

Timing Attacks

Side Channel Attacks

- Timing
- Power
- Source
- Light

Usability and Security

- Depending on Users
- Role Playing

Modular Exponentiation

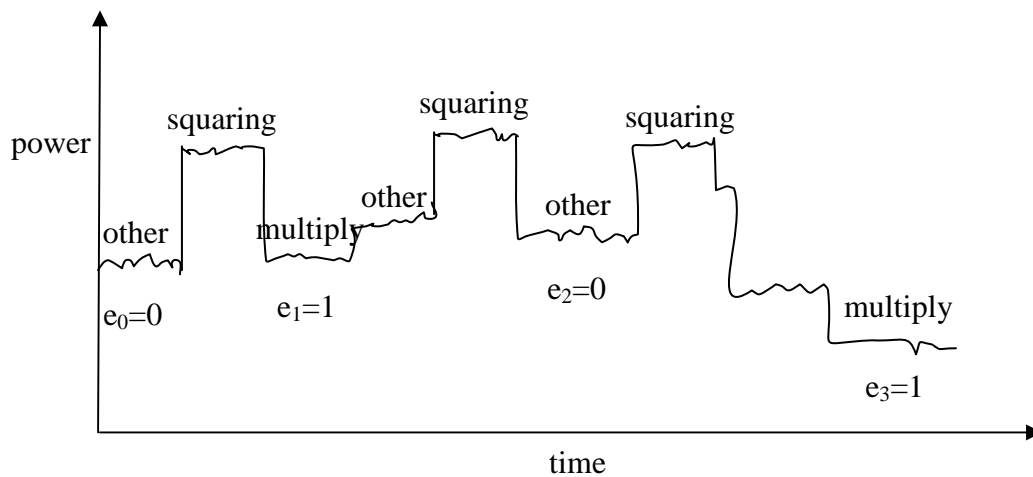
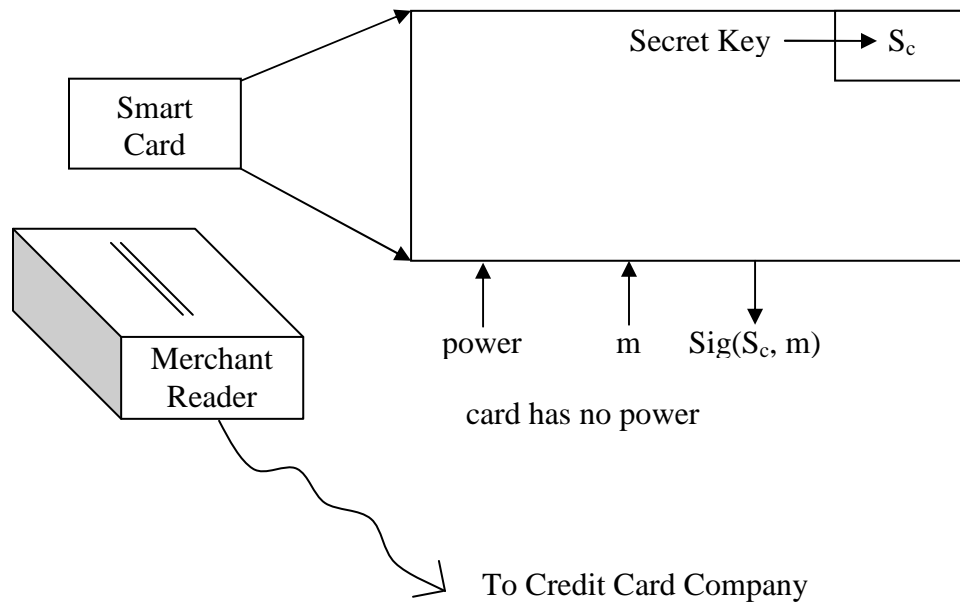
Modexp(m,e,n) // $m^e \bmod n$
let e_ℓ, \dots, e_0 // the bits of e

Square and Multiply

```
let acc = 1
for i=0 ...  $\ell$ 
  if  $e_i == 1$ 
    acc = acc * m mod n
  m = m2 mod n
return acc
```

$$\begin{aligned} m^e \bmod n &= m^{(2^0 e_0 + 2^1 e_1 + 2^2 e_2 + \dots + 2^\ell e_\ell)} \bmod n \\ &= m^{e_0} * (m^2)^{e_1} * (m^4)^{e_2} * \dots * (m^{2^\ell})^{e_\ell} \bmod n \end{aligned}$$

Power Analysis (Smart Cards)



Timing Attack against Square and Multiply

Some reductions take longer than others

-fast reductions

-slow reductions

-If attacker knows $acc * m$, he can predict whether reduction is fast or slow.

Attacker

input

m_0

m_1

m_2

time

t_0

t_1

t_2

·	·
·	·
·	·
m_k	t_k

Suppose e_{-1}

-can compute acc and match end of round 0

-Predict whether round 1 will be fast or slow for each message m_i .

let f_1 = average time of “fast” messages

s_1 = average time of “slow” messages

Two Cases

1. $|s_1 - f_1|$ is small and $|s_0 - f_0|$ is small $\Rightarrow e_1 = 0$

2. Suppose $|s_b - f_b|$ is large and $|s_{\tilde{b}} - f_{\tilde{b}}|$ is small $\Rightarrow e_1 = 1$ and $e_0 = b$

Repeat for rounds 2 ... ℓ

Safeguard against Timing Attacks

-modify the code

let acc = 1

for $i=0 \dots \ell$

 if $e_i == 1$

 acc = acc * m mod n

 else

 tmp = m mod n

 m = m² mod n

return acc

RSA Blinding

To compute $m^e \bmod n$

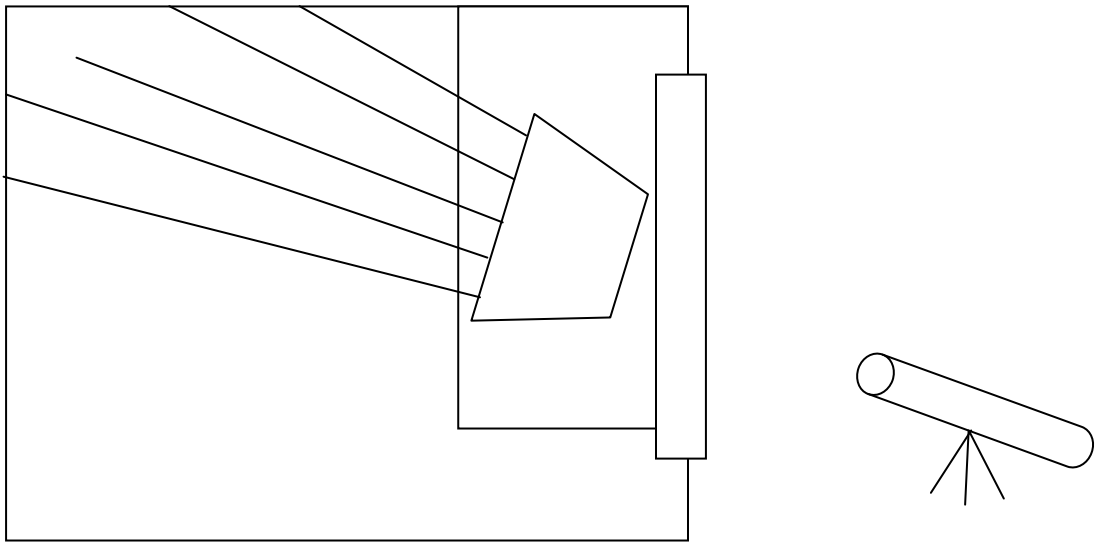
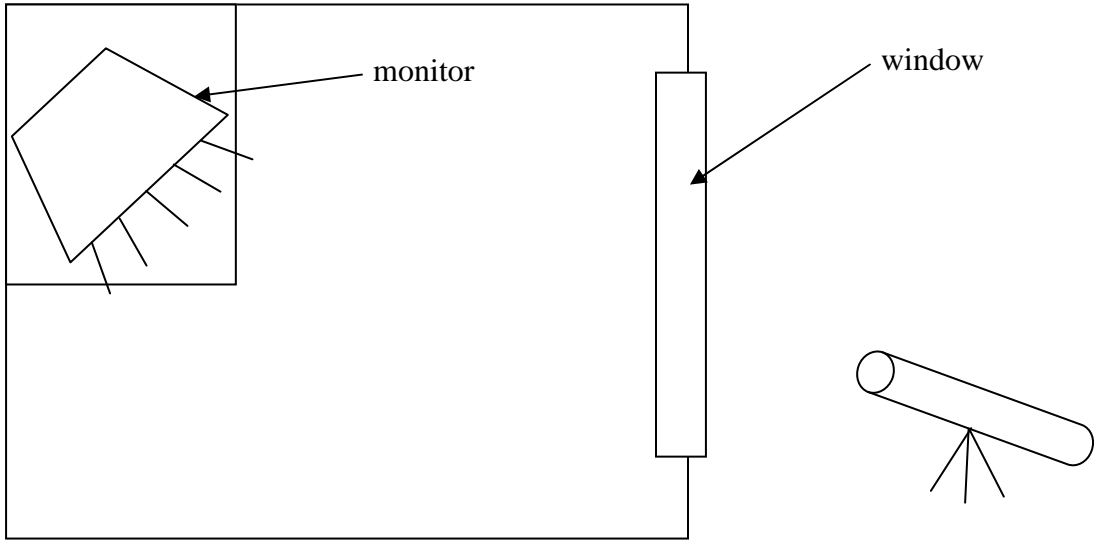
1. pick random r

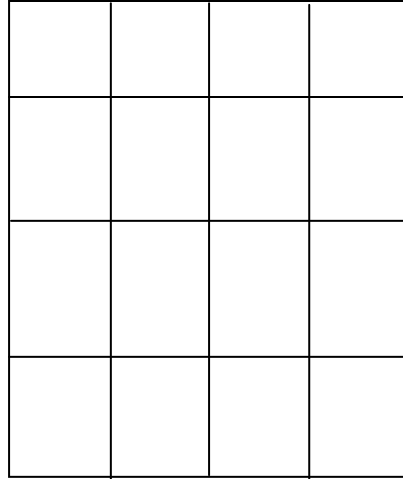
2. compute $x = (r * m)^e \bmod n$

3. compute $y = r^e \bmod n$

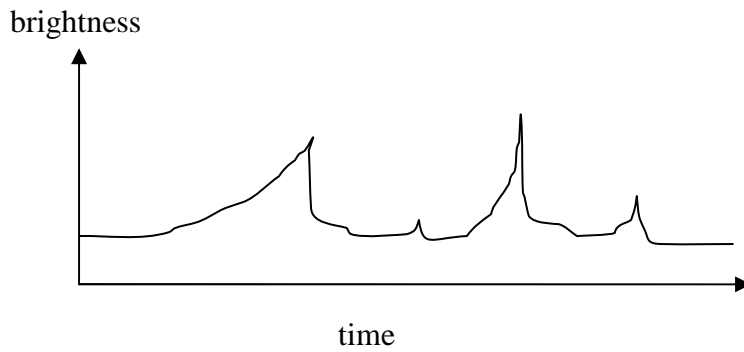
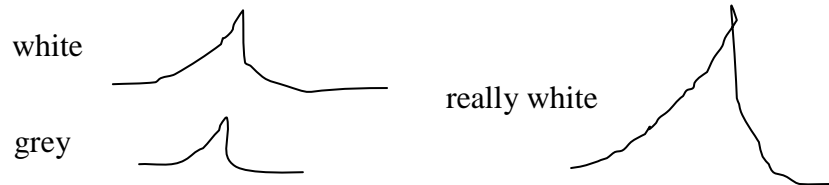
4. return $x/y \bmod n$

Light Attacks





Screen, cathode ray tubes monitor



Can determine what is displayed on the screen by recording the brightness change since the monitor updates one pixel at a time. To prevent such an attack would be to update all pixels at once.