CSE 509 class note (4/19)

Side Channel Attacks

- Timing
- Power
- Sound
- Light

- Usability and Security
    - Depending on Users
    - Role playing

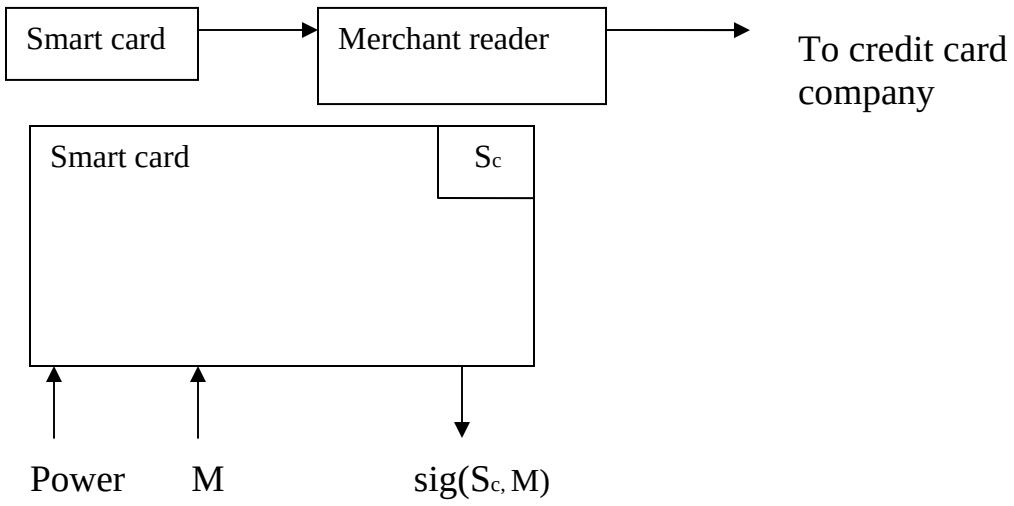
- Modular Exponentiation

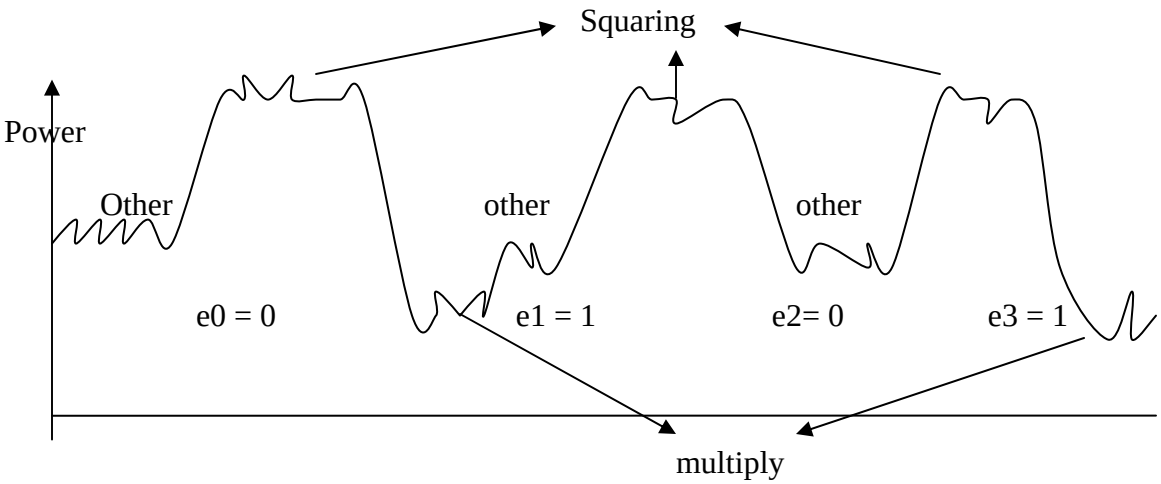
```
ModExp(m,e,n)              // M^e mod n
        Let el ... e0 = e  // the bits of e
        Let acc = 1
        For i=0 … l
                if ei == 1
                        acc = acc*m mod n
                m = m^2 mod n
        return acc;
```

$m^e \bmod n = m^{(2^0 e_0 + 2^1 e_1 + 2^2 e_2 + \ldots + 2^l e_l)} \bmod n$

$= m^{e_0} * (m^2)^{e_1} * (m^4)^{e_2} * (m^{(2^l)})^{e_l} \bmod n$

- Power Analysis (smart cards)





- timing attack against square and multiply
  some reductions take longer than others
  - fast reductions
  - slow reductions
  - if attackers knows acc*m, he can predict whether reduction is fast or slow

attacker input	m0	m1	m2	m3	…	mk
time	t0	t1	t2	t3	…	tk

suppose e0 = 1

- can compute
  - acc and m at end of round 0
- predict whether eound 1 will be fast or slow for each message mi
- let
  - f = average time of “fast” messages
  - s = average time of “slow” messages

- two cases
  - | s1 – f1| is small and |s0 –f0| is small
    ➔ e1 = 0
- suppose |sb – fb| is large and |s(~b) – f(~b)| is small
  ➔ e1 = 1 and e0 = b

repeat for round 2, … , l

- defense
  - for i=0 … l

```
if ei==1
        acc = acc*m mod n
else
        tmp = acc*m mod n
m = m^2 mod n
```

- RSA binding
    to compute m^e mod n
    - pick random r
    - compute x = (rm)^e mod n
    - compute y = r^e mod n
    - return x/y mod n