDH *algorithm $\mathcal{A}$ for $\mathbb{G}$ is a probabilistic polynomial time (in $|\mathfrak{p}|$) algorithm satisfying, for some fixed $\alpha > 0$ and sufficiently large $n$:*

$$\Pr\left[\mathcal{A}(\mathfrak{p}, g, g^a, g^b) = g^{ab}\right] > \frac{1}{n^\alpha}$$

  *where $g$ is a generator of $G_\mathfrak{p}$. The probability is over the random choice of $\langle \mathfrak{p}, g \rangle$ according to the distribution induced by $\mathcal{IG}(n)$, the random choice of $a, b$ in the range $[1, |G_\mathfrak{p}|]$ and the random bits used by $\mathcal{A}$. The group family $\mathbb{G}$ satisfies the* CDH *assumption if there is no* CDH *algorithm for $\mathbb{G}$.*

- *A* DDH *algorithm $\mathcal{A}$ for $\mathbb{G}$ is a probabilistic polynomial time algorithm satisfying, for some fixed $\alpha > 0$ and sufficiently large $n$:*

$$\left|\Pr[\mathcal{A}(\mathfrak{p}, g, g^a, g^b, g^{ab}) = \text{“}\mathit{true}\text{”}] - \Pr[\mathcal{A}(\mathfrak{p}, g, g^a, g^b, g^c) = \text{“}\mathit{true}\text{”}]\right| > \frac{1}{n^\alpha}$$

  *where $g$ is a generator of $G_\mathfrak{p}$. The probability is over the random choice of $\langle \mathfrak{p}, g \rangle$ according to the distribution induced by $\mathcal{IG}(n)$, the random choice of $a, b, c$ in the range $[1, |G_\mathfrak{p}|]$ and the random bits used by $\mathcal{A}$. The group family $\mathbb{G}$ satisfies the* DDH *assumption if there is no* DDH *algorithm for $\mathbb{G}$.*

The difference between the two probabilities in the definition of DDH is often called the *advantage* of algorithm $\mathcal{A}$. The definition captures the notion that the distributions $\langle \mathfrak{p}, g, g^a, g^b, g^{ab} \rangle$ and $\langle \mathfrak{p}, g, g^a, g^b, g^c \rangle$ are *computationally indistinguishable*. We will occasionally refer to the related notion of *statistically indistinguishable* distributions, defined as follows:

**Definition 2.2** *Let $\{\mathcal{X}_\mathfrak{p}\}$ and $\{\mathcal{Y}_\mathfrak{p}\}$ be two ensembles of probability distributions, where for each $\mathfrak{p}$ both $\mathcal{X}_\mathfrak{p}$ and $\mathcal{Y}_\mathfrak{p}$ are defined over the same domain $\mathcal{D}_\mathfrak{p}$. We say that the two ensembles are* statistically indistinguishable *if the statistical distance between them is negligible, i.e.*

$$Var(\mathcal{X}_\mathfrak{p}, \mathcal{Y}_\mathfrak{p}) = \sum_{a \in \mathcal{D}_\mathfrak{p}} |\mathcal{X}_\mathfrak{p}(a) - \mathcal{Y}_\mathfrak{p}(a)| < \epsilon$$

*where $\epsilon = \epsilon(|\mathfrak{p}|)$ is negligible.*

# 3 Known results on the security of DDH

We survey some of the evidence that adds to our confidence in DDH. At the moment, this evidence is circumstantial. Proving a link between DDH and a known hard problem is a critical open problem in this area.

## 3.1 Randomized reduction

When studying the security of DDH one asks for the weakest assumption that implies DDH. Ideally, one would like to prove CDH implies DDH, or some other classic problem (e.g. factoring) implies DDH. At the moment these questions are open. Fortunately, one can prove that DDH is implied by a slightly weaker assumption: *perfect*–DDH.

**perfect–**DDH: Let $\mathbb{G} = \{G_\mathfrak{p}\}$ be a family of finite cyclic groups. A perfect–DDH algorithm $\mathcal{A}$ for $\mathbb{G}$ correctly decides with overwhelming probability whether a given triplet $(x, y, z) \in G_\mathfrak{p}^3$ is a proper Diffie-Hellman triplet. That is, for large enough $n$ we have

$$\begin{aligned} \Pr[\mathcal{A}(\mathfrak{p}, g, g^a, g^b, g^c) = \text{"true"} \mid a = bc] &> 1 - \epsilon \\ \Pr[\mathcal{A}(\mathfrak{p}, g^a, g^b, g^c) = \text{"true"} \mid a \neq bc] &< \epsilon \end{aligned}$$

where the probability is taken over the random bits of $\mathcal{A}$, the random choice of $a, b, c \in [1, |G_\mathfrak{p}|]$, and the choice of $\langle \mathfrak{p}, g \rangle$ according to the distribution induced by $\mathcal{IG}(n)$. As usual, $\epsilon = \epsilon(n)$ is a negligible function. We say that $\mathbb{G}$ satisfies the perfect–DDH assumption if there is no polynomial time perfect–DDH algorithm. A perfect–DDH algorithm does more than a DDH algorithm. Namely, it correctly decides whether $\text{DH}_g(x, y) = z$ for most triplets. In contrast, a DDH algorithm is only required to correctly decide with non-negligible advantage.

Stadler [31, Prop. 1] and independently Naor and Reingold [24] showed that the two assumption, DDH and perfect–DDH, are equivalent. This conversion of an imperfect oracle into a perfect one is done via a *random reduction*. We slightly strengthen the result by applying it to groups in which only an upper bound on size of the group is given, rather than the exact order. This is useful when discussing DDH in the group $\mathbb{Z}_N^*$ for some $N = pq$.

**Theorem 3.1** *Let $\mathbb{G} = \{G_\mathfrak{p}\}$ be a family of finite cyclic groups of prime order. Let $s(\mathfrak{p})$ be an efficiently computable function such that $|G_\mathfrak{p}| \leq s(\mathfrak{p})$ for all $\mathfrak{p}$. Then $\mathbb{G}$ satisfies the* DDH *assumption if and only if it satisfies the perfect–*DDH *assumption.*

**Proof Sketch** The fact that the DDH assumption implies perfect–DDH is trivial. We prove the converse. Let $\mathcal{O}$ be a DDH oracle. That is, there exists an $\alpha > 0$ such that for large enough $n$,

$$\left|\Pr[\mathcal{O}(\mathfrak{p}, g, g^a, g^b, g^{ab}) = \text{“true”}] - \Pr[\mathcal{O}(\mathfrak{p}, g, g^a, g^b, g^c) = \text{“true”}]\right| \geq \frac{1}{n^\alpha}$$

The probability is over the random choice of $a, b, c$ in $[1, |G_\mathfrak{p}|]$, and the random choice of $\langle \mathfrak{p}, g\rangle$ according to the distribution induced by $\mathcal{IG}(n)$. We construct a probabilistic polynomial time (in $s(\mathfrak{p})$ and $|\mathfrak{p}|$) perfect–DDH algorithm, $\mathcal{A}$, which makes use of the oracle $\mathcal{O}$. Given $\mathfrak{p}, g$ and $x, y, z \in G_\mathfrak{p}$ algorithm $\mathcal{A}$ must determine with overwhelming probability whether it is a valid Diffie-Hellman triplet or not. Consider the following statistical experiment: pick random integers $u_1, u_2, v$ in the range $[1, s(\mathfrak{p})^2]$ and construct the triplet

$$(x', y', z') = (x^v g^{u_1},\ y g^{u_2},\ z^v y^{u_1} x^{v u_2} g^{u_1 u_2})$$

**Case 1.** Suppose $(x, y, z)$ is a valid triplet, then $x = g^a,\ y = g^b,\ z = g^{ab}$ For some $a, b$. It follows that $(x', y', z')$ is also a valid triplet. Furthermore, one can show that $(x', y', z')$ is chosen from a distribution which is statistically indistinguishable from the uniform distribution on proper Diffie-Hellman triplets in $G_\mathfrak{p}$.

**Case 2.** Suppose $(x, y, z)$ is not a valid triplet. Then $x = g^a,\ y = g^b,\ z = g^{ab+c}$ for some $c \neq 0$. In this case, $x' = g^{a'},\ y' = g^{b'},\ z' = g^{a'b'} g^{cv}$. Note that since $c \neq 0$ we know that $g^c$ is a generator of $G_\mathfrak{p}$. Consequently, the distribution of $g^{cv}$ is statistically indistinguishable from uniform. It is not difficult to show that the distribution on $(x', y', z')$ is statistically indistinguishable from the uniform distribution on $G_\mathfrak{p}^3$.

We see that based on whether $(x, y, z)$ is a valid Diffie-Hellman triplet we either generate a uniformly random valid triplet or a completely random triplet. Consequently, standard amplification techniques can be used to construct the algorithm $\mathcal{A}$. We describe a simple approach. Algorithm $\mathcal{A}$ performs two experiments: it first generates $k$ independent triplets $(x', y', z')$ as described above and queries the oracle at those triplets. Let $w_1$ be a random variable counting the number of times the oracle answers “true”. In the second experiment, $\mathcal{A}$ generates $k$ random triplets in $G_\mathfrak{p}^3$ and queries the oracle. Let $w_2$ be a random variable counting the number of “true” answers. Clearly, $E[|w_1 - w_2|] = 0$ if $(x, y, z)$ is an invalid triplet and $E[|w_1 - w_2|] > \epsilon k$ otherwise. Here $\epsilon = \epsilon(n) \geq 1/n^\alpha$ is the advantage produced by the oracle $\mathcal{O}$. Algorithm $\mathcal{A}$ outputs “true” if $|w_1 - w_2| > \epsilon k/2$ and outputs “false” otherwise. Using standard large deviation bounds one can show that when $k > \frac{1}{\epsilon} \log^2 \frac{1}{\delta}$ algorithm $\mathcal{A}$ outputs the right answer with probability at least $1 - \delta$. □

Observe that the only place where we use the fact that the group order is prime is in arguing that $g^c$ is a generator of $G_\mathfrak{p}$. This fact remains true, with high probability over the choice of $c$, as long as the smallest prime divisor of the group order is sufficiently large. Hence the theorem also applies in any group family $\mathbb{G}$ in which the smallest prime divisor of $|G_\mathfrak{p}|$ is super-polynomial in $|\mathfrak{p}|$. in particular, it applies to the group of quadratic residues in $\mathbb{Z}_N^*$ when $N = pq$ and $p = 2p_1 + 1$ and $q = 2q_1 + 1$ for some large primes $p, q, p_1, q_1$.

A random reduction such as Theorem 3.1 is an important part of any hardness assumption. Essentially, it shows that assuming one cannot decide the Diffie-Hellman problem with overwhelming probability then one cannot decide it in any non-negligible fraction of the input space.

## 3.2 Generic algorithms

Nechaev [26] and Shoup [30] describe models enabling one to argue about lower bounds on computations of discrete log as well as DDH. We use Shoup's terminology.

To disprove DDH one may first try to come up with a DDH algorithm that works in all groups. Indeed, such an algorithm would be devastating. However, the best known *generic* algorithm for DDH is a generic discrete log algorithm, namely the Baby-Step-Giant-Step [9]. When applied in a group of prime order $p$ this algorithm runs in time $O_\epsilon(\sqrt{p})$. Shoup shows that this is the best possible *generic* algorithm for DDH. We discuss the implications of this result at the end of the section.

**Definition 3.1 (Shoup)**

**An encoding function** *on the additive group* $\mathbb{Z}_p^+$ *is an injective map* $\sigma : \mathbb{Z}_p \to \{0,1\}^n$ *for some integer* $n > 0$.

**A generic algorithm** $\mathcal{A}$ *for* $\mathbb{Z}_p^+$ *is a probabilistic algorithm that takes as input an* encoding list $(\sigma(x_1), \ldots, \sigma(x_k))$ *where* $\sigma$ *is an encoding function and* $x_i \in \mathbb{Z}_p^+$. *During its execution, the algorithm may query an oracle by giving it two indices* $i, j$ *into the encoding list and a sign bit. The oracle returns the encoding* $\sigma(x_i \pm x_j)$ *according to the sign bit. This new encoding is then added to the encoding list. Eventually, the algorithm terminates and produces a certain output. The output is denoted by* $\mathcal{A}(\sigma; x_1, \ldots, x_k)$.

To illustrate these concepts we describe two encodings of $\mathbb{Z}_p^+$. Let $q$ be a prime with $p$ dividing $q-1$. Let $g \in \mathbb{Z}_q^*$ have order $p$. Then $\sigma$ defined by $\sigma(a) = g^a \bmod q$ is an encoding of $\mathbb{Z}_p^+$ inside $\mathbb{Z}_q^*$. Another encoding could be defined using an elliptic curve over $\mathbb{F}_q$ with $p$ points. Let $P$ be a points on the curve. Then $\sigma(a) = aP$ is another encoding of $\mathbb{Z}_p^+$. As an example of a generic algorithm we mentioned the Baby-Step-Giant-Step algorithm for discrete log. On the other hand, the index calculus method for computing discrete log is not generic. It takes advantage of the encoding of group elements as integers.

Shoup proved a number of lower bounds on generic algorithms. These include lower bounds on computing discrete log, computing Diffie-Hellman, deciding Diffie-Hellman and a few others. Here, we are most interested in the lower bound on deciding Diffie-Hellman.

**Theorem 3.2 (Shoup)** *Let* $p$ *be a prime and* $S \subset \{0,1\}^*$ *a set of at least* $p$ *binary strings. Let* $\mathcal{A}$ *be a generic algorithm for* $\mathbb{Z}_p^+$ *that makes at most* $m$ *oracle queries. Let* $a, b, c \in \mathbb{Z}_p^+$ *be chosen at random, let* $\sigma : \mathbb{Z}_p^+ \to S$ *be a random encoding function, and let* $s$ *be a random bit. Set* $w_0 = ab$ *and* $w_1 = c$. *Then*

$$\left| \Pr[\mathcal{A}(\sigma; 1, a, b, w_s, w_{1-s}) = s] - \frac{1}{2} \right| < m^2/p$$

*where the probability is over the random choice of* $a, b, c$ *in* $[1,p]$, *the random encoding* $\sigma$ *and the random bits used by the algorithm.*

**Proof Sketch** We bound the amount of information available to the algorithm after $m$ queries. Each time the algorithm interacts with the oracle it learns the encoding $\sigma(x_i)$ of some $x_i \in \mathbb{Z}_p^+$. One can easily see that $x_i = F_i(a, b, c, ab)$ where $F_i$ is a linear function that can be easily deduced by examining the oracle's previous queries. Suppose that for all $i, j$ such that $F_i \neq F_j$ one has that $\sigma(x_i) \neq \sigma(x_j)$. This means the algorithm learned the random encoding of *distinct* values. Since these values are independent random bit strings they provide no information to the algorithm.

The only way the algorithm obtains any information is if for some $i, j$ with $F_i \neq F_j$ we have that $\sigma(x_i) = \sigma(x_j)$. In this case the algorithm may learn a linear relation on the values $a, b, c, ab$. We give the algorithm the benefit of the doubt, and say that if it is able to find such an $F_i, F_j$ then it is able to produce the correct output. Hence, to bound the success probability, it suffices to bound the probability that given arbitrary distinct $m$ linear polynomials and random $a, b, c, ab \in \mathbb{Z}_p$ there exists an $i \neq j$ such that $F_i(a, b, c, ab) = F_j(a, b, c, ab)$. Let $R$ be this event. We bound $\Pr[R]$. For a given $F_i \neq F_j$ the number of solutions to $F_i(x, y, z, xy) = F_j(x, y, z, xy)$ can be bounded by considering the polynomial $G(x, y, z) = F_i - F_j$. This is a polynomial of total degree 2. Consequently, the probability that a random $(x, y, z) \in Z_p^3$ is a zero of $G$ is bounded by $2/p$ (see [29]). There are $\binom{m}{2}$ such pairs $F_i, F_j$ to consider. Hence, the probability that a random $(x, y, z, xy)$ is the root of some $F_i - F_j$ is bounded by

$$\Pr[R] \leq \binom{m}{2} \cdot \frac{2}{p} < \frac{m^2}{p}$$

The theorem now follows. When $R$ does not occur the algorithm can only guess the answer getting it right with probability half. The only information comes from the event $R$ which occures with probability less than $m^2/p$. $\square$

The theorem shows that any generic algorithm whose running time is less that $(\sqrt{p})^{1-\epsilon}$ fails to solve DDH, with non-negligible advantage, on a *random* encoding of the group $\mathbb{Z}_p^+$. It follows that there *exists* an encoding where the algorithm must fail. Hence, the theorem shows that if a generic algorithm is to obtain a non-negligible advantage in solving DDH it must run in exponential time (in $\log p$). This lower bound shows there is no efficient *generic* DDH algorithm that works in *all* groups. It is important to keep this in mind when searching for efficient DDH algorithms. The algorithm must make use of the particular group encoding.

Using a similar argument Maurer and Wolf [22] show that no efficient generic algorithm can reduce CDH to DDH. That is, suppose that in addition to the group action oracle, the algorithm also has access to an oracle for deciding DDH (i.e. given $\langle \sigma(a), \sigma(b), \sigma(c) \rangle$ the oracle returns "true" if $a = bc$ and "false" otherwise). Then any generic algorithm given $\sigma(x), \sigma(y)$ and making a total of at most $m$ oracle queries will succeed in computing $\sigma(xy)$ with probability at most $m^2/p$. This is important to keep in mind when searching for a reduction from CDH to DDH.

At a first reading the implications of Theorem 3.2 may not be clear. To avoid any confusion we point out a few things the theorem *does not* imply.

- The theorem cannot be applied to any specific group. That is, the theorem does not imply that in $\mathbb{Z}_p^*$ there is no sub-exponential algorithm for DDH. In fact, we know that such an algorithm exists. Similarly, the theorem implies nothing about the group of points on an elliptic curve.
- The theorem *does not* imply that there *exists* an encoding of $\mathbb{Z}_p^+$ for which DDH is true. It is certainly possible that for *every* encoding there exists a DDH algorithm that takes advantage of that particular encoding.

### 3.3 Security of segments of the Diffie-Hellman secret

Ideally, one would like to prove that CDH implies DDH. To so, one must provide a reduction showing that an oracle for breaking the decision problem can be used to break the computational problem.

This is appears to be a hard open problem. Nonetheless, one may try to prove weaker results regarding the security of Diffie-Hellman bits. Unfortunately, even proving that computing one bit of $g^{ab}$ given $g^a$ and $g^b$ is as hard as CDH is open. Currently, the only result along these lines is due to Boneh and Venkatesan [4]. At the moment these results only apply to the group $\mathbb{Z}_p^*$ and its subgroups. We define the $k$ most significant bits of an elements $x \in \mathbb{Z}_p^*$ as the $k$ most significant bits of $x$ when viewed as an integer in the range $[0, p)$.

**Theorem 3.3 (Boneh-Venkatesan)** *Let $p$ be an $n$-bit prime and $g \in \mathbb{Z}_p^*$. Let $\epsilon > 0$ be a fixed constant and set $k = k(n) = \lceil \epsilon\sqrt{n} \rceil$. Suppose there exists an expected polynomial time (in $n$) algorithm, $\mathcal{A}$, that given $p, g, g^a, g^b$ computes the $k$ most significant bits of $g^{ab}$. Then there is also an expected polynomial time algorithm that given $p, g, g^a, g^b$ computes all of $g^{ab}$.*

**Proof Sketch** The proof relies on lattice basis reductions and the LLL algorithm [19]. Given some $g^a$ and $g^b$ we wish to compute all of $g^{ab}$. To do so, we pick one random $r$ and apply $\mathcal{A}$ to the points $g^{a+r}, g^{b+t}$ for many random values of $t$. Consequently, we learn the most significant bits of $g^{(a+r)b} \cdot g^{(a+r)t}$. Notice that, with sufficiently high probability, $g^{a+r}$ is a generator of $\langle g \rangle$, the group generated by $g$. Hence, $g^{(a+r)t}$ is a random element of $\langle g \rangle$. The problem is now reduced to the following: let $\alpha = g^{(a+r)b}$; we are given the most significant bits of $\alpha$ multiplied by random elements in $\langle g \rangle$; find $\alpha$. To solve this problem one makes use of the LLL algorithm. This requires some work since one must prove that even though LLL does not produce a shortest vector, one is still able to find the correct $\alpha$. Indeed, the quality of the shortest vector produced by LLL implies the $\sqrt{\log p}$ bound on the number of necessary bits. To prove the result for $\epsilon < 1$ one makes use of Schnorr's improvement of the LLL algorithm [28]. Once $\alpha$ is found, recovering $g^{ab}$ is trivial. $\square$

The result shows that under CDH there is no efficient algorithm that computes roughly $\sqrt{\log p}$ bits of the Diffie-Hellman secret. To illustrate this, one may take $\epsilon = 1$. In this case when $p$ is 1024 bits long, under CDH one cannot compute the 32 leading bits. The same result holds for the least significant bits as well. The smaller the value of $\epsilon$ the longer the running time of the reduction algorithm. The running time is exponential in $1/\epsilon$.

The result is a first step in arguing about the security of segments of the Diffie-Hellman secret based on CDH. Hopefully, future results will show that fewer bits are required to reconstruct the entire secret. Interestingly, this is the only result where the LLL algorithm is used to prove the security of a cryptographic primitive. Usually, LLL is used to attack cryptosystems (for example, consider Coppersmith's low exponent attacks on RSA [10]).

### 3.4 Statistical results

Although we cannot give bounds on the computational complexity of DDH some results are known on the statistical distribution of proper Diffie-Hellman triples in the group $\mathbb{Z}_p^*$. Recently, Canetti, Friedlander and Shparlinski [7] showed that the triples $(g^a, g^b, g^{ab})$ are uniformly distributed modulo $p$ in the sense of Weyl.

Let $p$ be a prime and $g$ a generator of $\mathbb{Z}_p^*$. Let $B$ be a box of size $|B| = h_1 h_2 h_3$. That is,

$$B = [k_1, k_1 + h_1 - 1] \times [k_2, k_2 + h_2 - 1] \times [k_3, k_3 + h_3 - 1]$$

where $0 \le k_i \le k_1 + h_i - 1 \le p - 1$. We denote by $N(B)$ the number of Diffie-Hellman triples $(g^a, g^b, g^{ab})$ that when reduced modulo $p$ fall in the box $B$. Suppose Diffie-Hellman triples were randomly scattered in $(\mathbb{Z}_p)^3$. Since there are $(p-1)^2$ triples over all, one would expect $(p-1)^2 \cdot |B|/(p-1)^3$ of these to fall inside the box. Denote the discrepancy by

$$\Delta = \sup_B \left| N(B) - \frac{|B|}{p-1} \right|$$

Then we know [7] that this discrepancy is small.

**Theorem 3.4 (CFS)** *Let $p$ be an $n$-bit prime and $g$ a generator of $\mathbb{Z}_p^*$. Then*

$$\Delta \leq O_\epsilon(p^{31/16}) = o(p^2)$$

The result shows that Diffie-Hellman triples are close to being uniformly distributed among the boxes in $\mathbb{Z}_p^3$. The proof is based on bounding certain exponential sums. One can give an interesting interpretation of this result using statistical independence. For binary strings $x, y, z$ define $M_k(x, y, z)$ to be the string obtained by concatenating the $k$ most significant bits of $x$ to the $k$ most significant bits of $y$ to the $k$ most significant bits of $z$. Recall that the statistical distance between two distributions $\mathcal{P}_1$ and $\mathcal{P}_2$ over $\{0,1\}^{3k}$ is defined by

$$\mathrm{Var}(\mathcal{P}_1, \mathcal{P}_2) = \sum_x |\mathcal{P}_1(x) - \mathcal{P}_2(x)|$$

**Corollary 3.5 (CFS)** *Let $p$ be an $n$-bit prime and set $k = \lceil \gamma n \rceil$ for some constant $\gamma < 1/48$. Let $g$ be a generator of $\mathbb{Z}_p^*$. Define the following two distributions over $\{0,1\}^{3k}$:*

- *$\mathcal{P}_1$ is the uniform distribution among all strings in the set $\left\{ M_k(g^a, g^b, g^{ab}) \right\}$ where $a, b$ are in the range $[1, p]$ and $g^a, g^b, g^{ab}$ are reduced modulo $p$.*
- *$\mathcal{P}_2$ is the uniform distribution on $\{0,1\}^{3k}$.*

*Then the statistical distance between $\mathcal{P}_1$ and $\mathcal{P}_2$ is $Var(\mathcal{P}_1, \mathcal{P}_2) \leq e^{-c(\gamma)n}$ where $c(\gamma) > 0$ is a constant depending only on $\gamma$.*

The corollary shows that given the $k$ most significant bits of $g^a, g^b$ one cannot distinguish (in the statistical sense) the $k$ most significant bits of $g^{ab}$ from a truly random $k$ bit string. This is quite interesting although it does not seem to apply to the security analysis of existing protocols. In most protocols the adversary learns all of $g^a$ and $g^b$. The authors claim that a similar result holds for subgroups of $\mathbb{Z}_p^*$ as long as the index is "not too large".

# 4 Applications of Decision Diffie-Hellman (DDH)

We briefly describe some applications of DDH that show why it is so attractive to cryptographers.

## 4.1 ElGamal encryption

Let $p$ be a prime and $g \in \mathbb{Z}_p^*$. The ElGamal public key system encrypts a message $m \in \mathbb{Z}_p$ given a public key $g^a$ by computing $\langle g^b,\ m \cdot g^{ab} \rangle$. Here $b$ is chosen at random in $[1, \mathrm{ord}(g)]$. Decryption using the private key $a$ is done by first computing $g^{ab}$ and then dividing to obtain $m$.

When $g$ is a generator of $\mathbb{Z}_p^*$ the system in not semantically secure[2]. Some information about the plaintext is revealed. Namely, the Legendre symbol of $g^a, g^b$ completely exposes the Legendre symbol of $m$. In case the symbol of $m$ encodes important information, the system is insecure. This is an example where even though the CDH assumption is believed to be true, the system leaks information. To argue that the ElGamal system is semantically secure one must rely on the DDH assumption. Let $G$ be a group in which the DDH assumption holds and $g$ a generator of $G$. Then, assuming the message space is restricted to $G$ it is easy to show that the system is semantically secure under DDH. This follows since given $g^a, g^b$ the secret pad $g^{ab}$ cannot be distinguished from a random group element. It follows that $m \cdot g^{ab}$ cannot be distinguished from a random group element. Consequently, given the ciphertext, an attacker cannot deduce any extra information about the plaintext.

To summarize, DDH is crucial for the security analysis of the ElGamal system. CDH by itself is insufficient. Notice that in the above argument we rely on the fact that the plaintext space is equal to the group $G$. This is somewhat cumbersome since often one wishes to encrypt an $n$-bit string rather than a group element. This can be easily fixed using hashing. Suppose $|G| > 2^n$. Then assuming DDH, the string $g^{ab}$ has at least $n$ bits of *computational entropy* [18]. Note that the bit string representing $g^{ab}$ may be much longer. Hashing $g^{ab}$ to an $m$-bit string for some $m \leq n$ results in a bit-string indistinguishable from random. Encryption can be done by xoring this $m$ bit hashed string with the plaintext. To formally argue that this hashing results in a pseudo random string one makes use of the leftover hash lemma [18] and pairwise independent hash functions.

## 4.2 Efficient pseudo random functions

Naor and Reingold [24] describe a beautiful application of DDH. They show how to construct a collection of efficient pseudo random functions. Such functions can be used as the basis of many cryptographic schemes including symmetric encryption, authentication [14] and digital signatures [1]. Prior to these results, existing constructions [15, 23] based on number theoretic primitives were by far less efficient.

Pseudo random functions were first introduced by Goldreich, Goldwasser and Micali [15]. At a high level, a set $F_n$ of functions $A_n \mapsto B_n$ is called a *pseudo random function ensemble* if no efficient statistical test can distinguish between a random function chosen in the set and a truly random function, i.e. a function chosen at random from the set of all functions $A_n \mapsto B_n$. Here $A_n, B_n$ are finite domains. The statistical test is only given "black-box" access to the function. That is, it can ask an oracle to evaluate the given function at a point of its choice, but cannot peak at the internal implementation. We refer to [24] for the precise definition.

Let $\mathbb{G} = \{G_\mathfrak{p}\}$ be a group family. For a given value of $n \in \mathbb{N}$, the Naor-Reingold pseudo-random function ensemble, $F_n$, is a set of functions from $\{0,1\}^n$ to $G_\mathfrak{p}$ for some $\mathfrak{p}$ (the index $\mathfrak{p}$ may be different for different functions in the ensemble). A function in the set is parameterized by a seed $s = \langle \mathfrak{p}, g, \vec{a} \rangle$ where $g$ is a generator of $G_\mathfrak{p}$ and $\vec{a} = (a_0, \ldots, a_n)$ is a vector of $n+1$ random integers in the range $[1, |G_\mathfrak{p}|]$. The value of the function at a point $x = x_1 x_2 \ldots x_n \in \{0,1\}^n$ is defined by

$$f_{\mathfrak{p},g,\vec{a}}(x) = g^{a_0 \prod_{i=1}^{n} a_i^{x_i}}$$

The distribution on the seed $s$ is induced by the random choice of $\vec{a}$ and the distribution induced on

[2]Semantic security [17] is the standard security notion for an encryption scheme. It essentially says that any information about the plaintext an eavesdropper can obtain given the ciphertext, can also be obtained without the ciphertext.

$\langle \mathfrak{p}, g \rangle$ by $\mathcal{IG}(n)$.

In what follows, we let $\mathcal{A}^f$ denote the algorithm $\mathcal{A}$ with access to an oracle for evaluating the function $f$. The following theorem is the main result regarding the above construction.

**Theorem 4.1 (Naor-Reingold)** *Let $\mathbb{G}$ be a group family and let $\{F_n\}_{n\in\mathbb{N}}$ be the Naor-Reingold pseudo-random function ensemble. Suppose the* DDH *assumption holds for $\mathbb{G}$. Then for every probabilistic polynomial time algorithm $\mathcal{A}$ and sufficiently large $n$, we have that*

$$\left|\Pr[\mathcal{A}^{f_{\mathfrak{p},g,\vec{a}}}(\mathfrak{p},g) = \text{“true”}] - \Pr[\mathcal{A}^{R_{\mathfrak{p},g,\vec{a}}}(\mathfrak{p},g) = \text{“true”}]\right| < \epsilon$$

*where $\epsilon = \epsilon(n)$ is negligible. The first probability is taken over the choice of the seed $s = \langle \mathfrak{p}, g, \vec{a} \rangle$. The second probability is taken over the random distribution induced on $\mathfrak{p}, g$ by $\mathcal{IG}(n)$ and the random choice of the function $R_{\mathfrak{p},g,\vec{a}}$ among the set of all $\{0,1\}^n \mapsto G_{\mathfrak{p}}$ functions.*

The evaluation of a function $f_{\mathfrak{p},g,\vec{a}}(x)$ in the Naor-Reingold construction can be can be done very efficiently (compared to other constructions). Essentially, one first computes the product $r = a_0 \prod_{i=1}^{n} a_i^{x_i} \bmod |G_{\mathfrak{p}}|$ and then computes $g^r$. Hence, the evaluation requires $n$ modular multiplications and one exponentiation. Note that we are assuming the order of $G_{\mathfrak{p}}$ is known.

### 4.3 A cryptosystem secure against adaptive chosen ciphertext attack

Recently, Cramer and Shoup [11] presented a surprising application of DDH. They describe an efficient public key cryptosystem which is secure against adaptive chosen ciphertext attack. Security against such a powerful attack could only be obtained previously by extremely inefficient techniques [25, 27, 13] relying on constructions for non-interactive zero-knowledge (efficient *heuristic* constructions are described in [33]). In light of this, it is remarkable that the DDH assumption is able to dramatically simplify things.

An adaptive ciphertext attack is an attack where the adversary has access to a decryption oracle. The adversary is given a ciphertext $C = E(M)$. He can then query the oracle at arbitrary inputs of his choice. The only restriction is that the queries must be different than the given ciphertext $C$. The adversary's goal is to then deduce some information about the plaintext $M$ with non-negligible advantage. To motivate this notion of security we point out that the standard semantic security model [17] provides security against passive (i.e. eavesdropping) attacks. It does not provide any security against an *active* attacker who is able to influence the behavior of honest parties in the network. In contrast, security against adaptive chosen ciphertext attacks provides security against *any* active adversary.

Clearly, a cryptosystem secure against an adaptive attack must be *non-malleable* – given $C$ one should not be able to construct a $C'$ such that the decryption of $C$ and $C'$ are correlated in any way. Indeed, if this were not the case, the attacker would simply query the decryption oracle at $C'$ and learn information about the decryption of $C$. Thus, the Cramer-Shoup cryptosystem is also non-malleable (assuming DDH). Non-malleable systems are needed in many scenarios (see [13]). For instance, to cheat in a bidding system, Alice may not need to discover Bob's bid. She may only want to offer a lower bid. Thus, if Bob encrypts his bid using a *malleable* system, Alice may be able to cheat by creating the encryption of a lower bid without having to break Bob's cipher. In case Bob encrypts his bid with a non-malleable system, this form of cheating is impossible.

### 4.4 Others

The DDH assumption is used in many other papers as well. We very briefly mention four (see also the summary in [24]). Recently, Canetti [6] described a simple construction based on DDH for a primitive called "Oracle Hashing". These are hash functions that let one test that $b = h(a)$, but given $b$ alone, they reveal *no information* about $a$. Bellare and Micali [2] use DDH to construct a non-interactive oblivious transfer protocol. Brands [5] pointed out that several suggestions for undeniable signatures [8] implicitly rely on DDH. Steiner, Tsudik and Waidner [32] show that DDH implies generalized–DDH. They consider a generalization of Diffie-Hellman enabling a group of parties to exchange a common secret key. For example, in the case of three parties, each party picks a random $x_i$, they publicly compute $g^{x_i}, g^{x_i x_j}$ for $1 \leq i < j \leq 3$ and set their common secret to $g^{x_1 x_2 x_3}$. This suggests a generalization of the DDH assumption. Fortunately, Steiner, Tsudik and Waidner show that, for a constant number of parties, DDH implies the generalized–DDH.

## 5 Conclusions and open problems

The Decision Diffie-Hellman assumption appears to be a very strong assumption, yet the best known method for breaking it is computing discrete log. The assumption plays a central role in improving the performance of many cryptographic primitives. We presented the known evidence for its security. This evidence includes (1) a worst-case to average case reduction for DDH. (2) no generic algorithm can break DDH. (3) certain pieces of the Diffie-Hellman secret are provably as hard to compute as the entire secret. (4) statistically, Diffie-Hellman triplets are uniformly distributed (in the sense of Weyl).

We conclude with a list of the main open problems in this area. Progress on any of these would be most welcome.

**Open problems:**

1. Is there an algorithm for DDH in a prime order subgroup of $\mathbb{Z}_p^*$ whose running time is better than the fastest discrete log algorithm in that subgroup? This is perhaps the most interesting problem related to DDH. It is almost hard to believe that computing discrete log is the best method for testing that a triplet $\langle x, y, z \rangle$ satisfies the Diffie-Hellman relation. At the moment we are powerless to settle this question one way or another.

2. Is there a group family in which DDH is implied by some "standard" cryptographic assumption, e.g. CDH, or factoring? For instance, let $N = pq$ where $p = 2p_1 + 1$ and $q = 2q_1 + 1$ with $p, q, p_1, q_1$ prime. Does the DDH assumption in $\mathbb{Z}_N^*$ follow from the hardness of distinguishing quadratic residues from non residues with Jacobi symbol $+1$ ?

3. Can one improve the results of [4] (see Section 3.3) and show that in $\mathbb{Z}_p^*$ the single most significant bit of the Diffie-Hellman secret is as hard to compute as the entire secret? Also, does a similar result to that of [4] hold in the group of points of an elliptic curve?

## Acknowledgments

The author thanks Victor Shoup for many insightful comments on an early draft of this paper.

## References


[1] M. Bellare, S. Goldwasser, "New paradigms for digital signatures and message authentication based on non-interactive zero-knowledge proofs" Crypto '89, pp. 194–211.

[2] M. Bellare, S. Micali, "Non-interactive oblivious transfer and applications", Crypto '89, pp. 547–557.

[3] D. Boneh, R. Lipton, "Black box fields and their application to cryptography", Proc. of Crypto '96, pp. 283–297.

[4] D. Boneh, R. Venkatesan, "Hardness of computing most significant bits in secret keys of Diffie-Hellman and related schemes", Proc. of Crypto '96, pp. 129–142.

[5] S. Brands, "An efficient off-line electronic cash system based on the representation problem", CWI Technical report, CS-R9323, 1993.

[6] R. Canetti, "Towards realizing random oracles: hash functions that hide all partial information", Proc. Crypto '97, pp. 455–469.

[7] R. Canetti, J. Friedlander, I. Shparlinski, "On certain exponential sums and the distribution of Diffie-Hellman triples", Manuscript.

[8] D. Chaum, H. van Antwerpen, "Undeniable signatures", Proc. Crypto '89, pp. 212–216.

[9] H. Cohen, "A course in computational number theory", Springer-Verlag.

[10] D. Coppersmith, "Finding a Small Root of a Bivariate Integer Equation; Factoring with high bits known", Proc. Eurocrypt '96, 1996.

[11] R. Cramer, V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack", manuscript.

[12] W. Diffie, M. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, vol. 22, no. 6, pp. 644–654, 1976.

[13] D. Dolev, C. Dwork, M. Naor, "Non-malleable cryptography", Proc. STOC' 91, pp. 542–552.

[14] O. Goldreich, S. Goldwasser, S. Micali, "On the cryptographic applications of random functions", Crypto' 84, pp. 276–288.

[15] O. Goldreich, S. Goldwasser, S. Micali, "How to construct random functions", J. ACM, Vol. 33, 1986, pp. 792–807.

[16] O. Goldreich, L.A. Levin, "Hard core bits based on any one way function", Proc. STOC '89.

[17] S. Goldwasser, S. Micali, "Probabilistic encryption", J. Computer and Syst. Sciences, Vol. 28, 1984, pp. 270–299.

[18] J. Hastad, R. Impaglizzo, L. Levin, M. Luby, "Construction of pseudo random generators from one-way functions", SIAM J. of Computing, to appear. Also see preliminary version in STOC' 89.

[19] A. Lenstra, H. Lenstra, L. Lovasz, “Factoring polynomial with rational coefficients”, Mathematiche Annalen, 261:515–534, 1982.

[20] U. Maurer, “Towards proving the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms”, Proc. of Crypto ’94, pp. 271–281.

[21] U. Maurer, S. Wolf, “Diffie-Hellman oracles”, Proc. of Crypto ’96, pp. 268–282.

[22] U. Maurer, S. Wolf, “Lower bounds on generic algorithms in groups”, Proc. of Eurocrypt ’98, to appear.

[23] M. Naor, O. Reingold, “Synthesizers and their application to the parallel construction of pseudo-random functions”, Proc. FOCS ’95, pp. 170–181.

[24] M. Naor, O. Reingold, “Number theoretic constructions of efficient pseudo random functions”, Proc. FOCS ’97. pp. 458–467.

[25] M. Naor, M. Yung, “Public key cryptosystems provable secure against chosen ciphertext attacks”, STOC ’90, pp. 427–437

[26] V. Nechaev, “Complexity of a determinate algorithm for the discrete logarithm”, Mathematical Notes, Vol. 55 (2), 1994, pp. 165–172.

[27] C. Rackoff, D. Simon, “Non-interactive zero knowledge proof of knowledge and chosen ciphertext attack”, Crypto’ 91, pp. 433–444.

[28] C. Schnorr, “A hierarchy of polynomial time lattice basis reduction algorithms”, Theoretical Computer Science, Vol. 53, 1987, pp. 201–224.

[29] J. Schwartz, “Fast probabilistic algorithms for verification of polynomial identities”, J. ACM, Vol. 27 (4), 1980, pp. 701–717.

[30] V. Shoup, “Lower bounds for discrete logarithms and related problems”, Proc. Eurocrypt ’97, pp. 256–266.

[31] M. Stadler, “Publicly verifiable secret sharing”, Proc. Eurocrypt ’96, pp. 190–199.

[32] M. Steiner, G. Tsudik, M. Waidner, “Diffie-Hellman key distribution extended to group communication”, Proc. 3rd ACM Conference on Communications Security, 1996, pp. 31–37.

[33] Y. Zheng, J. Seberry, “Practical approaches to attaining security against adaptively chosen ciphertext attacks”, Crypto ’92, pp. 292–304.