Federal Office for Information Security

Attacks Against White-Box ECDSA and Discussion of Countermeasures

Sven Bauer [1] Hermann Drexler [2] Maximilian Gebhardt [3] Dominik Klein [3]
Friederike Laus [3] Johannes Mittmann [3]

[1]Siemens AG

[2]G+D Mobile Security

[3]Federal Office for Information Security (BSI)

September, 20th 2022

WhibOx Contest 2021

Rules & Setting

- designers submit ECDSA in C source code
- full source code available to attackers
- attackers extract private key d
- scores depend on execution time, RAM usage and time to break

ECDSA

INPUT hash h

OUTPUT signature (r, s)

1. set $k \leftarrow \text{rand32}()$
2. set $r \leftarrow \big((k\,G)_x \bmod p\big) \bmod q$
3. set $s \leftarrow k^{-1}(rd + h) \bmod q$
4. if $r = 0$ or $s = 0$, go to **step 1**, otherwise return (r, s)

Federal Office for Information Security

ECDSA

INPUT hash h

OUTPUT signature (r, s)

1. set $k \leftarrow \text{rand32}()$ ▶ no source of randomness!
2. set $r \leftarrow \big((k\,G)_x \bmod p\big) \bmod q$
3. set $s \leftarrow k^{-1}(rd + h) \bmod q$
4. if $r = 0$ or $s = 0$, go to **step 1**, otherwise return (r, s)

Deterministic ECDSA

INPUT hash h

OUTPUT signature (r, s)

1. set $z \leftarrow \text{seed}(h)$
2. set $(k, z) \leftarrow \text{rand}(z)$ ▶ deterministic RNG seeded with h
3. set $r \leftarrow \big((k\,G)_x \bmod p\big) \bmod q$
4. set $s \leftarrow k^{-1}(rd + h) \bmod q$
5. if $r = 0$ or $s = 0$, go to **step 2**, otherwise return (r, s)

Signature Equational System

- signature computation

$$\begin{aligned} s &= k^{-1}(rd + h) \\ \Leftrightarrow \quad rd - sk &= -h \end{aligned}$$

- suppose $(r_1, s_1), \ldots, (r_m, s_m)$ for $h_1, \ldots, h_m$

$$\begin{aligned} r_1\, d - \; s_1\, k_1 &= \; - h_1 \\ &\vdots \\ r_m\, d - s_m\, k_m &= \; - h_m \end{aligned}$$

- m equations, but $m + 1$ unknowns: $k_1, \ldots, k_m$ and d

Key Collision Attacks

Ephemeral Key Collision

- two hashes $h_1 \neq h_2$ with same k
- aka Playstation3-ECDSA

$$r_1 d - s_1 k = -h_1 ,$$
$$r_2 d - s_2 k = -h_2 ,$$

- two equations, solve for d, k
- broken challenges: 33 / 97

Federal Office for Information Security

Ephemeral Key Differential Collision

- some bits of k only depend on small part of h
- suppose h_1, h_2, h_3, h_4 with $h_i \neq h_j$
- e.g. $h_2 = h_1 + \Delta$ and $h_4 = h_3 + \Delta$
- with k_1 and $k_2 = k_1 + t$, and k_3 and $k_4 = k_3 + t$

$$
\begin{aligned}
r_1\, d - s_1\, k_1 \qquad\qquad\qquad\quad &= -h_1 \\
r_2\, d - s_2\, k_1 \qquad\qquad - s_2\, t &= -h_2 \\
r_3\, d \qquad\quad - s_3\, k_3 \qquad\quad &= -h_3 \\
r_4\, d \qquad\quad - s_4\, k_3 - s_4\, t &= -h_4
\end{aligned}
$$

- solve for d and k_1, k_3, t
- broken challenges: 49 / 97

Collision Fault Analysis

Fault Model

Crucial Steps

2. set $(k, z) \leftarrow \text{rand}(z)$
3. set $r \leftarrow (k\,G)_x$
4. set $s \leftarrow k^{-1}(rd + h)$

Fault Attack

- let v be intermediate value ($k, k^{-1}, r, d, rd, rd + h$)
- fault v to e **(value fault)**
- fault v to $v + e$ **(differential fault)**
- find collisions, i.e. same e for different h

Differential Fault in r

INPUT hash h

OUTPUT signature (r, s)

1. set $z \leftarrow \text{seed}(h)$
2. set $(k, z) \leftarrow \text{rand}(z)$
3. set $r \leftarrow (k\,G)_x$
4. set $s \leftarrow k^{-1}(rd + h)$
5. if $r = 0$ or $s = 0$, go to **step 2**, otherwise return (r, s)

Federal Office
for Information Security

Differential Fault in r

INPUT hash h

OUTPUT signature (r, s)

1. set $z \leftarrow \text{seed}(h)$
2. set $(k, z) \leftarrow \text{rand}(z)$
3. set $r \leftarrow (k\,G)_x$
4. set $s \leftarrow k^{-1}((r + e)d + h)$ ▶ fault r to $r + e$
5. if $r = 0$ or $s = 0$, go to **step 2**, otherwise return (r, s)

Differential Fault in r

Obtain Equations

$$
\begin{aligned}
r_{c,i}\, d - s_{c,i}\, k_i \qquad &= -h_i\,, \\
r_{f,i}\, d - s_{f,i}\, k_i + e_i d &= -h_i
\end{aligned}
$$

with unknowns $d, k_i, e_i d$

- find $h_i \neq h_j$ with $e_i = e_j$ (fault collision)
- solve for $(d, k_i, k_j, e_i d) = (d, k_i, k_j, e_j d)$.
- broken challenges: 39 / 97 (includes faults in rd, h or $rd + h$)

Experimental Results

Experimental Setup

- *automated* faults by `NOP`ing out instructions
- *static* faults only (patching binary)
- *few fixed* input hash values
- *small* number of generated signatures
- all designs broken by faults or key collisions

This round goes to the attackers!

Key Collision Attacks: Summary

Ephemeral Key Collision (constant key)	12/97
Ephemeral Key Collision (chosen hashes)	21/97
Cross-Challenge Ephemeral Key Collision	40/97
Ephemeral Key Differential Collision	49/97

Simple & Collision Fault Attacks: Summary

uncontrolled Fault in r	88/97
Value Fault in r (correct r returned) or rd	51/97
Value/Differential Fault in d	53/97
Value Fault in h	57/97
Value Fault in $rd + h$	11/97
Value Fault in k or k^{-1}	75/97
Differential Fault in r, rd, h or $rd + h$	39/97
Differential Fault in k	34/97
Differential Fault in k^{-1}	0/97

Countermeasures

Countermeasures

Motivation & Idea

- prevent described (single) fault attacks
- *infective computation* by Romailler and Pelissier
- protects against *uncontrolled fault in r*
- but: additive blinding not effective against *differential faults*
- here: combine with infective computation w/ *multiplicative blinding*

Conclusion

Conclusion

- various computational and fault attacks apply to WBC
- all challenges broken by automated, static faults or key collisions, in particular
  - uncontrolled fault in r
  - ephemeral key differential collision
- difficult to prevent by program obfuscation

Future Work

- countermeasures?
- differential computational analysis for asymmetric WBC?

Thanks for your attention!

Contact

Dominik Klein
Head of Section
Section TK11 - Chip Security

firstname.lastname@bsi.bund.de
Tel. +49 (0) 228 9582 0
Fax +49 (0) 228 10 9582 5400

Federal Office for Information Security (BSI)
Godesberger Allee 185-189
53175 Bonn
www.bsi.bund.de