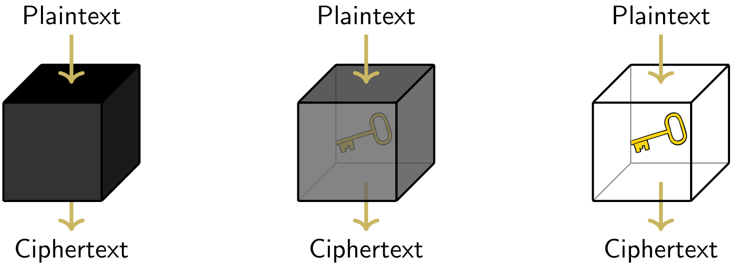
ECDSA White-Box Implementations
Feedback on CHES 2021 WhibOx Contest

Agathe Houzelot

October 18, 2023

Black-Box, Grey-Box, White-Box



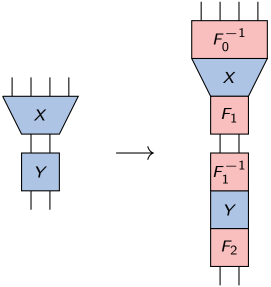
Cryptanalysis	Side-channels/Faults	Read/modify the binary/memory

Look-up tables and encodings

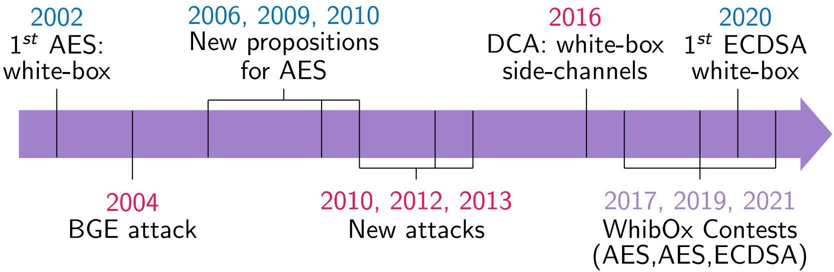
input	0	1	...	$2^{128}-1$
output	$AES_k(0)$	$AES_k(1)$	...	$AES_k(2^{128}-1)$

Look-up tables and encodings

input	0	1	...	$2^{128}-1$
output	$AES_k(0)$	$AES_k(1)$	...	$AES_k(2^{128}-1)$



State of the Art



- New designs
- New attacks
- Both designs and attacks

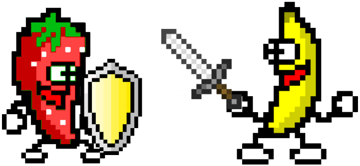
CHES 2021 Challenge - the WhibOx Contest

Designers

- Post C codes computing ECDSA
- Challenges gain strawberries (depending on performances and time until break)

Attackers

- Try to extract the secret key
- Receive bananas (number of strawberries of the challenge)



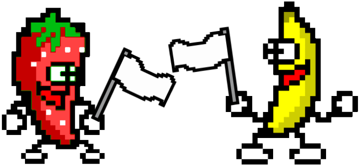
Our Contributions [1]

zerokey

- Posted the 2 winning challenges
- Described the implementations

TheRealIdefix

- Broke the most challenges
- Described attacks, showing which ones succeeded for each candidate



ECDSA

- Let G be a point of order n on an elliptic curve E
- Let d be a 256-bit key
- Let m be a message and $e = H(m)$ its hash value

Algorithm 1: ECDSA signature

1 $k \stackrel{\$}{\leftarrow} [\![1, n-1]\!]$
2 $R \leftarrow kG$
3 $r \leftarrow x_R \bmod n$
4 $s \leftarrow k^{-1}(e + rd) \bmod n$
5 **if** $r == 0$ *or* $s == 0$ **then**
6   Go to step 1
7 **end**
8 Return (r,s)

ECDSA Sensitive Values

- Let G be a point of order n on an elliptic curve E
- Let d be a 256-bit key
- Let m be a message and $e = H(m)$ its hash value

Algorithm 1: ECDSA signature

```
1 k ←$ [[1, n − 1]]
2 R ← kG
3 r ← x_R mod n
4 s ← k^{−1}(e + rd) mod n
5 if r == 0 or s == 0 then
6 |   Go to step 1
7 end
8 Return (r,s)
```

ECDSA Sensitive Values

- Let G be a point of order n on an elliptic curve E
- Let d be a 256-bit key
- Let m be a message and $e = H(m)$ its hash value

Algorithm 1: ECDSA signature

1 $k \xleftarrow{\$} [\![1, n-1]\!]$
2 $R \leftarrow kG$
3 $r \leftarrow x_R \bmod n$
4 $s \leftarrow k^{-1}(e + rd) \bmod n$
5 **if** $r == 0$ *or* $s == 0$ **then**
6   Go to step 1
7 **end**
8 Return (r,s)

Deterministic ECDSA

- Let G be a point of order n on an elliptic curve E
- Let d be a 256-bit key
- Let m be a message and $e = H(m)$ its hash value

Algorithm 1: ECDSA signature

1 $k \xleftarrow{\$} [\![1, n-1]\!]$ WB model $\Rightarrow$ No reliable source of randomness!
2 $R \leftarrow kG$
3 $r \leftarrow x_R \bmod n$
4 $s \leftarrow k^{-1}(e + rd) \bmod n$
5 **if** $r == 0$ *or* $s == 0$ **then**
6 | Go to step 1
7 **end**
8 Return (r,s)

Deterministic ECDSA

- Let G be a point of order n on an elliptic curve E
- Let d be a 256-bit key
- Let m be a message and $e = H(m)$ its hash value

Algorithm 1: ECDSA signature

1 $k \leftarrow f(e)$ WB model $\Rightarrow$ No reliable source of randomness!
2 $R \leftarrow kG$
3 $r \leftarrow x_R \bmod n$
4 $s \leftarrow k^{-1}(e + rd) \bmod n$
5 **if** $r == 0$ *or* $s == 0$ **then**
6   Go to step 1
7 **end**
8 Return (r,s)

Memory dumps

Idea

Find some secret values that could be manipulated in the clear

- Easy since we had access to a C code and not a binary
- Usual encoding techniques not suited for operations on big numbers $\rightarrow$ one has to design new techniques

Biased Nonce

First possibility

Find collisions: signatures generated with the same nonce

- Find (r_1, s_1) and (r_2, s_2) such that $r_1 = r_2$ (so $k_1 = k_2$)
- Solve the following system in k, d:

$$\begin{cases} s_1 = k^{-1}(e_1 + r_1 d) \\ s_2 = k^{-1}(e_2 + r_2 d) \end{cases}$$

Biased Nonce

First possibility

Find collisions: signatures generated with the same nonce

- Find (r_1, s_1) and (r_2, s_2) such that $r_1 = r_2$ (so $k_1 = k_2$)
- Solve the following system in k, d:

$$\begin{cases} s_1 = k^{-1}(e_1 + r_1 d) \\ s_2 = k^{-1}(e_2 + r_2 d) \end{cases}$$

Second possibility

Exploit biases in the nonce generation

- Use lattice-based attacks
- Allows to recover d from a few bits of k for several signatures

Grey-Box Attacks in the White-Box Model

- Side-channel attacks
  - ≻ Difficult to apply (huge size of the traces)

Grey-Box Attacks in the White-Box Model

- Side-channel attacks
  - ≻ Difficult to apply (huge size of the traces)
- Fault injections
  - ≻ Modify the binary, use debugging tools ⇒ very precise faults
  - ≻ Many fault attacks on deterministic ECDSA, for example:

Valid signature

$$r = x_R \bmod n$$

$$s = k^{-1}(e + rd) \bmod n$$

Grey-Box Attacks in the White-Box Model

- Side-channel attacks
  - ≻ Difficult to apply (huge size of the traces)
- Fault injections
  - ≻ Modify the binary, use debugging tools ⇒ very precise faults
  - ≻ Many fault attacks on deterministic ECDSA, for example:

Valid signature	Faulty signature
$r = x_R \bmod n$	$r' = x_{R^{\lightning}} \bmod n$
$s = k^{-1}(e + rd) \bmod n$	$s' = k^{-1}(e + r'd) \bmod n$

Grey-Box Attacks in the White-Box Model

- Side-channel attacks
  - ≻ Difficult to apply (huge size of the traces)
- Fault injections
  - ≻ Modify the binary, use debugging tools ⇒ very precise faults
  - ≻ Many fault attacks on deterministic ECDSA, for example:

Valid signature

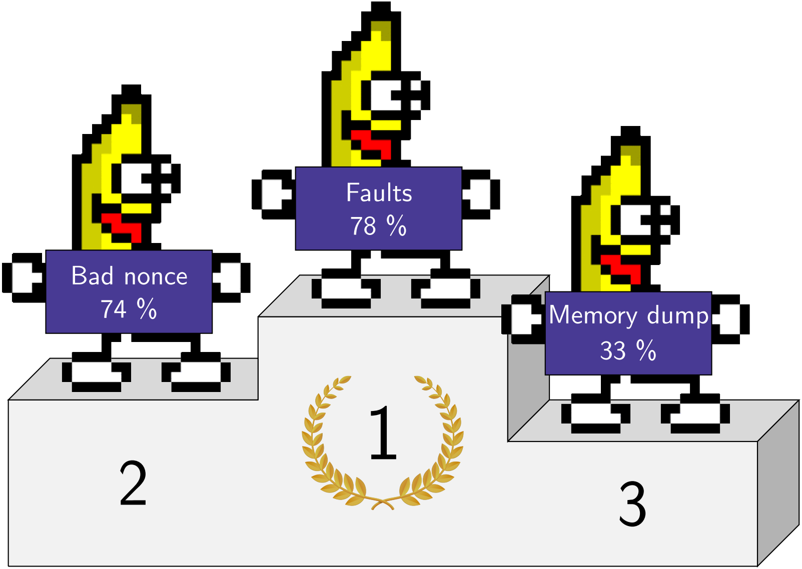
$$r = x_R \bmod n$$
$$s = k^{-1}(e + rd) \bmod n$$

Faulty signature

$$r' = x_{R^{\lightning}} \bmod n$$
$$s' = k^{-1}(e + r'd) \bmod n$$

$$d = (s(r - r')(s - s')^{-1} - r)^{-1} e \bmod n$$

Percentage of Challenges Broken by Each Attack



Conclusion

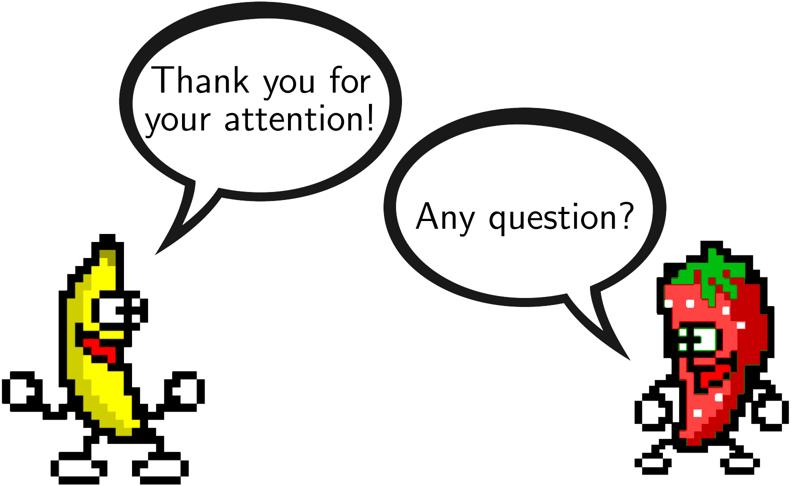
- Securing ECDSA seems even more difficult than the AES
  - Our automated attacks broke 95 out of 97 challenges
  - All the challenges were broken in less than 33 hours

Conclusion

- Securing ECDSA seems even more difficult than the AES
  - ➢ Our automated attacks broke 95 out of 97 challenges
  - ➢ All the challenges were broken in less than 33 hours
- What about the ECDSA white-box published in 2020?
  - ➢ Broken too but with a more sophisticated fault attack [2]

Conclusion

- Securing ECDSA seems even more difficult than the AES
  - Our automated attacks broke 95 out of 97 challenges
  - All the challenges were broken in less than 33 hours
- What about the ECDSA white-box published in 2020?
  - Broken too but with a more sophisticated fault attack [2]
- Is there any hope ?
  - Possible to increase a lot the workload of the attacker
  - Companies sell ECDSA white-boxes evaluated by specialized labs and not broken


Thank you for your attention!
Any question?

G. Barbu, W. Beullens, E. Dottax, C. Giraud, A. Houzelot, C. Li, M. Mahzoun, A. Ranea, and J. Xie.
Ecdsa white-box implementations: Attacks and designs from ches 2021 challenge.
IACR Transactions on Cryptographic Hardware and Embedded Systems, pages 527–552, 2022.

C. Giraud and A. Houzelot.
Fault attacks on a cloud-assisted ecdsa white-box based on the residue number system.
Workshop on Fault Detection and Tolerance in Cryptography (FDTC), 2023.