Number Theory in Cryptography

Jeremy Booher

July 21, 2008

Modern cryptography attempts to solve problems of security in an electronic age. It solves important problems like how to communicate secretly, verify identities, and setting up verifiable secret computations. After all, surely you'd like to be able to purchase a copy of Hardy and Wright online at the end of PROMYS without having your credit card number stolen. Cryptography provides mathematical solutions drawing on number theory and abstract algebra.

1 Public Key Encryption

When buying Hardy and Wright, Alice needs a way to send information (the credit card number) in such a way that only Bob, the intended recipient, can access it. The simplest way to do so is to use public key cryptography, which eliminates the need for secret, previously agreed-upon information.

Public key encryption refers to methods that use a publicly available key to encrypt a message, while the user keeps a separate, private key secret that is required to decrypt the message. The standard situation is that Alice, trying to send a message to Bob, uses the public key to encrypt her message. Bob then uses the private key to decrypt the message. Ideally, Eve, who eavesdrops on the encrypted message, gets no useful information about the message. The most widely used and studied public key system is RSA. Therefore, let's study Rabin encryption, which like RSA makes heavy use of modular arithmetic.

1.1 Rabin Encryption

To set up Rabin encryption, Bob picks two large prime numbers p and q, both of which are congruent to 3 mod 4. He publishes $n = pq$, the public key. p and q are the private decryption keys, so keep them secret.

For Alice to send a message M, it needs to be an integer satisfying $0 \leq M < n$. (If the message is text, it can be converted into a number using ASCII and then split

into digit blocks of the appropriate size.) Alice then calculates $E(M) = M^2 \mod n$, and sends $E(M)$ to Bob.

Bob can decrypt the ciphertext $E(M) = C$ by finding the square roots of C. Since C is a square, Bob knows from number theory that $C^{\frac{p-1}{2}} = 1 \mod p$. Write $p = 4k - 1$. Then $C^{2k-1} = 1 \mod p$, and $(C^k)^2 = C \mod p$. Therefore, C^k is a square root of $E(M)$ modulo p. $-C^k$ is the other. Doing the same thing modulo q, Bob uses the Chinese Remainder Theorem to find the four square roots of C modulo n. One of these is the message which Alice sent him. However, this poses a problem: how does Bob tell which square root is the correct one? Without help, he can't. There are various clever ways to solve the disambiguation problem. The simplest involves padding the message with a prearranged sequence of bits. For example, Alice and Bob agree that at the start of every message they'll put the word PROMYS. (So instead of saying "Hello Bob", the message would be "PROMYSHello Bob". When Bob sees all four square roots, it's unlikely that any of the 3 extraneous square roots will start with PROMYS, so Bob can figure out which of the four choices is Alice's message. However, this complication means that RSA encryption is more often used in practice, despite it requiring substantially more modular arithmetic than Rabin encryption.

The other important question is whether Eve can read Alice's message. In theory she can: all she needs to do is factor n, find its two prime factors, and follow the same procedure Bob did. However, if both p and q were chosen to have 800 digits each, the fastest known methods of factoring n would require centuries. Researchers have been searching for efficient factoring algorithms for centuries, and the fastest still essentially use (clever) brute force. Therefore, cryptographers (and companies and governments) agree that factoring n to decode the message is not practical and not a threat.

Suppose Eve had a magic algorithm that decrypted messages encrypted using Alice's public key n. Using this, Eve can factor n. To do so, Eve picks a random integer x from the range $[0, n)$. Then she calculates $x^2 \mod n$, and uses the decryption algorithm to find a square root of $x^2 \mod n$. Since x was chosen randomly, there is a one-half chance that the decryption algorithm produces $\pm x \mod n$. In this case, just try again. Otherwise, Eve has x and y with $x^2 = y^2 \mod n$ and $x \neq \pm y \mod n$. Then $(x - y)(x + y) = 0 \mod n$, but neither factor is 0. This means that $x - y$ or $x + y$ is a multiple of p, while the other is a multiple of q. Using the Euclidean algorithm (which is very efficient) produces the greatest common factor of $x - y$ and n, which is one of the prime factors. Therefore, Eve can factor n if she can break Rabin encryption. Conditional on the widely believed fact that factoring is hard, Rabin encryption is unbreakable. This is in contrast to the situation with RSA, for which no one has published a way to break it but no one can prove how secure it is.

1.2 Paillier Encryption

Rabin encryption is nice for many purposes. However, suppose Alice wanted to vote electronically. If she simply encrypts her vote, Eve can just try encrypting a vote for Hardy and a vote for Wright and check which of the two Alice voted for. Thus, even though the message is encrypted, information leaks out because the range of possible messages is so limited. One fix would be to append a random string (a nonce) to the start of the message. A nicer one is to use Paillier encryption.

To set up Paillier encryption, Bob picks two large primes p and q such that $(pq, \phi(pq)) = 1$. The public key is $n = pq$. Note that $\phi(n) = (p-1)(q-1)$ and $\phi(n^2) = pq(p-1)(q-1) = n\phi(n)$. In particular, for $a \in U_{n^2}$, $a^{n(p-1)(q-1)} = 1 \mod n^2$ by Euler's theorem.

Alice encrypts a message M (an integer between 0 and n) by first randomly selecting r from U_{n^2}. The encryption is $E(M, r) = (1 + Mn)r^n \mod n^2$. Note that because of the random unit, it is no longer possible to decrypt messages when the space of possible messages is constrained.

Bob decrypts $C = E(M, r)$ by first raising C to the $\phi(n) = (p-1)(q-1)$ power. Since $((1 + Mn)r^n)^{\phi(n)} = (1 + Mn)^{\phi(n)} r^{n\phi(n)} = (1 + Mn)^{\phi(n)} \mod n^2$, the random r goes away. Using the binomial theorem, $(1 + Mn)^{\phi(n)} = 1 + Mn\phi(n) + (Mn)^2 \frac{\phi(n)\phi(n-1)}{2} + \ldots = 1 + Mn\phi(n) \mod n^2$, where all the remaining terms disappear modulo n^2. Therefore, given C, Bob can calculate $1 + Mn\phi(n) = y \mod n^2$. But subtracting one and multiplying by $\phi(n)^{-1}$ (which exists since $\phi(n)$ and n are relatively prime), Bob knows $Mn = (y-1)\phi(n)^{-1} \mod n^2$. Therefore, he knows M modulo n, which suffices to uniquely recover the message.

Can Eve decrypt the message without the knowledge of the factorization of n? She can't easily calculate $\phi(n)$ without knowing the prime factorization, since if she directly counts the number of integers less than n and relatively prime to n, the first one that fails will give a factor of n. A more careful analysis shows that breaking Paillier encryption would allow taking roots modulo n, which is believed to be as hard as factoring.

Paillier encryption is called homomorphic because $E(M_1, r_1) \cdot E(M_2, r_2) = E(M_1 + M_2, r_1 r_2) \mod n^2$. Unwrapping the definitions,

$$\begin{aligned} E(M_1, r_1)E(M_2, r_2) &= (1 + M_1 n)r_1^n (1 + M_2 n)r_2^n \\ &= (1 + M_1 n + M_2 n + M_1 M_2 n^2)(r_1 r_2)^n \\ &= E(M_1 + M_2, r_1 \cdot r_2) \mod n^2 \end{aligned}$$

In other words, it is possible to produce an encryption of $M_1 + M_2$ knowing encryptions for M_1 and M_2 but not the actual messages. This could be useful in electronic voting. A voting machine would encrypt a vote, where it could be sent to a central

collecting point and added to the running total without determining the actual vote and violating the voter's privacy. Only after the election would the private keys be released and the totals decoded. In an election, this malleability is useful, although in general it can be worrying. An adversary could produce an encryption of $M_1 + M_2$ with no knowledge of M_1 or M_2 just by multiplying the encryptions, which in a few situations is dangerous. For sending text, this doesn't matter, since if M_1 is a number representing a string of text, it's unlikely M_1+M_2 will be a meaningful sentence unless M_2 was chosen with knowledge of M_1.

2 Secret Sharing

Another use of cryptography involves sharing a secret among multiple parties. Suppose Dan wants to place a bid in a secret auction on a copy of Hardy and Wright, but he doesn't trust the auctioneer. Dan thinks the auctioneer will give his bid to Eve, so that she can overbid him by one cent. Dan does trust that no more than 5 of the 11 people involved in the auction will talk to Eve, though. Is there a way to distribute his bid so that 6 of the people can reconstruct it but no 5 of them have any information about it?

More generally, (n, k) secret sharing is the problem of distributing a secret number s among n people so that no $k-1$ of them have any information about s but k of them can determine s. Shamir's secret sharing does this by giving the n^{th} party $f(n)$, where f is an appropriately chosen polynomial. The dealer Dan picks random field elements $a_1, \ldots a_{k-1}$, and uses the polynomial $f(t) = s + a_1 t + a_2 t^2 + \ldots + a_{k-1} t^{k-1}$. He gives the n^{th} person $f(n)$. For k people to recover the secret, they just pool their shares and use Lagrange interpolation to find the unique degree $k-1$ polynomial passing through the k points. (Lagrange interpolation works over any field. In practice, a large finite field would probably be used.) The secret is just the constant term.

Any $k-1$ people have no knowledge about s. They have $k-1$ points on the polynomial, but there is a degree $k-1$ polynomial going through their $k-1$ points and $(0, t)$ for any integer t, so their combined knowledge reveals nothing about s.

3 Zero Knowledge Proofs

Finally, Peggy would like to prove her identity to Victor. After doing so, Eve should not be able to use the information in Peggy and Victor's conversation to impersonate Peggy, and Victor needs to be almost certain of Peggy's identity (in any conversation, there is a non-zero chance that the impostor could randomly choose the correct responses to the verifier's queries, so we only require that the probability of error be arbitrarily small.)

Peggy and Victor can accomplish this through a zero knowledge proof: Peggy proves that she knows a secret associated to her without giving Victor (or Eve) any information beyond the fact that she knows it. This protocol allows Peggy to identify herself securely, and it can also be modified into a static cryptosystem to allow Peggy to sign messages.

Peggy picks a large value of n for which it is hard to compute square roots (for example, a product of two large primes as in the Rabin cryptosystem). She chooses a secret s_a in U_n. She publishes $s_a^2 \mod n$ along with the modulus n, which serve as her identity.

Peggy can prove her identity to Victor through a series of repeated conversations. In each round, Peggy begins by picking a random z_1 from U_n, and sending Victor $z_1^2 \mod n$. Victor chooses to ask for z_1 or $z_1 s_a \mod n$ at random. Peggy then sends Victor the quantity he requested. He checks that the square of Peggy's reply matches $s_a^2 z_1^2 \mod n$ or $z_1^2 \mod n$, both of which he can calculate from the public information s_a^2 and z_1^2. Peggy passes the round if this matches. They then repeat this until Victor is satisfied with Peggy's identity.

There are three properties to check: that this is complete, sound, and zero-knowledge. To be complete means that Peggy can always pass this test. Being sound means that if Eve passes the protocol with no prior knowledge gained by eavesdropping, Eve can find a square root of s_a^2 modulo n, which was assumed to be hard. Finally, to be zero knowledge means nothing can be learned by eavesdropping. In other words, the proof doesn't give anything away.

This is complete since Peggy can easily meet either of Victor's requests as she knows both s_a and z_1. It is also sound. In order to always pass the protocol, the prover needs to be able to produce a square root for both z_1^2 and $s_a^2 z_1^2$ modulo n when challenged at random by Victor. But if the prover knows both of these, then he can calculate $\pm s_a \mod n$ through simple modular arithmetic. Therefore, an impostor has at most a $\frac{1}{2}$ chance of passing one round of the conversation. If Victor wants the probability that an impostor passes the test is less than ϵ, he simply requires that Peggy participate in n rounds, where $2^{-n} < \epsilon$.

Finally, is this zero-knowledge? To be zero knowledge means that Peggy doesn't give any information away by participating in the protocol. To show it is, it suffices to show that Eve can simulate conversations with the same probability distribution as valid conversations between Peggy and Victor. If she can do this, then she gains no information if Peggy and Victor actually use the protocol, since she can produce sample conversations on her own instead of bothering to listen. Eve can simulate a round easily: she first decides whether Victor will ask for $z_1 s_a$ or z_1 at random. In the first case, she decides the first message will be $x^2 (s_a)^{-2} \mod n$, where she picks x at random. Then the third step of the protocol has the prover give the verifier a square root for $x^2 (s_a)^{-2} s_a^2$, which because of her choice of message is just x, which

Eve knows. If she decided on the second option (that the verifier demands z_1) she just selects x at random, lets the first message be $x^2 \mod n$, and lets the provers reply be x. In either case this is a valid round. Since she randomly selects Victor's response, this aspect has the same probability distribution as a real conversation. She also randomly selects x from U_n, so both x and xs_a are random variables from U_n. In a real conversation both Peggy and Victor are choosing things at random as well, so Eve's simulation has an identical probability distribution to real conversations. Since she can simulate rounds alone, the protocol is zero knowledge.

4 Further Cryptography

There are many more interesting topics in cryptography, including verifiable secret sharing, key exchange, oblivious transfer, and digital signatures. In addition, these can be combined into complicated systems to run secure actions, conduct electronic voting, or create anonymous digital cash. Two nice sources, available freely online, are "A Taste of Elliptic Curve Cryptography"[2] and the "Handbook of Applied Cryptography"[1].

References

[1] Vanstone Menezes, Oorschot, *Handbook of applied cryptography*, CRC Press, 2001.

[2] Shrenik Shah, *A taste of elliptic curve cryptography*, The Harvard College Mathematics Review (2008).