Cryptology & Applied Cryptography

Evan Wong, University Honors in Mathematics *Dr. Joshua Lansky*

Spring, 2009

I should be able to whisper something in your ear, even if your ear is 1000 miles away.

~Philip Zimmermann

Ipsa scientia potestas est.
~Sir Francis Bacon,
Meditationes Sacræ. De Hæresibus(1597)

If Francis Bacon's famous aphorism holds, it follows quickly that secret knowledge has a special potency. Secrets being in their own way as valuable a commodity as silver and gold, humans have crafted the means both to protect and to plunder them since earliest civilization. We inherit this effort today as the discipline of cryptology, the study of secrecy systems. Once labor intensive and requiring the detailed attention of expert individuals, cryptology is traditionally the province of the political and social elite – those individuals with free time enough to study and information worth spending much effort to protect.

In the second half of the 20th Century, however, the advent of cheap computing power changed this dynamic profoundly. Mass cryptography became feasible, and by the end of the century the American public took for granted that an average individual could easily record or transmit a message without significant concern that it would be read by an unintended recipient. This has had a major impact on the way governments conduct business, but the truly fundamental impact of freely available security measures is that the communications that are intrinsic to globalization can be made securely. While previously, there had been no way of transmitting information privately to a stranger, the advent of public key cryptography in the 1970s allowed for the commercial development of the internet, among a multitude of other benefits provided by secure communication.

This paper will examine the key aspects of modern cryptography that allow such an assumption to function and describe in a simple way the actual implementation of a modern secrecy system. First, however, some terms should be defined:[1]

- **Cryptography** is the creation of secrecy systems, methods of securing information against unwanted examination.
- **Cryptanalysis** is the breaking of secrecy systems. It aims to overcome the efforts of the cryptography systems.
- **Cryptology** is a blanket term encompassing both cryptanalysis and cryptography. It is used because effective cryptographers must practice as cryptanalysts, and vice versa.
- **Alice** and **Bob** refer to the two hypothetical individuals who might be using a cryptographic system to communicate securely; usually Alice is said to be sending Bob a message, though their situation is reversible. **Eve** is an eavesdropper who is able to listen to messages Alice sends Bob, thus necessitating the use of encryption so as to be confound to Eve.
- **Plaintext** is whatever unencrypted message Alice wishes Bob to receive. The **alphabet** is the set of characters used in creating a plaintext message (usually, this includes at least the 26 letters of the alphabet, plus the space ' ' symbol). All the possible messages Alice might create by combining characters of the alphabet constitute the set of plaintext.

[1] Technical definitions taken from Koblitz (1994). The placeholder names Alice, Bob and Eve are common convention among cryptologists.

- **Ciphertext** is an encrypted string of text Alice transmits to Bob, which is assumed to be intercepted by Eve but unintelligible to her. Usually, it is written in the same alphabet as plaintext.
- **Encryption** and **decryption** are algorithmic functions performed on plaintext and ciphertext, respectively. Encryption functions are one to one, from the set of plaintext to the set of ciphertext, while decryption functions are from the ciphertext to plaintext. Each encryption function has a related decryption function, which acts as its inverse. That is, if Alice performs an encryption function 'E(p)' on a string of plaintext 'p' and transmits the resulting ciphertext 'c' to Bob, he can perform the related decryption function 'D(c)' and obtain the original message 'p' in turn.
- A **cryptosystem** is a general method defining an alphabet (and therefore the allowed plaintext and ciphertext) and encryption and decryption functions. When Alice and Bob communicate, the cryptosystem they are using is usually public information, available to Eve.
- A **key** is a unique piece of information that Alice and Bob share in order to implement a cryptosystem. The security of the communication comes from the secrecy of the decryption key, which only Bob (and possibly Alice) have access to, and which is necessary to performing the decryption function on any given ciphertext. Generally, if Eve is to successfully cryptanalyze any particular conversation between Alice and Bob, she must somehow obtain Bob's decryption key.[2]

Computing Power

Computers in the future may
Have only 1,000 vacuum tubes
And weigh only 1.5 tons.
~ **Popular Mechanics**, March 1949

Cryptology before the 20th Century was a labor intensive process. Both encryption and decryption generally involved repetitive mathematical calculations, making long or descriptive messages undesirable, even apart from the vulnerability of wordiness to cryptanalysis. Cryptanalysis itself was similarly subject to the limitations of human endurance, making data encryption reasonably secure, but the amount of work invested in any particular secured message greatly limited how many people could afford to send information securely. Certainly, an average American of the 19th Century could not entrust their private banking information to the mail system unencrypted, nor would it be economically feasible for her to establish a cryptosystem with a company supplying a mail order catalogue, so the figure of the travelling salesman predominated as a mean s of overcoming this difficulty.

However, because encryption generally entails algorithmic (i.e. precisely defined and unambiguous) operations, it is highly amenable to being implemented in computer code. The development of electric and semi-electric computing devices had a major influence on the course of the

[2] The requirement that the only information that absolutely must remain secret should be the decryption key is called Kerckoff's Principal, for the Dutch linguist and cryptographer of that name who first stated it.

Second World War, in the form of the German cryptosystem implemented by means of the Enigma machine and the Allied cryptanalysis effort codenamed ULTRA, in addition to numerous less infamous instances.[3] After the war, the dual trends of increasingly powerful computer hardware and increasingly efficient computing algorithms gave a significant edge to cryptologists using more advanced technology – cryptographers could implement more computationally intense systems, while cryptanalysts could more easily conduct brute force attacks on systems (that is, attempt to simply guess all the possible keys to any given cryptosystem by having a computer run through them relatively quickly). This was (and remains) a major motivator for technological development during the Cold War, and is the *raison d'être* of organizations such as the National Security Agency, which acts as the cryptological arm of the US government, [4] as well as the National Institute of Standards and Technology, which purports to track the security of various cryptosystems for the public good.[5]

Despite the greater – and even indispensible – role played by computers in modern cryptology, the discipline remains fundamentally outside the realm of computer science. This is because despite the wonderful speed and efficiency with which computers perform calculations, the operations they perform are but abstract representations of physical or mathematical analogues. In other words, a program is able to implement a hypothetical cryptosystem that would not otherwise be viable, but no program will achieve anything resembling encryption without being based on a mathematical or physical cryptosystem. This is not to say that computers and computer science has not impacted the evolution of cryptology; in fact, the opposite can quite easily be shown. Primarily, the rise of the cheap computer has encouraged mathematical underpinnings for modern cryptosystems, as opposed to certain physical[6] or linguistic models of previous eras.

Mathematics

Linear improvements in computer power can't stand up
to exponential improvements in difficulty.
~Unknown

Today, major advances in cryptology are usually the result of mathematical innovations, and modern cryptographers use primarily mathematical methods in developing new systems. In particular, the study of prime numbers and fields has been of particular importance to the development of cryptography.

Historically, all cryptosystems used what is call a symmetric key paradigm – that is, both Alice and Bob would use the same key to encrypt their messages, and this encryption key would immediately suggest the appropriate decryption key (and vice versa). For example, the encryption key {a->b, b->c, …, z->' ', ' "-> a} immediately suggests the decryption key {a->' ', b->a, …, z -> y, ' '-> z}. This allows for a very simple system, but presents two difficulties: first, that the encryption keys must be kept secure,

[3] Kahn, 1991.
[4] National Security Agency, Central Security Service
[5] National Institute of Standards and Technology, Computer Security Division
[6] The theorized ancient Greek scytale is an example of this.

and second, that since both Alice and Bob use the same keys, both Alice and Bob must somehow agree on a key. This is known as the key distribution problem, because without a preexisting secure channel (the lack of which being what necessitates cryptography in the first place), there is no way for Alice and Bob to establish a key without Eve listening in on their original key negotiation conversation.[7] Two solutions to this seeming conundrum presented themselves in the 1970s. In 1976, Whitfield Diffie and Martin Hellman published an article detailing a procedure that would allow Alice and Bob to determine a symmetric key securely while communicating in an unsecured channel. In 1978, Ron Rivest, Adi Shamir, and Leonard Adleman of MIT published an article that detailed the first widely acknowledged public key cryptosystem, obviating many of the disadvantages of the private key system.

The Diffie-Hellman key exchange protocol makes use of certain characteristics of finite fields – a class of abstract constructions in algebra and number theory. Alice, before initiating communication with Bob, must first pick a finite field to work in (which amounts to picking a large prime number, 'q'), an element of that field (i.e. a smaller number, 'g'), and a secret exponent (another smaller number, 'a'). Alice would then compute 'g^a modulo q' (that is, the remainder of 'g' raised to the power 'a' and divided by 'q'), and transmit 'q' 'g' and 'g^a mod q' to Bob. Bob in turn chooses a random number 'b' and computes 'g^b mod q,' transmitting the result back to Alice. Both Alice and Bob are then able to compute 'g^{ab} mod q' (Alice by computing '$(g^b)^a$ mod q' and Bob by computing '$(g^a)^b$ mod q'), which becomes the private key they use to establish a secure line of communication. Eve, in the meantime, is assumed to have intercepted 'q,' 'g,' 'g^a,' and 'g^b.' However, one characteristic of finite fields is that as of this writing there is no known method of reliably computing 'g^{ab}' given only 'g,' 'g^a,' and 'g^b.' The assumption that 'g^{ab}' is difficult to compute from this information is known as the discrete logarithm problem (i.e. computing the logarithm 'a' of 'g^b' in the "discrete," or finite, field of 'q' elements). Thus, Alice and Bob have agreed at a number, 'g^{ae},' which is indecipherable to Eve, and therefore suitable for use as a key for a traditional symmetric key cryptosystem.[8]

The algorithm published by Rivest, Shamir and Adleman (referred to as "RSA") is slightly less elegant than the Diffe-Hellman protocol, but has the advantage of being a genuine public key cryptosystem. Prior to any correspondence, Bob is supposed to have published his personal "public key," chosen by following this procedure:

1) Bob picks a large number 'n' that is the product of two distinct primes, 'p' and 'q.'
2) Bob computes the Euler phi function of n, $\phi(n) = (p - 1) * (q - 1)$
3) Bob picks some number 'e' which is both less than 'n' and relatively prime to $\phi(n)$.
4) Bob calculates a value 'd,' which is the multiplicative inverse of 'e modulo $\phi(n)$'
5) Bob publishes his encryption (or "public") key, "$E_B(n, e)$"
6) Bob stores his private decryption key, "$D_B(n, d)$"

When Alice wishes to send a secure message to Bob, her first task is to assign a number value to whatever plaintext she wishes to send (this is generally done by converting "a" into the numeral "0," "b" to "1," and so on). Having done this and achieved a plaintext number 'p,' Alice uses the fact that 'n' and 'e' are publically known to compute 'p^e modulo n,' which she transmits to Bob. Bob, in turn, computes '$(p^e)^d$ modulo n,' which is the same as 'p^1 modulo n,' by Euler's theorem. Thus, Bob has calculated

[7] Koblitz, p84
[8] Koblitz p99

Alice's plaintext number, while the only values Eve has been able to intercept are 'n,' 'e,' and 'p^e,' which is insufficient to calculate 'p' for the same reasons that made 'g^{ab}' inscrutable in Diffe-Hellman. [9]

These descriptions are of necessity incomplete. In particularly, the importance of the Euler phi function and Euler's theorem are neglected, as the existence of 'ϕ(n)' is what allows the calculation of 'd' from 'e' and 'ϕ(n).' This calculation in turn relies on the extended Euclidean algorithm, and implies that 'd' is easily calculable by Eve if only she could determine 'ϕ(n).' Because the Euler phi function of 'n' is a direct consequence of the factorization of 'n,' it follows that breaking any RSA encryption is at least as easy as factoring 'n.' Therefore, it is important that 'n' be sufficiently large to ensure a measure of security, as must be the other components of both the RSA and Diffe-Hellman algorithms.

It should be immediately apparently that computing power is essential to performing even a single practical demonstration of either Diffe-Hellman or RSA; taking high powers of large integers can be streamlined through clever mathematical techniques, but will always require a certain number of simple calculations that would nevertheless be tiresome to perform by hand. However, it should be similarly clear that the underlying principles at work in modern cryptography – of which these two algorithms are at least somewhat representative – are mathematical in nature. Appropriately, the most sought after cryptanalytic advances of today – breaking the Diffe-Hellman or RSA cryptosystems, for example – can also be reduced to mathematical challenges, such as solving the general discrete logarithm or rapidly factoring the product of large prime numbers. There is, of course, no guarantee that a solution for either or both of these problems will not be published tomorrow, or even that some solution has not already been found.

For nothing is secret, that shall not be made manifest;
Neither any thing hid,
That shall not be known and come abroad.
Book of Luke, Ch. VIII, v. 17

[9] Rosen p310-311

Implementation

When cryptography is outlawed, bayl bhgynjf jvyy unir cevinpl.
'When cryptography is outlawed, [only outlaws will have privacy].'
~Anonymous, using an ROT-13 encryption

The second half of this project consists of three implementations of cryptographic algorithms. All three are written in the Java programming language, interface with the user via the command line, and are intended for demonstrational rather than practical purposes. I have tried to include significantly more than normal amounts of commenting, so that examining the code of each program will give the reader a good idea of the actual principals at work.

The first program, DiffeHellmanExchange, is surely the simplest, and merely implements the most straightforward interpretation of the Diffe-Hellman protocol I could think of. Two users are each assumed to have the same program at different locations, as well as an insecure channel through which they can communicate. Either Alice or Bob. can pick the values for the modulus and the base, both pick their own exponentiation values, and relevant data is exchanged. Finally, the program computes the private key Alice and Bob will share and declares it to both users.

The second set of programs, RSA Key Exchange, is one possible implementation of a key exchange using RSA rather than Diffe-Hellman protocols. Whichever user (I assumed it to be Alice) wishes to initiate correspondence generates and publishes her private key using the program AliceRSAKeyGen, after which they send their target a request to generate a private key to share. Bob, receiving this request, uses BobRSA to input Alice's public key information, generate a private key for Alice and himself to share, and encipher and transmit that key. Alice would then use AliceDecriptionRSA to input the ciphertext sent by Bob, resulting in both parties sharing the same secure private key for their continuing correspondence.

The final set, Affine Transformation System, implements a primitive cryptosystem known as an affine transform – one of many private key systems that lend themselves to computer implementation. Alice and Bob are assumed to have previously established a secure channel and private encryption key (perhaps by using one of the previous programs). When Alice would like to send Bob a message, she runs SallyAffineEncryption, which prompts her to enter her plaintext as well as the parameters of the cryptosystem she shares with Bob, and generates a string of ciphertext to transmit. Bob, in turn, would run RichardAffineDecryption with the same parameters to decipher Alice's message. The major feature of this program is that is implements an affine transformation on digraphs rather than on single characters, making it significantly more secure than the simplest cryptosystems, though still unsuitable for practical use. It is included in this project as a demonstration of a private key system.

Works Cited

Diffie, Whitfield and Martin E. Hellman. "New Directions in Cryptography." IEEE Transactions on Information Theory. Vol. IT-22, No. 6. Pp. 644-654. November, 1976. Accessed 3rd May 2009. <citeseer.ist.psu.edu/diffie76new.html>

Kahn, David. Seizing the Enigma: The Race to Break the German U-Boats Codes, 1939-1943. Houghton Mifflin, 1991. ISBN 0 395 42739 8.

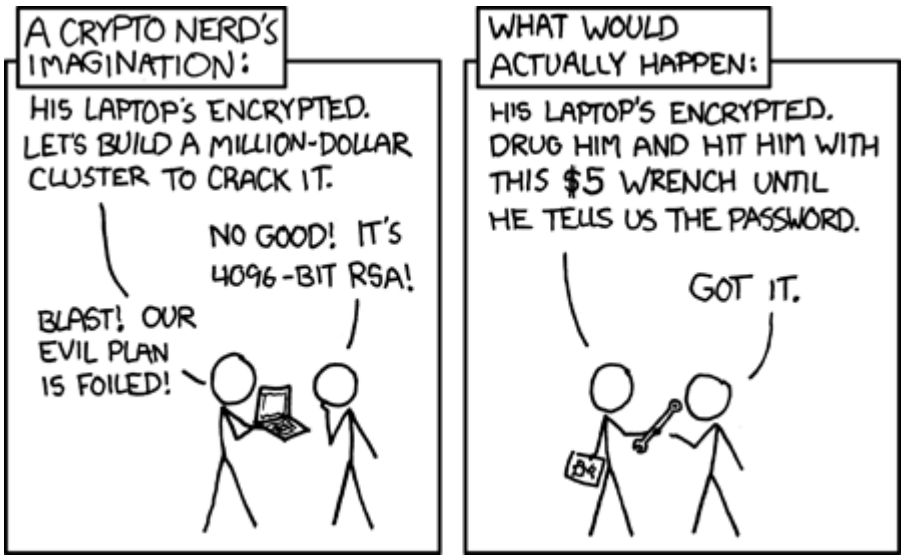
National Institute of Standards and Technology, Computer Security Division. "About CSD." Accessed 3rd May, 2009. <http://csrc.nist.gov/about/index.html>

National Security Agency, Central Security Service. "About NSA." Accessed 3rd May, 2009. <http://www.nsa.gov/about/index.shtml>

Neal Koblitz. A Course in Number Theory and Cryptography. Springer Science+Business Media, Inc. 2nd ed. Sept. 1994.

Rosen, Kenneth H. Elementary Number Theory and Its Applications. Addison-Wesley, 2004.

Rivest, R.L., A. Shamir and L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems." Communications of the ACM. Vol. 21. Pp. 120-126. 1978.



www.xkcd.com