University of Maryland
CMSC456 — Introduction to Cryptography
Professor Jonathan Katz

Lecture 30

1 The Diffie-Hellman Problems

Last time we showed an encryption scheme based on the hardness of deciding quadratic residuosity. Unfortunately, this scheme (as currently defined) only allows encryption of 1-bit messages! Even worse, the scheme is inefficient: encrypting a single bit results in a ciphertext of length k (where this might be on the order of 1024 or so). An expansion factor of k is typically too inefficient to use.

Today, we will see an encryption scheme (based on a relatively strong — but reasonable — cryptographic assumption) with expansion factor 2. Furthermore, the scheme will immediately allow encryption of long messages. As usual, we will begin by introducing and formalizing our hardness assumptions, then construct a scheme, and then prove security of the scheme.

For the remainder of this topic, we will be working over a finite, cyclic group $\mathbb{G}$. Since $\mathbb{G}$ is cyclic, this means there is a *generator* $g \in \mathbb{G}$ for which $\mathbb{G} = \{g^0, \ldots, g^{|\mathbb{G}|-1}\}$ (recall that $g^{|\mathbb{G}|} = 1 = g^0$ so the cycle repeats at that point). Note that because g is a generator, $\log_g h$ is well-defined for any $h \in \mathbb{G}$ as the unique integer x in $\mathbb{Z}_{|\mathbb{G}|}$ for which $g^x = h$.

Given $\mathbb{G}$, we may define a number of problems; if these problems are hard we may formalize these as cryptographic assumptions. We discuss some of these problems now.

Discrete logarithm problem. The discrete logarithm problem is the following: given (random) generator g and random element $h \in \mathbb{G}$, compute $\log_g h$. If this problem is hard, we say *the discrete logarithm assumption holds in* $\mathbb{G}$. Note that it is always "easy" to compute discrete logarithms with probability $1/|\mathbb{G}|$: simply guess a random element in $\mathbb{Z}_{|\mathbb{G}|}$, and this will be the correct answer with probability $1/|\mathbb{G}|$. Thus, if we want the discrete logarithm to be "hard", it will have to be the case that $|\mathbb{G}|$ is large.

Computational Diffie-Hellman (CDH) problem. This problem is the following: given random generator g and random elements $h_1, h_2 \in \mathbb{G}$, compute $g^{(\log_g h_1)(\log_g h_2)}$ (i.e., if $h_1 = g^x$ and $h_2 = g^y$ the output should be g^{xy}). If this problem is hard, we say *the CDH assumption holds in* $\mathbb{G}$.

Before continuing, we give an example. Consider the group $\mathbb{Z}_{11}^* = \{1, \ldots, 10\}$ under multiplication. Since 11 is prime, it is a fact that $\mathbb{Z}_{11}^*$ is cyclic (for those who have had some more advanced algebra, this is because $\mathbb{Z}_{11}^*$ is a field and the non-zero elements of a finite field form a cyclic group). In this case, it can be verified that 2 is a generator. So, an instance of the CDH problem might be: given 2, 7, and 9, compute $2^{(\log_2 7)(\log_2 3)}$. Since $2^7 = 7$ and $2^6 = 9$, the correct answer is $2^{7 \cdot 9 \bmod 10} = 2^3 = 8$.

We may immediately note that the CDH assumption is *stronger* than the discrete logarithm assumption. In particular, the CDH assumption implies the discrete logarithm assumption, as the following claim shows:

Claim *If the discrete logarithm problem is "easy" in $\mathbb{G}$ then the CDH problem is "easy" in $\mathbb{G}$.*

Proof Assume the discrete logarithm problem is easy in $\mathbb{G}$, so that there is an efficient algorithm A that computes discrete logarithms with some probability ϵ (over choice of g and element h). We may then construct the following algorithm A' for the CDH problem:

$A'(g, h_1, h_2)$
Run $A(g, h_1)$ to get output x
if $g^x = h_1$ output h_2^x
otherwise, simply abort

Note that A' can always tell whether A has output the correct answer by computing g^x to see whether this matches h_1. Also, A' outputs the correct answer whenever A outputs the correct answer (since $h_2 = g^{\log_g h_2}$, the output of A' satisfies $h_2^x = g^{(\log_g h_1)(\log_g h_2)}$). So, A' outputs the correct answer with probability ϵ and if the discrete logarithm problem is easy (i.e., ϵ is large) then so is the CDH problem. ■

Decisional Diffie-Hellman (DDH) problem. Motivated by the CDH problem, define $\mathsf{CDH}_g(h_1, h_2)$ as $g^{(\log_g h_1)(\log_g h_2)}$. The DDH problem is the following: given four elements (g, h_1, h_2, h_3) (where g, as usual, is a generator), determine whether $h_3 = \mathsf{CDH}_g(h_1, h_2)$ or not. In particular (we define this problem more formally since we will use it to prove security of an encryption scheme, below), define a *Diffie-Hellman tuple* to be $(g, h_1, h_2, \mathsf{CDH}_g(h_1, h_2))$ and define a *random tuple* to be (g, h_1, h_2, h_3) (where h_3 will be random and independent of h_1, h_2). Then the DDH problem is to distinguish between random tuples and Diffie-Hellman tuples.

More formally, we may say that the DDH problem is (t, ϵ)-hard if for all algorithms A running in time t we have:

$$\left|\Pr[x, y \leftarrow \mathbb{Z}_{|\mathbb{G}|} : A(g, g^x, g^y, g^{xy}) = 1] - \Pr[x, y, z \leftarrow \mathbb{Z}_{|\mathbb{G}|} : A(g, g^x, g^y, g^z) = 1]\right|.$$

Note that the first tuple in the above expression corresponds to a Diffie-Hellman tuple, while the second corresponds to a random tuple.

We may immediately note that the DDH assumption is stronger than the CDH assumption. In particular, the DDH assumption implies the CDH assumption, as the following claim shows:

Claim *If the CDH problem is "easy" in $\mathbb{G}$ then the DDH problem is "easy" in $\mathbb{G}$.*

Proof Assume the CDH problem is easy in $\mathbb{G}$, so there is an efficient algorithm A that, given g, h_1, h_2, outputs $\mathsf{CDH}_g(h_1, h_2)$ with probability ϵ. We may then construct the following algorithm A' for the DDH problem:

$A'(g, h_1, h_2, h_3)$
Run $A(g, h_1, h_2)$ to get output h_3'
if $h_3' = h_3$ output 1
otherwise, output 0

Note that now there is no way for A' to verify whether A has output the correct answer or not (why?).

Let's analyze the output of A'. When (g, h_1, h_2, h_3) is a Diffie-Hellman tuple, A' outputs 1 iff A outputs the correct answer. Since this happens with probability ϵ, we have:

$$\Pr[x, y \leftarrow \mathbb{Z}_{|\mathbb{G}|} : A'(g, g^x, g^y, g^{xy}) = 1] = \epsilon.$$

On the other hand, when (g, h_1, h_2, h_3) is a random tuple, A' outputs 1 iff A happens to output $h_3' = h_3$. Since h_3 is random and independent of h_1 and h_2, this happens exactly with probability $1/|\mathbb{G}|$. So:

$$\Pr[x, y, z \leftarrow \mathbb{Z}_{|\mathbb{G}|} : A'(g, g^x, g^y, g^z) = 1] = 1/|\mathbb{G}|.$$

The difference in probabilities is then $\epsilon - 1/|\mathbb{G}|$. So, if ϵ is much greater than $1/|\mathbb{G}|$, then A' can distinguish between Diffie-Hellman tuples and random tuples easily. (Note that $\epsilon \leq 1/|\mathbb{G}|$ does not constitute "breaking" CDH; it is easy to achieve success probability $1/|\mathbb{G}|$ for the CDH problem by simply "guessing" a random group element!) ■

Thus, we have shown that the DDH assumption implies the CDH assumption which in turn implies the discrete logarithm assumption. In general, the converse of these statements is not true. There are groups in which the discrete logarithm problem is conjectured to be hard but the DDH problem is known to be easy, and there are also groups in which the CDH problem is conjectured to be hard but the DDH problem is known to be easy.

2 Application to Secure Encryption

With this machinery in place, we can now construct a more efficient public-key encryption scheme. Given a fixed group $\mathbb{G}$, we define the *El Gamal encryption scheme* [1] as follows:

1. $\mathcal{K}$ chooses a random generator $g \in \mathbb{G}$ and a random number $x \in \mathbb{Z}_{|\mathbb{G}|}$. It then computes $h = g^x$. The public key is (g, h) and the secret key is x.
2. The message space will be the group itself. To encrypt a message $m \in \mathbb{G}$, the sender picks a random $r \in \mathbb{Z}_{|\mathbb{G}|}$ and sends ciphertext $(g^r, h^r m)$.
3. To decrypt a ciphertext (A, B), the receiver computes $m = B/A^x$. (For any group elements g', h', the notation g'/h' simply means $g'(h')^{-1}$.)

We verify that decryption is done correctly. Since $(A, B) = (g^r, h^r m)$ for some r, we have:

$$\frac{B}{A^x} = \frac{h^r m}{(g^r)^x} = \frac{h^r m}{(g^x)^r} = \frac{h^r m}{h^r} = m.$$

We may immediately note that for the scheme to be secure, at the minimum we will require the discrete logarithm problem to be hard in $\mathbb{G}$ (if not, then anyone who knows the public key (g, h) can compute $x = \log_g h$ and learn the secret key). In fact, the security of the scheme relies on the stronger DDH assumption in $\mathbb{G}$. We will give a proof of security for El Gamal encryption next time.

References

[1] T. El Gamal. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory* 31(4): 469–472 (1985).