A Brief History of Cryptography

· Most of cryptography stemmed from two ideas transposition and substitution.

· Substitution is simply substituting letters, words or parts of words with other symbols. For example, a Caesar Shift is a cipher in which each letter is replaced with another letter x places from it. Where x is a number between 1 and 25.

· Transposition is simply mixing up the letters, words or parts of words in your message.

· These ideas and the combinations of them all involve a symmetric key. This is the idea that what algorithm you use to encrypt the message, you use that same algorithm in reverse to decrypt the message.

· The problem of key distribution then arises. If two parties, Alice and Bob want to communicate secretly they must either first meet so they can decide on a key, or they must have a third party transfer the key between them. If the key then falls into the wrong hands they are unable to communicate secretly.

· The first attempt to combat this was the Diffie-Helman key-exchange. We will describe the mathematics of this later, but the idea is Alice and Bob are able to exchange information over public lines of communication and agree on a key from that information. And if a third party, Eve, is listening she will not be able to discover the key.

· Later came the RSA encryption, in which Alice has a separate public key and private key. Anyone can look up her public key. Bob can encrypt a message using Alice's public key, but he cannot use the same information to decrypt the message. To decrypt the message one must have special information contained in Alice's private key.

Some More Number Theory

Theorem: xy = x (mod pq), where p,q are primes, if and only if
xy = x (mod p) and xy = x (mod q).

Proof:

· If xy = x (mod p) then xy = mp+x
· and if xy= x (mod q) then xy= nq+x.
· Therefore mp+x = nq+x
· So mp = nq
· Since every integer has a unique factorization:
· mp and nq can be written as a product of primes.
· mp = m1m2p and nq=n1n2q
· Since mp = nq, they must have the same factorization.
· So there must exist an mi = q and an ni = p
· Let mp = rpq = nq
· So xy = rpq+x
· So xy = x (mod pq)
· Now assume xy = x (mod pq)
· Therefore xy = rpq+x = (rp)q+x = (rq)p+x
· It follows clearly that xy = x (mod q) and xy = x (mod p)

Euler Phi-Function

Previously defined Zm* = {aÎ Zm : gcd(a, m) = 1} is group under multiplication we call this group the prime residue group.

The Euler phi-function, f , for every nÎ Z+, is defined as f (n) is equal to the number of positive integers less than n that are relatively prime to n.

It follows clearly that |Zn*| = f (n).

Size of f(n):

Suppose n = p (where p is a prime).
- It is clear that f(p) = p-1

Now suppose n = pm
- The numbers that divide pm are of the form p, 2p, . ,p2, 2p2, .,pm
-There are pm/p=pm-1 numbers that are not relatively prime to pm
- It follows that f(pm)= pm -pm-1

I will not prove it, but if n=x1x2x3 and x1 , x2, x3 are relatively prime, then f(n)=f(x1)f(x2)f(x3).
Fermat's theorem states: If p is a prime and a is an element of Z+ and such that p does not divide a then ap-1=1 (mod p)
LaGrange's theorem says: if b is an element of Zn* then bf(n)=1 (mod n). This is true because f(n) is the size of Zn*.
RSA:
Encryption:
Step 1: Pick two giant prime numbers, p and q. Multiply the primes together to get a number, N.
Step 2: Choose an integer, e (it should be relatively prime to f(N)). e is called the encryption exponent.
Step 3: (e, N) is your public key.
Step 4: To encrypt your message you must first convert it to a number. This can be done using ASCII, each letter corresponds to a block of binary numbers. This binary string can then be treated as a decimal number.
Step 5: Use the function E(x)=xe (mod N) to encrypt your message.
Decryption:
Step 6: Find and integer, d (your decryption exponent) using the following:
ed=1 (mod f(N))
Step 7: (d, N) is your private key.
Step 8: Use D(x)=xd (mod N) to decrypt the message.
Verifying Encryption and Decryption are inverses:
We want to verify that D(E(x))=x
ed=1 (mod f(N))
ed=kf(N)+1
D(E(x))=(xe)d (mod N)=x kf(N)+1 (mod N)
Case 1: x is an element of Zn*
D(E(x)) = x kf(N)+1 (mod N) = (xf(N))kx (mod N)
By Lagrange's theorem xf(N)=1 (modN)
so (xf(N))kx (mod N)= (1)kx (mod N)=x(mod N)
Verify Encryption and Decryption are Inverses
We want to show that E(D(x))=x=D(E(x))
D(E(x)) = (xe (mod N))d (mod N) = xed (mod N)
= (xd (mod N)e (mod N) = E(D(x))
Note: xed = x (mod pq) if and only if xed = x (mod p) and xed = x (mod q).
If p divides x
then x = 0 (mod p), and xed = x (mod p)

If x is not congruent to 0 (mod p) (p does not divide x) then
xed = xtf (n) + 1 (mod p)
= (x(p - 1))t(q - 1)x (mod p)
= (1)t(q - 1)x (mod p) (Fermat's Theorem)
= x (mod p)
We can prove the same for q
So xed = x (mod p) and xed = x (mod q), so xed = x (mod pq)
Some notes on the security of RSA:
· There is no known way to find d other than knowing f(N).
· There is no quick and easy algorithm to factor large numbers into primes.
· The average computer would take around 50 years to factor a number on the order of 10130.
· For most banking transactions, N is of the order 10308.
· To factor something this size it would take a hundred million personal computers over one thousand years.
An Example of RSA:
· Alice picks p=17 and q=11 (in order for this to be secure these numbers should be enormous)
· N=pq=(11)(17)=187
· Alice now picks e=7
-note: f(N)=(p-1)(q-1)=10*16=160 and gcd(e, f(N))=1
· Next Alice finds d:
· d=7-1 (mod 160)
· d=23
· Alice's public key is (7, 187) and her private key is (23, 187)
· Bob wants to send Alice a message:
· His message is X, which is equivalent to 88 in ASCII
· E(88)=887 (mod 187)=11
· Now Bob sends 11 to Alice.
· Alice receives Bob's message and uses her private key to decrypt the message,
· D(11)=1123 (mod 187)=88
· Alice then uses ASCII to determine the integer,
· 88 corresponds to X.
Creating a Signature:
· Everyone has their own Public and Private Keys.
· Alice wants to send a message to Bob.
· For Bob to be sure that the message is from Alice she uses RSA to create a signature
· First Alice encrypts her message using her own private key
· Then Alice encrypts the result with Bob's public key
· After Bob receives the message he uses his private key to decrypt it
· Then he looks up Alice's public key and uses that to decrypt the message again.
· Since only Alice knows her private key, Alice must have done the encryption.
PGP-Pretty Good Privacy
· Developed by Phil Zimmerman
· PGP is a program designed to help the average person encrypt messages
PGP does the following:
1. Selects random primes and creates a public key and a private key
2. Uses RSA to encrypt a symmetric key (this is a faster method for encryption and decryption)
3. PGP will encode the message with a digital signature

Cracking the Code?

Tempest Attacks:

· Electromagnetic signals are emitted from in a computer's display unit.

· Eve can park her van outside Alice's house and with sensitive tempest equipment identify every keystroke.

· Eve has then intercepted the message before it is encrypted.

· There are materials that can be used to line the walls that prevent the escape of electromagnetic signals.

· You need a special permit from the government to buy such materials.

Trojan Horses:

· A program that looks like PGP or some other genuine encryption software.

· The program actually also sends plaintext copy to the programs designer.

· CRYPTO AG, a Swiss cryptography company, build backdoors into some of its products and sold the info to the US Government.

If quantum computers become a reality the time to factor N will decrease very drastically. RSA would no longer be a secure system.