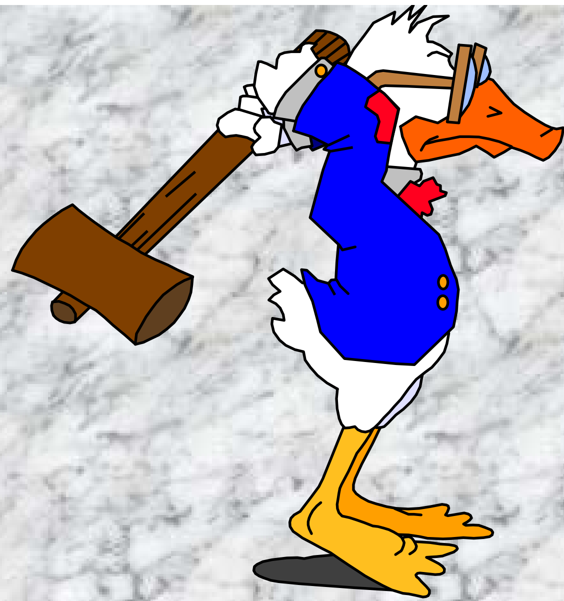


Breaking Smart Card Based ElGamal Signature and Its Variants

Asiacrypt96 Rump Session, 5 Nov. 1996

Yuliang Zheng
Monash University

Tsutomu Matsumoto
Yokohama Nat. University

Exploiting hardware faults

- **Secret key algorithms**
  - **DES, IDEA, FEAL, etc --- broken by Biham & Shamir (18 Oct. & 30 Oct.)**
  - **(see also J-J Quisquater, 23 Oct.)**
- **Public key signature**
  - **RSA --- broken by**
    - **Bellcore (25 Sept., no details were published)**
    - **Nat. Uni. of Singapore (23 Oct.)**
  - ***ElGamal family -- ???***

Completing the big picture

- We show how to break smart card based
  - ElGamal signature and
  - all its variants,
    by (temporally) falsifying part of the circuitry in the smart card
- Time: 31 Oct. 1996

The Attack Model

- An attacker, who is in possession of a target smart card,

  may introduce faults into tamper-proof hardware in the smart card,

  say by exposing it to certain physical effects (heat, laser, pressure, radiation, etc)

The Attack Model (cnt'd)

- **The attack may then compromise a secret in the smart card.**
  - **a DES encryption / decryption key**
  - **a RSA signature-generation / decryption key**
  - **certain authentication / identification cards**
  - **……**

ElGamal Signature

- **Alice's keys**
  - x_a **--- secret key for signature generation**
  - $y_a = g^{x_a} \bmod p$ **--- public key**
- **Alice's signature on a message *m* is a pair of number (r,s):**
  - $x \in_R [1, \ldots, p-1]$
    $r = g^x \bmod p$
  - $s = \dfrac{hash(m) - x_a \cdot r}{x} \bmod (p-1)$

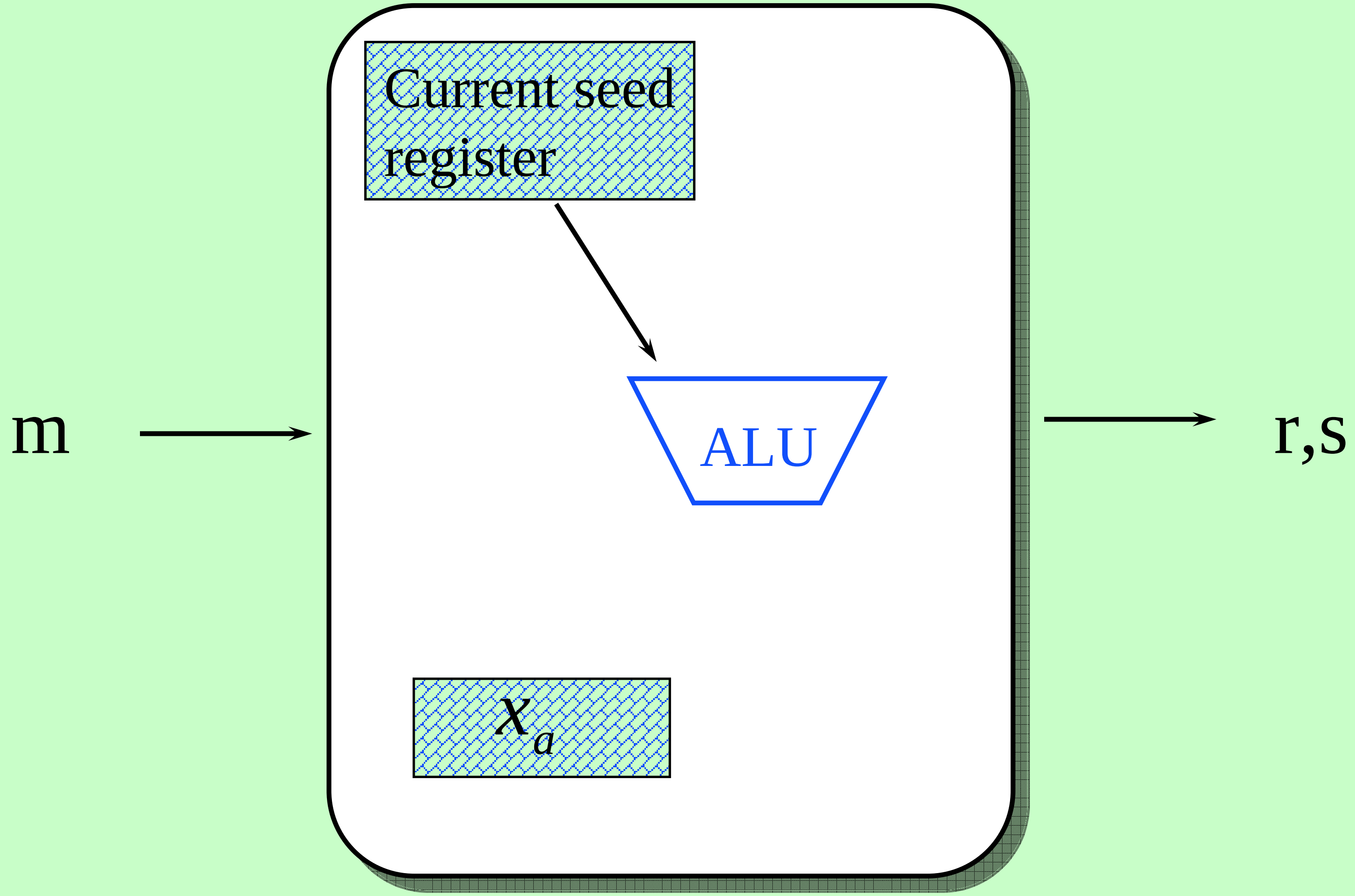
Idea for finding x_a

- An attacker may figure out x_a by making the "random" number x predictable, through the falsification of part of the hardware
  - software pseudo-random number generator --- suppressing the contents of the status register that stores the current "seed" number
  - hard pseudo-random number generator --- suppressing its output

Two possibilities

- software pseudo-random number generator
  - suppressing the contents of the status register that stores the current "seed" number
- hard pseudo-random number generator
  - suppressing its output

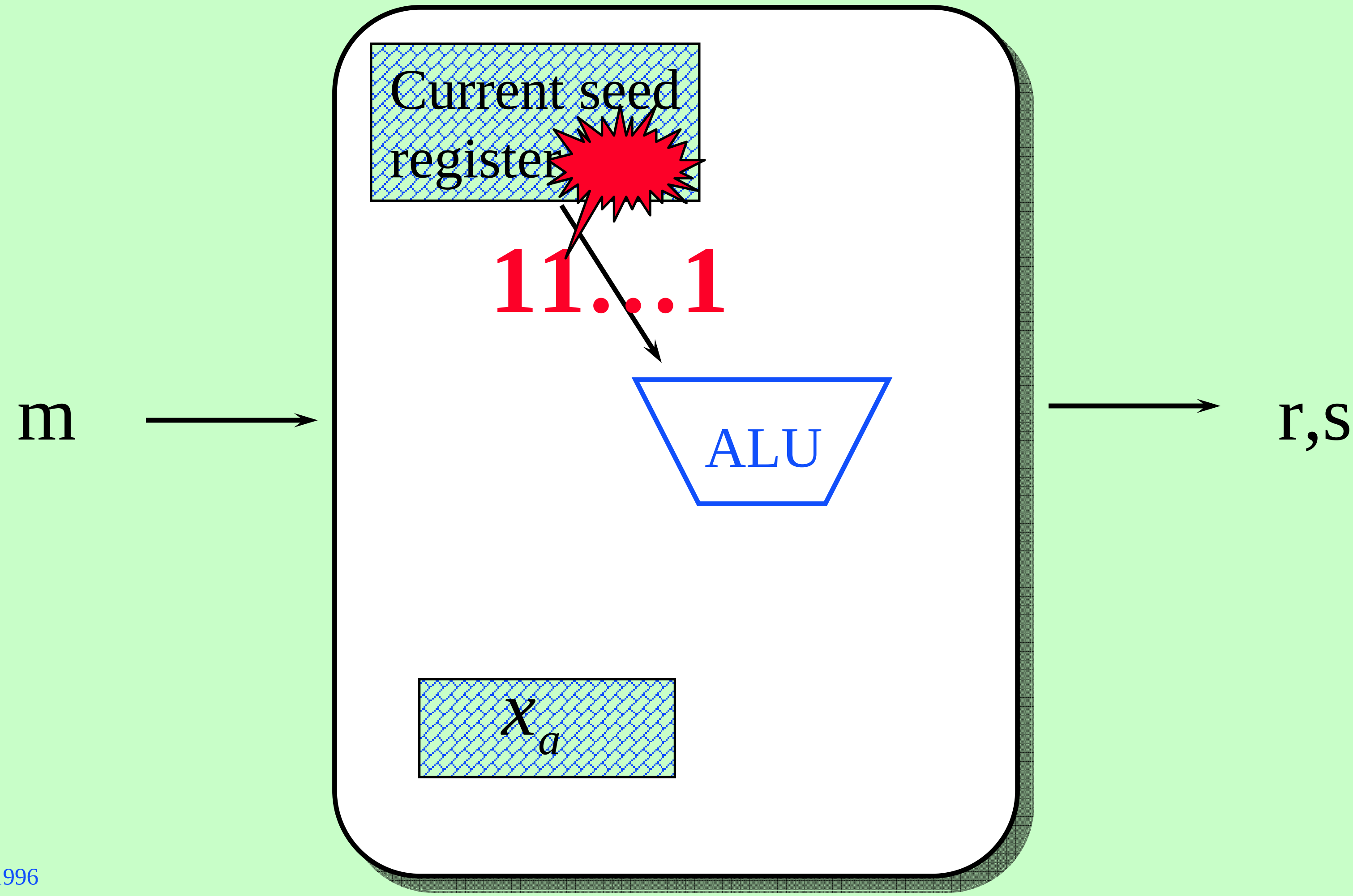
When software PRG is used



Main idea

- suppress the contents of the status register that stores the current seed for the PRG
- so that only a fixed number, such as the all-1 value, is (temporarily) available to the CPU
- collect the output (*r*,*s*) of the smart card

When software PRG is used

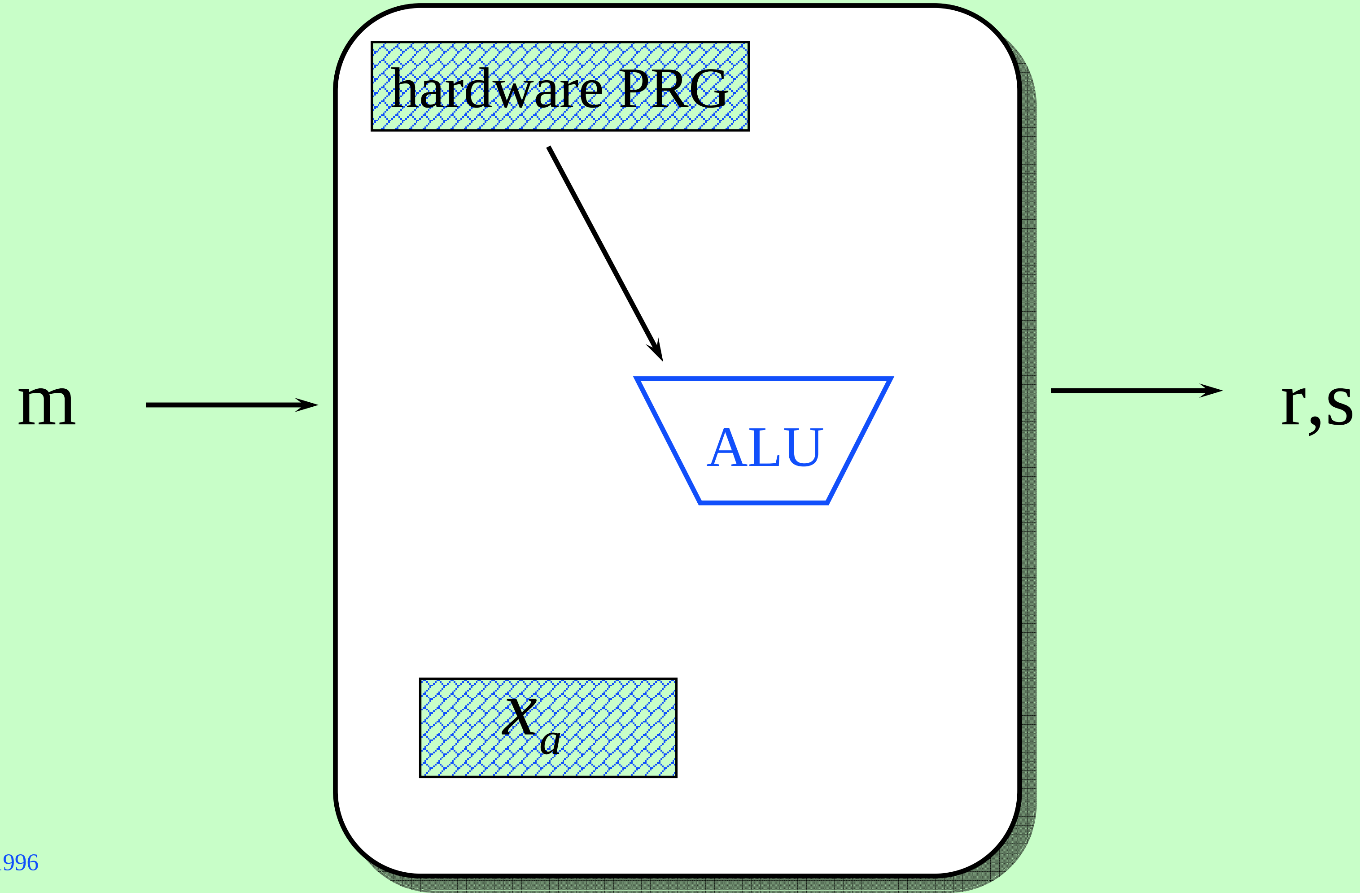


Obtain the secret key x_a

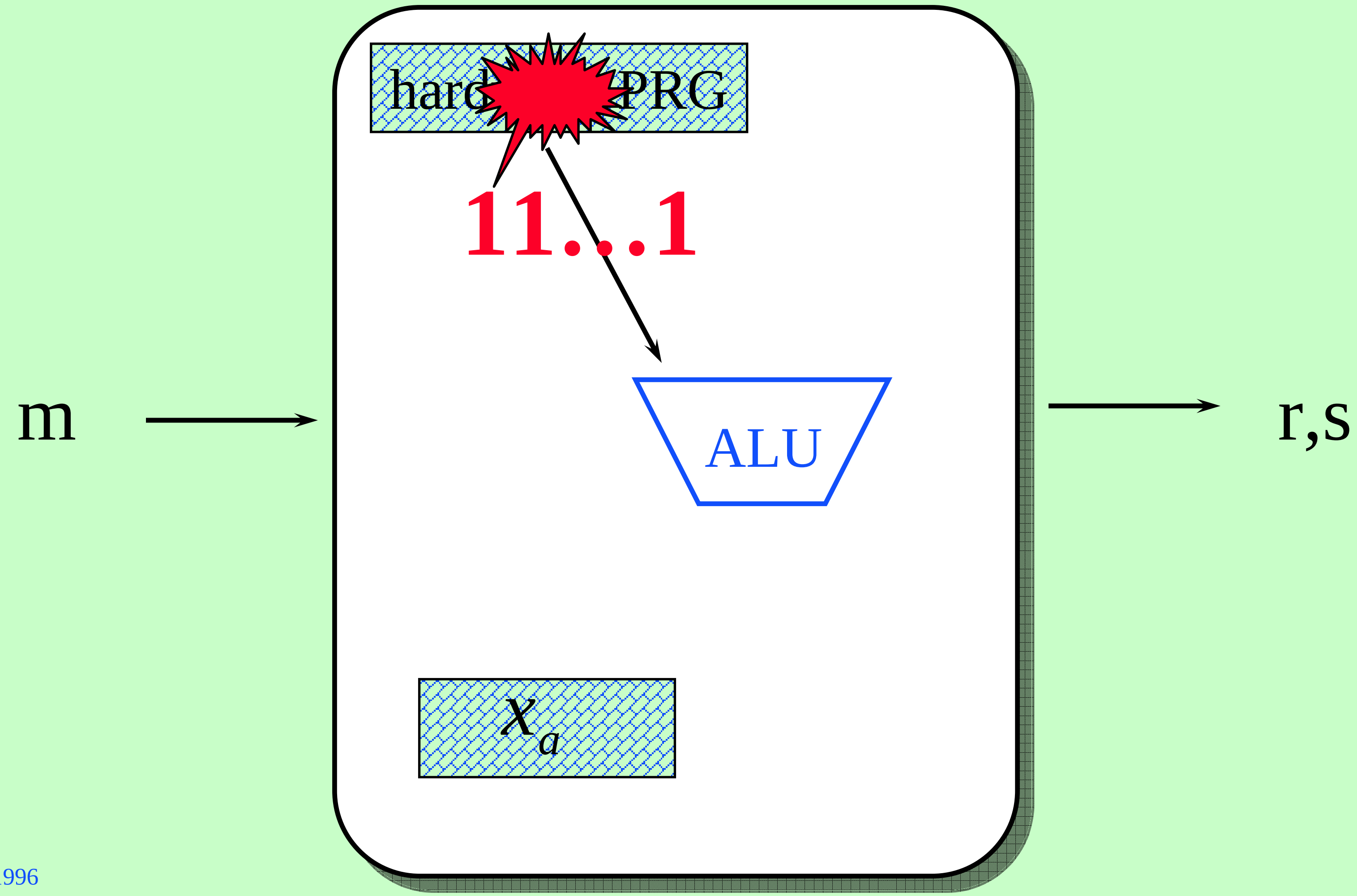
- **The attacker can now calculate x from**
  - **the fixed seed, say the all-1 value**
  - **the (public) PRG algorithm**
- **He can then find out x_a**

$$x_a = \frac{hash(m) - s \cdot x}{r} \bmod (p-1)$$

When hardware PRG is used



Suppressing output of hardware PRG



Obtain the secret key x_a

- **the attacker computes**

$$x_a = \frac{hash(m) - s \cdot x}{r} \bmod (p-1)$$

Another attack

- Some smart cards have a built-in PRG that would produce a constant output when a lower than normal voltage is supplied.
- For such a smart card, an attacker may obtain Alice's secret key x_a by asking the card to sign 2 different messages while keeping the voltage low.

Another attack (cnt'd)

- **We will have**

$$s_1 = \frac{hash(m_1) - x_a \cdot r}{x} \bmod (p-1)$$

$$s_2 = \frac{hash(m_2) - x_a \cdot r}{x} \bmod (p-1)$$

- **Hence**

$$x = \frac{hash(m_1) - hash(m_2)}{s_1 - s_2} \bmod p-1$$

$$x_a = \frac{hash(m_1) - x \cdot s_1}{r} \bmod p-1$$

Breaking other ElGamal variants

- All other ElGamal variants can be broken in a similar way
  - DSS
  - Schnorr
  - ……
  - (Elliptic curve based schemes)

Attacking RSA by NUS

- Nat Uni Singapore method
  - Finding an RSA decryption / signature generation key by

    complementing ***one or a few*** **bits at random positions in the unknown decryption/signature-generation key,**

    which is harder than suppressing the entire contents of a register !

A comparison with RSA (cnt'd)

- **Therefore, compared to the attacker on RSA signature / decryption card, our attack seems**
  - **simpler**
  - **more feasible to mount**
    - **say, on an entire data-bus**
  - **but, harder to prevent**
    - **reason --- (*r*,*s*) is a perfectly valid signature !**

Smart cards under threat
--- the big picture ---