Public Key Encryption.

Every "person" has two keys,
a <u>private</u> key and a <u>public</u> key.

person i : R_i (private), U_i (public).

Let P be any "message".
I'll use R_i and U_i for the
<u>keys</u> as well as the <u>transformations</u>.

$$P = R_i(U_i(P)) = U_i(R_i(P)).$$

(instead of $T_{R_i}(T_{U_i}(P)) = T_{U_i}(R_{U_i}(P))$.)

R_i : <u>secret</u> ; i keeps it.

U_i : <u>public</u> ; puts it in the NYT.
New York Times

Properties: given R, it is "easy" to find U.
given U, it is <u>hard</u> to find R.

Can that be done?

yes!

a few methods. A: Factorization.

(1) $R = (F_1, F_2)$

Two large primes.

say each $> 10^{70}$.

(2) $U = (F_1 . F_2)$ (product).

$(F_1, F_2) \rightarrow (F_1 . F_2)$ easy.

(< 1 sec. CPU time).

$(F_1 . F_2) \Rightarrow (F_1, F_2)$ hard

Few thousands of ~~seconds~~ years of CPU time.

But: a few years ago it was considered safe if $F_1, F_2 > 10^{60}$. no more!

Now: usually take $F_1 . F_2 > 10^{200}$.

RSA (Rivest, Shamir, Adleman) is based on the above. See Tanenbaum or see p 293 these notes.

~~A~~ B (discrete logarithms).

First ordinary logarithms:

$a^x = y$ given x, find y: easy.
given y, find x: $x = \frac{\log y}{\log a}$.
easy.

Now discrete logarithms.

$y = a^x \bmod p$ p: large prime.
$a \in \{$ ※2, ... $\}$.

Given p, a, x, find y: easy.
Give p, a, y, find x: can be hard.

(If you choose a right "primitive root of p".)
Don't worry

R: (p, a, x) (roughly) (Roughly!)
U: (p, a, y)

RSA:

How (F_1, F_2) translates into a Transformation U $(U(P))$,

and how (F_1, F_2) translates into a Transformation R $(R(P))$:

Not discussed here.

For a handwaving discussion: see Tanenbaum pp 752-754.

The primes F_1, F_2 have to be chosen in a special way.

Will Not be asked.

Modern Theory:

C. "Elliptic Curve based methods".

Don't worry

To send a secret message to A:

encrypt using U_A.

only A can decrypt (using R_A)

Authentication and Non-Repudiation:

A sends message, encrypted using R_A.

everybody can decrypt (using U_A).

must be from A

authentication, but NO confidentiality whatsoever!

How can we have both? Message P (Plaintext).

A → B.

① If A first encrypts using R_A ~~U_B~~, then again uses R_A U_B

$P = R_A U_B$

$\hat{P} = U_B(R_A(P))$

Nobody but B can find P:

$R_B(U_B(R_A(P))) = R_A(P)$

then

$U_A(R_A(P)) = P.$

privacy Ok.

Also: nobody but A could have produced $R_A(P)$:

authentication Ok

But: what about non-repudiation?

There is a problem!

in court (court of law, with judges and lawyers and a jury): B will have to give R_B to prove the message came from A.

Soon afterward:

"Everybody" knows R_B.

Everybody can make fake messages "provably" From B.

Everybody can decrypt recent messages encrypted by only U_B.

No Good.

---

(2) Better: A sends

$\hat{P} = R_A(U_B(P))$

check for yourself this is OK.

~~This~~ pp 278-280 only gave the basic ideas. A real system needs more bells and whistles.

Red system:

Protection against people falsely pretending to be somebody else, sending fake public keys.

Protection against "man in ~~de~~ the middle" attack.

Tanenbaum pp 955 - 959.

Will not be asked.

Message Digest.

P: Plaintext. Can be long.

Suppose we have a hash function H with the following properties:

- H(P) is fixed length (say 128 bits). independent of P.
- Given P, finding H(P) is easy.

(There are many different P_i with the same $H(P_i)$)

- Given H(P), it is very hard to find a Q with H(Q) = H(P).
- Even knowing P and H(P) and H(Q) and knowing that "Q is close to P" (but not knowing more about Q) it is hard to find an R with H(R) = H(Q). (and even harder to find Q exactly).

Then: H is a "Message Digest".

Use of Message Digest MD.

- Do invention.
- Write Patent Application P.
  (But you do not need to apply, yet).
- Compute MD(P).
- Put out in NYT:
  (MD(P) plus Money).

There is a company that does that for you.
"Time stamping".
& MD. Also used in
Digital Signatures.

~~MD5~~ Message Digest Algorithms:

~~281~~
~~282~~
284

Message digest Algorithms.

(1) MD4 (was broken)

(2) MD5 (128 bit digest).
Tanenbaum p 760.

(3) SHA-1 (160 bit digest)
Tanenbaum p 761.

Neither will be asked.