

Computer security: message authentication

Frans Kaashoek
6.033 Spring 2007

Security goals

- Confidentiality
- Authentication
 - Message
 - User
- Authorization

RC4 (or ARC4)

byte S[256];

procedure RC4_generate() return key-byte {

i ← (i + 1) mod 256;

j ← (j + S[i]) mod 256;

swap (S[i], S[j]);

t ← (S[i] + S[j]) mod 256;

return S[t];

Initialization from a seed

```
procedure RC4_init (seed)
  for i from 0 to 255 do {
    S[i]  $\leftarrow$  i;
    K[i]  $\leftarrow$  seed[i];
  }
  j  $\leftarrow$  0;
  for i from 0 to 255 do {
    j  $\leftarrow$  (j + S[i] + K[i]) mod 256;
    swap( S[i], S[j]);
  }
  i  $\leftarrow$  0; j  $\leftarrow$  0;
```

Sign and verify using Hmac

procedure sign (m, k) {

$t \leftarrow H((k \oplus \text{outerpad}) + H((k \oplus \text{innerpad}) + m))$

return t ;

}

procedure verify (m, t, k) {

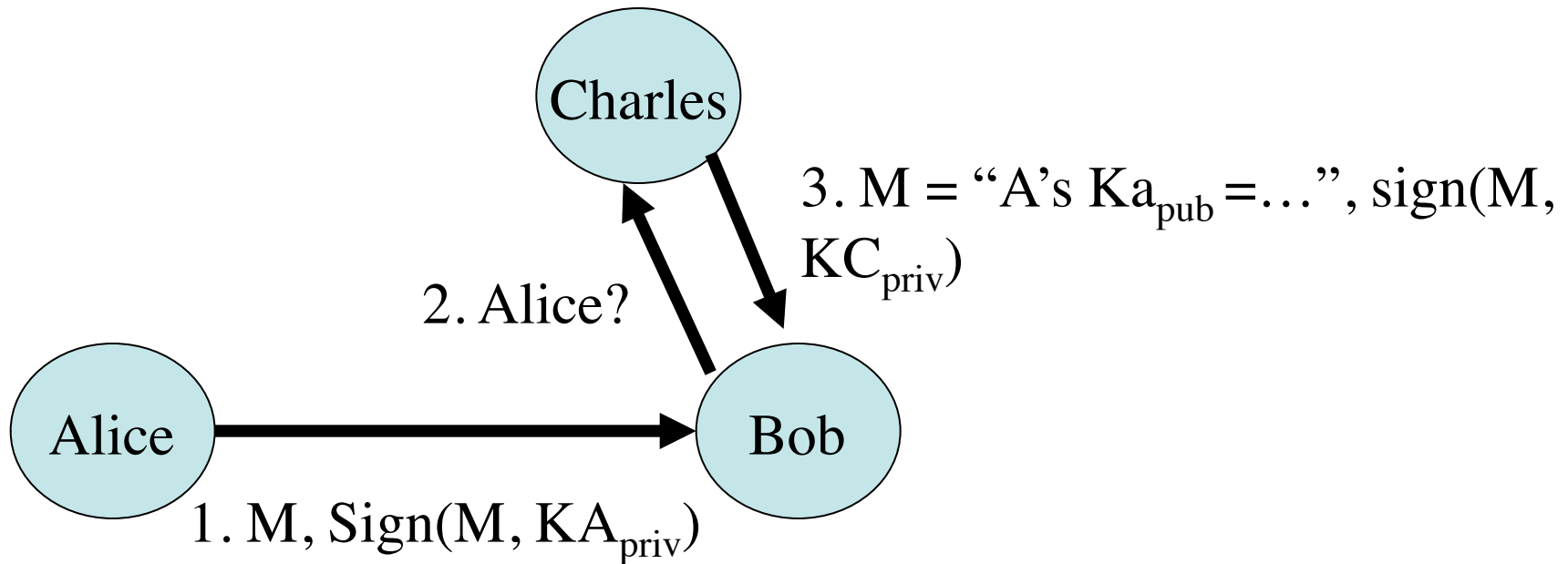
$h \leftarrow H((k \oplus \text{outerpad}) + H((k \oplus \text{innerpad}) + m))$

if ($h = t$) **return** accept;

else return reject;

}

key distribution



- 3 is a *certificate* for Alice's public key
- Charles is called a *certificate authority*

RSA public-key cipher

Transform

$$C \leftarrow m^e \pmod{n}$$

Reverse Transform

$$C^d \pmod{n} = m^{ed} \pmod{n} = m$$

- p, q primes
- $n \leftarrow p * q$
- $z \leftarrow (p-1) * (q-1)$
- Pick e relative prime to z
- Pick d s.t. $e*d = 1 \pmod{z}$
- $K1 = (e, n)$
- $K2 = (d, n)$
- Message m s.t. $0 \leq m < n$

$$p = 47, q = 59$$

$$n \leftarrow 2773$$

$$z \leftarrow 2668$$

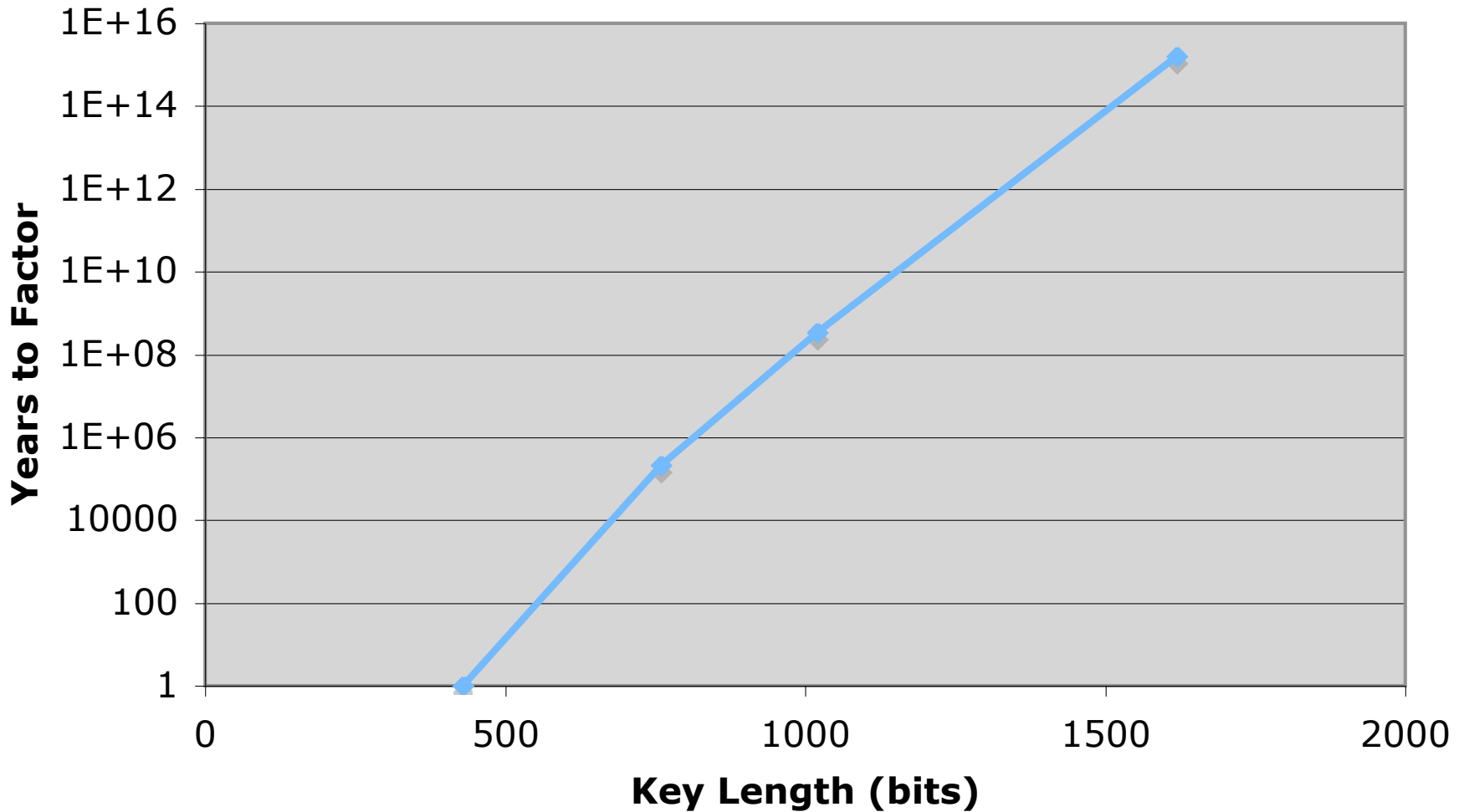
$$e = 17, d = 157$$

$$m \leftarrow 31$$

$$c \leftarrow 31^{17} \pmod{2773} = 587$$

$$587^{157} \pmod{2773} = m$$

Key Length vs. Machine-Years to Factor



2005: 640-bit key factored in 3 months
using 80 2.2Ghz Opterons

Sign and verify using RSA

```
procedure sign ( $m, K_{priv}$ ) {  
     $t \leftarrow \text{hash}(m)$   
     $t \leftarrow \text{RSA-transform}(h, K_{priv})$   
    return  $t$ ;  
}
```

```
procedure verify ( $m, t, K_{pub}$ ) {  
     $h \leftarrow \text{RSA-reverse}(t, K_{pub})$   
    if ( $\text{hash}(h) = t$ ) return accept;  
    else return reject;  
}
```

Needs further refinement!