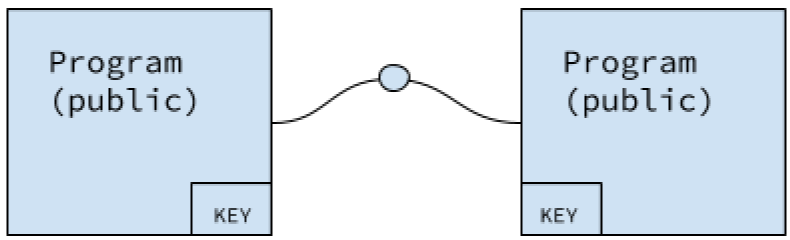
20. Encryption

When we send data over an untrusted network, we need to encrypt the data!
We design our system according to Kerchoff's Design Principal
~ minimize what needs to be kept secret

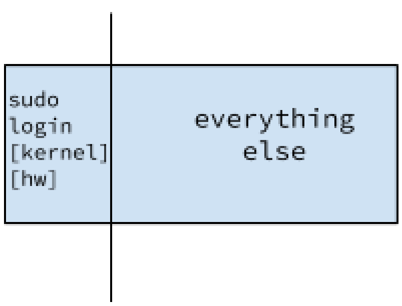


We denote the shared secret key with "K"

The question of security is a question of trust.

- Most programs run by the OS are not trusted (cat, grep, sh, gcc, make...)
- BUT a few are (login, sudo, su...), and can therefore call setuid()

If we are worried about security, the latter are the risk:
How do we choose who to trust?



- We call the set of all trusted programs the Trusted Computing Base
- We seek to minimize the TCB

- BUT can we really trust anything?
  Take Ken Thompson, creator of Unix:
  - wrote a Turing Award winning paper *Reflections on Trusting Trust*
  - paper described a bug he built into the system

```
// in the login program, Ken added the following code
if(strcmp(user,"ken")==0) uid = 0;
// however, this would be easily found out, so in gcc, he adds:
if(compiling(login.c)) generate strcmp
```

```
// now no one looking at the login source code can see the bug
// he also adds in gcc
if(compiling(gcc.c)) generate generate strcmp
// he compiles gcc and edits the code back -- the code is now gone!
```

Couldn't this be found in the assembly?

NO:

- he can modify the code for objdmp
- he can even make it so that using other code injects bugs!

The moral of the story is, "You need to trust your tools"

- as an aside, gcc really should be in the TCB

With this principal in mind, we model secure message passing:

A → B {"I'm Alice"}^K

- this is BAD; it is subject to replay attacks
  → we need a nonce ~ a nonsense string used to verify possession of a key

A → B "I'm Alice"
B → A "<nonce>"
A → B {"<nonce>"}
B → A "OK"
We can now effectively verify identify, so how do we use this to send messages?

- we use a Hashed Message Authentication Code

A → B M || HMAC(M, K)

- how do we find an HMAC?
  - we call our cryptographic friends for a (Cryptographically) Secure Checksum Algorithm
    - defined st if SHA(M) = N, it is hard to find M' st SHA(M') = N
  - we let HMAC(M, K) = f(SHA(K||M))

Now we are 100% secure, right?

- Of course not: SHA's can be broken — we are already on SHA 2.something

Distributed security can also be done with a virtual network inside of one Computer:

Typical authentication via ssh is multiphase
1. establish connection with public/private key encryption
2. use the encrypted nonce as a shared secret key

Shared secrets are the model we used above; the public/private model is as such:
- a message A → B is encrypted with B's public key
- person A can then decrypt the message with their private key

So it is better to use this first, since it is slower but more secure
- note: verifying a public key is not 100% exact

This model is utilized by the RSA encryption algorithm
- The keys for the RSA algorithm are generated in the following way:
    - Choose two distinct prime numbers p and q.
        - p & q should be chosen at random,
        - p & q should be similar in magnitude but differ in length by a few digits
        - p and q are kept secret.
    - Compute n = pq.
        - n is used as the modulus for both keys
        - the length of n expressed in bits, is the key length.
    - Compute Φ(n) = (p − 1) * (q − 1)
    - Choose an integer e such that 1 < e <Φ(n) & e and Φ(n) share no factors besides 0
        - an e with a short bit-length results in more efficiency
        - the most commonly chosen value for e is 216 + 1 = 65,537
        - the smallest (and fastest) possible value for e is 3
    - Determine d ≡ e^−1 %Φ(n)
- The public key consists of the modulus n and the public (or encryption) exponent e.
- The private key consists of the private (or decryption) exponent d
- d, p, q, and λ(n) must also be kept secret because they can be used to calculate d.
    - they can all (besides d) be discarded after d has been computed

```
Suppose P = 53 and Q = 59

n = P*Q = 3127

e = 3.
```

so, Φ(n) = 3016

For k = 2, value of d is 2011.

We need to calculate Φ(n) :

Such that **Φ(n) = (P-1)(Q-1)**

d = (k*Φ(n) + 1) / e for some integer k

For k = 2, value of d is 2011.