Computer security: message authentication

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

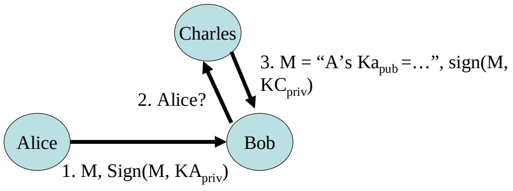
```
procedure RC4_init (seed)
    for i from 0 to 255 do {
        S[i] ← i;
        K[i] ← seed[i];
    }
    j ← 0;
    for i from 0 to 255 do {
        j ← (j + S[i] + K[i]) mod 256;
        swap( S[i], S[j]);
    }
    i ← 0; j ← 0;
```

Sign and verify using Hmac

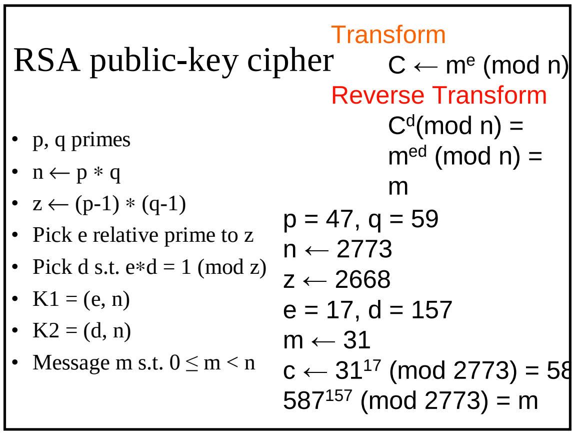
```
procedure sign (m, k) {
    t ←H((k ⊕ outerpad) +H((k ⊕ innerpad) + m))
    return t;
}

procedure verify (m, t, k) {
    h ←H((k ⊕ outerpad) + H((k ⊕ innerpad) + m))
    if (h = t) return accept;
    else return reject;
}
```

▪ key distribution


Charles
3. M = “A’s Ka$_{pub}$ =…”, sign(M, KC$_{priv}$)
2. Alice?
Alice
Bob
1. M, Sign(M, KA$_{priv}$)

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



RSA public-key cipher

- p, q primes
- n ← p ∗ q
- z ← (p-1) ∗ (q-1)
- Pick e relative prime to z
- Pick d s.t. e∗d = 1 (mod z)
- K1 = (e, n)
- K2 = (d, n)
- Message m s.t. 0 ≤ m < n

Transform

$C \leftarrow m^e \pmod n$

Reverse Transform

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

p = 47, q = 59

n ← 2773

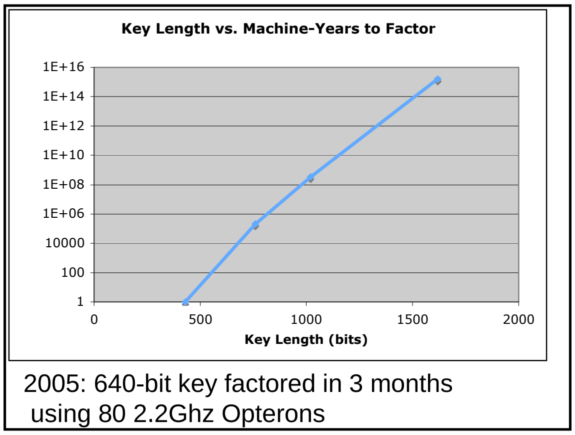
z ← 2668

e = 17, d = 157

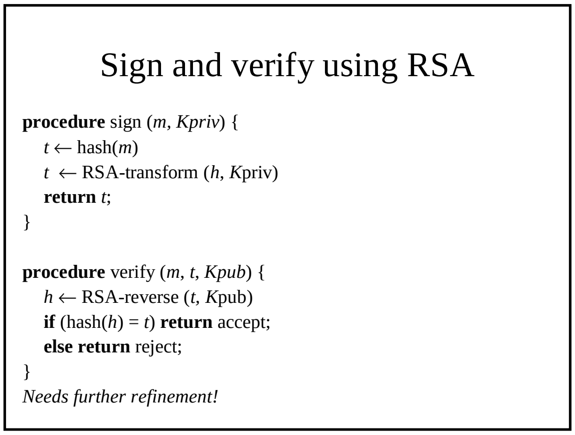
m ← 31

$c \leftarrow 31^{17} \pmod{2773} = 58$

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



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



Sign and verify using RSA

```
procedure sign (m, Kpriv) {
    t ← hash(m)
    t ← RSA-transform (h, Kpriv)
    return t;
}

procedure verify (m, t, Kpub) {
    h ← RSA-reverse (t, Kpub)
    if (hash(h) = t) return accept;
    else return reject;
}
```

Needs further refinement!