Cryptography, Card Tricks, and Kangaroos

Michael T. Lacey*
Georgia Institute of Technology

Abstract

A basic cryptographic protocol, the Diffie–Hellman Protocol serves as an introduction to cryptography, and the Discrete Logarithm Problem. The presumed hardness of the latter problem is the guarantor of the security of the Diffie–Hellman and many other commonly used protocols. This permits us to move to the real subject: Pollard's Kangaroo method to solve the discrete log problem. This method has a beautiful explanation by way of a card trick due to Kruskal.

1 Overview

This note sets out material appropriate for an undergraduate level talk, on a problem of significant current interest in cryptography. An annotated bibliography of the reading that the author has done in the field is included at the end of the note. But the reader is cautioned that the author is not an expert in this field, and deeper insights should be sought from those who are experts.

Cryptography has become a vital aspect of modern communication, one with a large variety of points of interest. These include the scientific, such as the development of novel protocols, and the study of the security of current protocols. These topics touch on deep aspects of combinatorics, group theory and number theory. Implementation issues are also of great interest, and highly non–trivial. Finally, societal and legal implications of cryptography effect daily life.

In this note, we only specify a single cryptographic protocol: The Diffie–Hellman Protocol. Invented in 1957, it initiated the era of public key cryptography. With it, two people can communicate a public way, and agree on a common private key. This private key can be used as the key for a longer private conversation using a secondary protocol.

*This work has been supported by an NSF grant, DMS–9706884.

The protocol is very simple, and as the patent on it expired in 1997, it is now commonly used in a wide variety of circumstances. For instance, the protocol can be used in a transparent way to secure cell phone conversations.

This protocol can be phrased in any cyclic group, a property shared by many cryptographic protocols. And the security of the protocol is not a fact, but experiential: Breaking it seems to be equivalent to the discrete logarithm problem, one that is widely believed to be hard.

And so, at that point, we will be at the threshold of an ingenious scheme to solve the discrete logarithm problem, the Kangaroo method of J.M. Pollard. This scheme is very nicely illustrated by a card trick. We describe the card trick. And then describe the Kangaroo method. It will solve the discrete log problem in time that is comparable to the square root of the size of the cyclic group.

So let us begin with the Diffie–Hellman protocol.

2 Diffie–Hellman Protocol

Alice and Bob are communicating over a public network, and want to agree on a private key: A piece of information known only to them. Here are the steps they take to do this.

0. Alice and Bob agree on an integer n and a choice of integer $g < n$ with g and n relatively prime.

1. Alice selects a private, random x and calculates $X = g^x \mod n$. Alice sends X to Bob.

2. Bob takes the same steps. Bob selects a private, random y and calculates $Y = b^y \mod n$. Bob sends Y to Alice.

3. Alice, receiving Y, computes $k = Y^x \mod n$.

4. Bob, receiving X, computes $k = X^y \mod n$.

At the end of the protocol, these pieces of information are public: n, g, X, and Y. It is most important to note that the private key $k = g^{xy} = X^y = Y^x \mod n$. That is, Alice and Bob have agreed on a common, private piece of information k.

In practice, k is then used to initiate a second pre–agreed encryption scheme, in order to secure a longer conversation.

What does it take to break the protocol? In practice, what this means is that a malicious person needs to discover x or y. That is, they need to solve for x in the equation

$$g^x = X \mod n$$

This is just one instance of the discrete logarithm problem.

We should hasten to mention that of course n should be large. g was chosen to be relatively prime to n so that it's powers generate a group in the integers mod n. g should have a large order. Namely, the least power p such that $g^p = 1 \mod n$ should be large. There are instances in which the problem turns out to be easy. But the "typical" instances have empirically turned out to be hard problems to solve, in that they take substantial amounts of computational time or space to solve.

Looking at the protocol again, you see that the only thing that is required is that there be a notion of "exponentiation" and the protocol will work. The general mathematical framework that is required for this protocol is that of a cyclic group G with generator g. Such groups have elements $\{1, g, g^2, \ldots, g^{n-1}\}$, where N is the number of elements in the group. The product of two elements is

$$g^x g^y = g^{x+y} = g^{x+y} \mod n$$

Note that the exponent addition works modulo n. And is otherwise just as you would expect.

Discrete Logarithm Problem. Given a cyclic group G with generator g and $X \in G$, find integer x such that $g^x = X$.

Below, we will argue that J.M. Pollard's Kangaroo method can solve this problem in about $\sqrt{\#G}$ steps.

3 Kruskal's Card Trick

This card trick is due to Kruskal. Each card in a full deck of cards will have a numerical value, the face value if it is a number card, the value 1 for Aces and face cards King, Queen and Jack receive the value 5.

Turn the first card face up. It's value is the size of the first jump. Turn over that many cards, noting the last card turned up. It is the first landing place and it's value

is the size of the second jump. Repeat this process until the whole deck is turned over, noting the last landing place in the deck.

After a short period of contemplation, mark the last jump. Then ask a friend to pick a number from 1 to 10. Use that number as the size of the first jump on a second sequence of jumps through the deck of cards. If you are lucky, the sequence of jumps will end at the marked card.

The trick does not always work! It rather succeeds about 5 times out of 6. We shall see why it succeeds in the next section.

4 The Kangaroo Method of J.M. Pollard

We are given a cyclic group G, and elements $g, X \in G$. g is a generator of the group, and we wish to solve $g^x = X$ for integer x.

We shall do so, by observing the sequence of jumps of two kangaroos, a "wild" and a "tame" one, through the group G. Set up hash[1] function $h : G \to J \subset \mathbb{N}$, where J is a set of jump sizes. It is appropriate to take J to be a set of "baby" jumps and "giant" jumps, thus $J = \{1, 2, 4, 8, 16, \ldots, 2^j\}$. The maximal jump size should be much smaller than $\sqrt{\#G}$.

We start the "tame" kangaroo at a known value, say $a_0 = g$. It's first jump size is $h(a_0)$, so it lands at $a_1 = a_0 g^{h(a_0)}$. Inductively continue the tame kangaroo as

$$a_{m+1} = a_m g^{h(a_m)}$$

It is most important to observe that at the mth jump, the tame kangaroo is a known power of g! That is $a_m = g^{A(m)}$, where $A(m)$ is known. We let this kangaroo jump about $c\sqrt{\#G}$ times.

We start the "wild" kangaroo at X. Thus, $b_0 = X$. It then jumps just as above, so that

$$b_{n+1} = b_n g^{h(b_n)}$$

At the nth step, b is a known power of g times X. Thus, $b_n = X g^{B(n)}$.

Therefore, if at any stage the wild kangaroo lands at one of the sites visited by the tame kangaroo, we would have the equation

$$g^{A(m)} = X g^{B(n)}, \qquad \text{for some } m, n$$

[1] In computer science, a "hash" function is any function which assigns numerical values to it's argument. For instance, each character on the key board has an ASCII numerical value as it's hash.

Everything in this equation is then known, except for X. We have solved the Discrete Logarithm Problem.

We should argue that we solve the problem in about $c\sqrt{\#G}$ steps. Assuming that the tame kangaroo has jumped this many times, at any of the wild kangaroo's jumps, it has chance of about $c/\sqrt{\#G}$ to land on the a site visited by the tame kangaroo. If the jumps were statistically independent at each stage, we can easily estimate the chance that the wild kangaroo takes $\sqrt{\#G}$ steps and at each step misses the tame kangaroo sites. It is

$$(1 - c/\sqrt{\#G})^{\sqrt{\#G}} \simeq e^{-c}$$

If c is big (but very small compared to $\#G$!) we are quite likely to have solved[2] the problem.

Comparing this to the card trick we see that the first sequence of jumps through the deck of cards is that of the tame kangaroo. The average jump size is 5, so that the tame kangaroo visits about 10 sites in the deck. The wild kangaroo begins with the selection of jump size by a friend. We see that indeed it fairly probable that the two kangaroos will visit a common site, and thus finish at the marked card

5 Further Remarks, Annotated Bibliography

Cryptography, it's protocols and security, require sophisticated mathematics! Some references are: First, a recent undergraduate text by Crandall and Pommerance is:

R. Crandall and C. Pommerance, *Prime Numbers: A Computational Perspective.* Springer–Verlag.

This book is highly recommended. Richard Crandall has served as chief scientific officer for both Next and Apple computers. His knowledge of algorithmic aspects of numerical methods is quite extensive. Carl Pommerance is a gifted number theorist, and developer of one of the powerful modern methods of factoring integers. He currently works for Lucent. Both are skilled authors. This text is quite sophisticated, accessible and readable.

[2]After I gave this talk at the University of Arkansas, Rajit Chatterjea, an EE major, rightfully points out that this argument is not a proof that the algorithm solves the problem, but only a heuristic argument. In fact, a not too convincing heuristic. I couldn't agree more! But remember that the search space to solve the problem is huge! A common feature of algorithms that search such big spaces is a probabilistic aspect. While the heuristics to analyze such algorithms is not exact, experience has shown that the arguments work quite well. Finally, one can make a much better analysis of this algorithm. See the paper of D.J. Pollard referenced below.

Two other well known books are by Neal Kobolitz, one of the first to understand the applications of elliptic curves to cryptography. And at this point, elliptic curves provide the most secure protocols around.

N. Kobolitz. *A Course in Number Theory and Cryptography*, Graduate Texts in Math. No. 114, Springer-Verlag, New York, 1987. Second edition, 1994.
N. Kobolitz.*Algebraic Aspects of Cryptography* Algorithms and Computation in Mathematics Vol. 3, Springer-Verlag, New York, 1998.

There is also a vast amount of literature devoted more to the development of algorithms which implement cryptographic protocols. This literature should be consulted by any serious student in the subject. But the author doesn't have any solid recommendations myself on a book to consult. (Remember the author is not an expert!)

The card trick by Kruskal is described in an article by Martin Gardner. Everything that he has written comes with high recommendations.

M. Gardner. Mathematical games, Scientific American Feb. 1978, pp. 19—32.

J.M. Pollard has a more detailed study of the card trick in the first article below. The next two are articles of his devoted to the Kangaroo method.

J.M. Pollard, Kruskal's Card Trick, Math. Gaz. **84** (July 2000) 46—49.

J.M. Pollard, Monte Carlo methods for index computation (mod p), Math. Comp. **32** (1978) 918—924.

J.M. Pollard, Kangaroos, Monopoly and Discrete Logarithms. J. Cryptology **13** (2000) 437—447.

The reader who is new to the subject might rightfully object that our argument that the Discrete Log Problem can be solved in $\sqrt{\#G}$ steps was probabilistic. There is no guarantee the algorithm stops in that many steps. Indeed, it is so. A feature of this problem is that the search space is huge. In typical applications, the order of the group can be 2^{1028}. Probabilistic methods are frequently employed in such problems as a way of specializing to the "typical" case of the problem.

Computational aspects of the Kangaroo methods are dictated by the issues of time and space. Time is controlled by the number of computations needed to solve a problem, and space is dictated by the amount of memory needed to carry out the computations.

Remarkably, the Kangaroo method can be implemented in ways that require only a constant amount of space, and in time that is given by $\sqrt{\#G}/P$ where P is the

number of parallel processors. Nevertheless, there are still some unresolved aspects of the algorithm and it's analysis. See the last article by Pollard for a guide to these issues.

Finally, it is known that the Discrete Log Problem is known to be hard in the average case iff it is hard in the worst case. Here "hard" can be understood to mean that there is an $\varepsilon > 0$ so that solutions will take $(\#G)^{\varepsilon}$ time or space to solve.

Before you seek fame and fortune by developing an argument to show that the Discrete Log Problem is indeed hard, consult the remarkable paper below. It shows, in section 4, that any such argument will have be quite "unnatural."

A.A. Razborov and S. Rudich, Natural Proofs, J. Comp. Sys. Sci. **55** (1997) 24—35.

These notes were prepared for a talk given at the University of Arkansas at Fayetteville, April 2002.

Michael T. Lacey
School of Mathematics
Georgia Institute of Technology
Atlanta GA 30332
lacey@math.gatech.edu
http://www.math.gatech.edu/~lacey