New Bleichenbacher Records: Fault Attacks on qDSA Signatures

CHES 2018

Akira Takahashi[1] Mehdi Tibouchi[1,2] Masayuki Abe[1,2]

September 12, 2018

[1]Kyoto University

[2]NTT Secure Platform Laboratories

Outline

Introduction

Motivating Example: Schnorr Signature Scheme

- One of the simplest and most widely-used digital signature schemes

Motivating Example: Schnorr Signature Scheme

- One of the simplest and most widely-used digital signature schemes
- Most notable variant: (EC)DSA

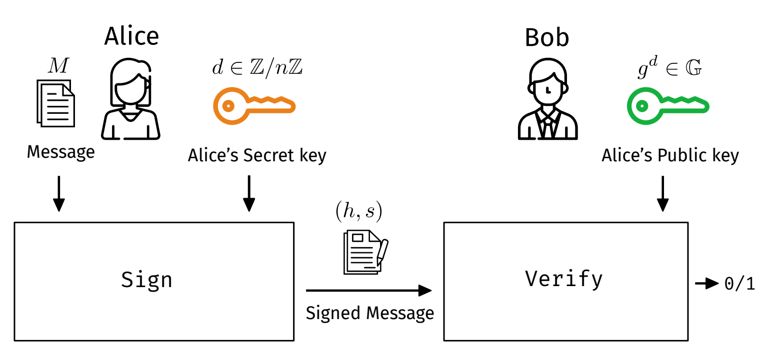
Motivating Example: Schnorr Signature Scheme

- One of the simplest and most widely-used digital signature schemes
- Most notable variant: (EC)DSA
- Secure in ROM if the discrete logarithm problem (DLP) is hard

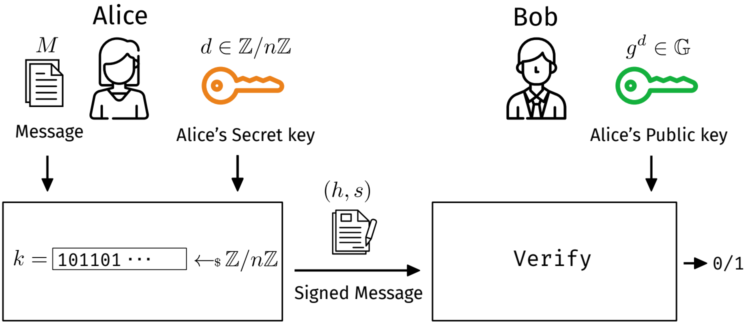
Motivating Example: Schnorr Signature Scheme

- One of the simplest and most widely-used digital signature schemes
- Most notable variant: (EC)DSA
- Secure in ROM if the discrete logarithm problem (DLP) is hard
- Relies on an ephemeral random value known as nonce

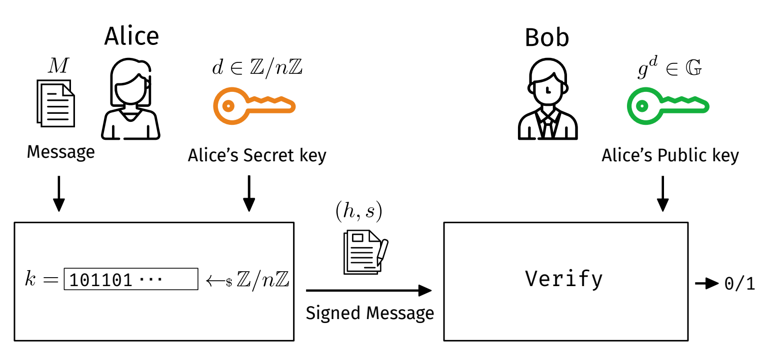
Nonce in Schnorr Signatures



Nonce in Schnorr Signatures



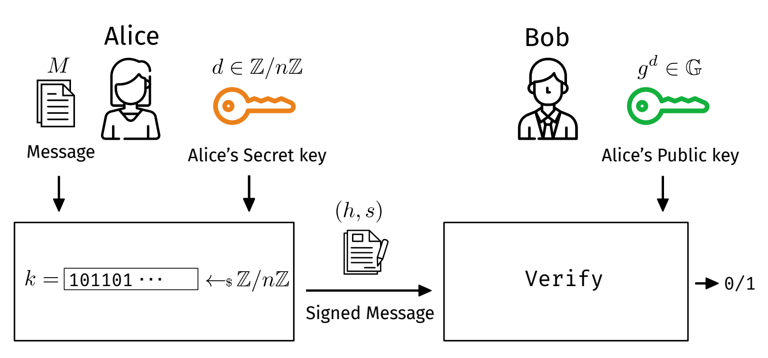
Nonce in Schnorr Signatures



- k is called nonce. It satisfies

$$k \equiv \underbrace{s}_{\text{public}} + \underbrace{h}_{\text{public}} d \mod n.$$

Nonce in Schnorr Signatures

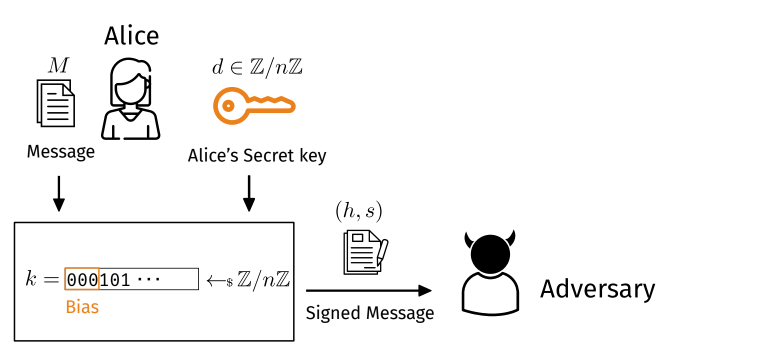


- k is called nonce. It satisfies

$$k \equiv \underbrace{s}_{\text{public}} + \underbrace{h}_{\text{public}} d \mod n.$$

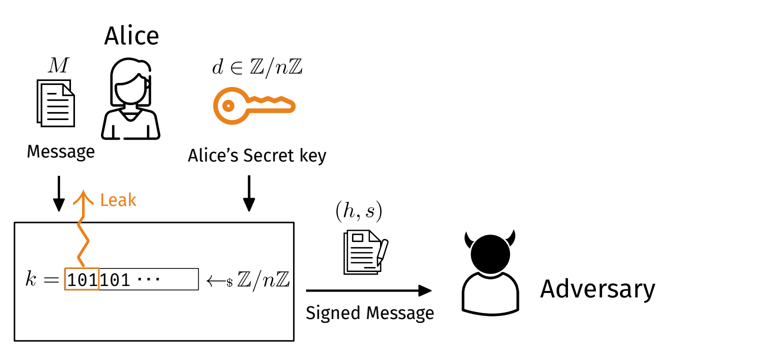
- k should NOT be reused/exposed

Risk of biased/leaky nonce



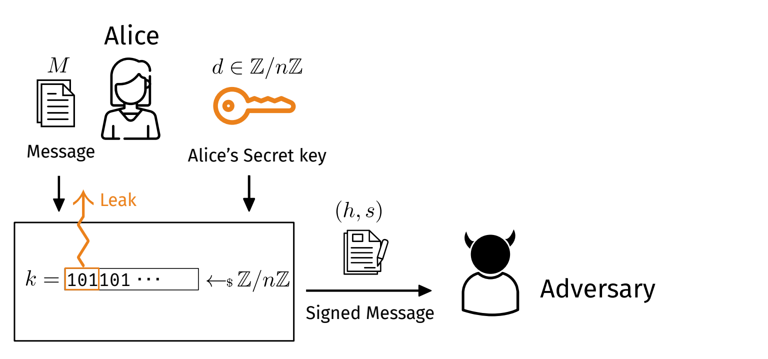
- But what if k is slightly biased ?

Risk of biased/leaky nonce



- But what if k is slightly biased or partially leaked?

Risk of biased/leaky nonce



- But what if k is slightly biased or partially leaked?

⇝ Adversary could bypass the (EC)DLP and steal the secret d by solving the hidden number problem (HNP)!

Nonce: very sensitive!

Our Contribution

1. Optimized a statistical attack framework, known as Bleichenbacher's attack, against nonces in Schnorr-like signatures

Our Contribution

1. Optimized a statistical attack framework, known as Bleichenbacher's attack, against nonces in Schnorr-like signatures
2. New fault attacks against recent, high-speed Schnorr-like signature scheme, qDSA, to obtain a few bits of nonces

Our Contribution

1. Optimized a statistical attack framework, known as Bleichenbacher's attack, against nonces in Schnorr-like signatures
2. New fault attacks against recent, high-speed Schnorr-like signature scheme, qDSA, to obtain a few bits of nonces
3. Implemented a full secret key recovery attack against Schnorr-like signatures
    - Over 252-bit group
    - Only 2 or 3-bit nonce leaks

We set new records!

	# Leaked bits of Nonces				
	1	2	3	4	5
384-bit	–	–	–	–	[DMHMP14]
252-bit	–	–	–	–	–
160-bit	[AFG+14]	[LN13]	[NS02]	–	–

Table 1: Comparison with previous published records

- Orange: Bleichenbacher's attack
- Others: Lattice attack

We set new records!

Leaked bits of Nonces

	1	2	3	4	5
384-bit	–	–	–	–	[DMHMP14]
252-bit	–	–	✓	–	–
160-bit	[AFG+14]	[LN13]	[NS02]	–	–

Table 1: Comparison with previous published records

- Orange: Bleichenbacher's attack
- Others: Lattice attack

We set new records!

	# Leaked bits of Nonces				
	1	2	3	4	5
384-bit	–	–	–	–	[DMHMP14]
252-bit	–	✓	✓	–	–
160-bit	[AFG+14]	[LN13]	[NS02]	–	–

Table 1: Comparison with previous published records

- Orange: Bleichenbacher's attack
- Others: Lattice attack

Optimizing Bleichenbacher's Attack

Bleichenbacher's Nonce Attack

- Originally proposed 18 years ago [Ble00], recently revisited by De Mulder et al. (CHES'13) and Aranha et al. (ASIACRYPT'14)

Bleichenbacher's Nonce Attack

- Originally proposed 18 years ago [Ble00], recently revisited by De Mulder et al. (CHES'13) and Aranha et al. (ASIACRYPT'14)
- Idea: quantify the nonce bias by defining "bias function" $B_n(K) \in [0, 1]$ and find the peak of it
    - $B_n(K) = 0$ if nonce is uniformly distributed over $\mathbb{Z}/n\mathbb{Z}$.
    - $B_n(K) \approx 1$ if nonce is biased.

Bleichenbacher's Nonce Attack

- Originally proposed 18 years ago [Ble00], recently revisited by De Mulder et al. (CHES'13) and Aranha et al. (ASIACRYPT'14)
- Idea: quantify the nonce bias by defining "bias function" $B_n(K) \in [0, 1]$ and find the peak of it
  - $B_n(K) = 0$ if nonce is uniformly distributed over $\mathbb{Z}/n\mathbb{Z}$.
  - $B_n(K) \approx 1$ if nonce is biased.
- Most important & costly phase is so-called range reduction of integers h

Bleichenbacher's Nonce Attack

- Originally proposed 18 years ago [Ble00], recently revisited by De Mulder et al. (CHES'13) and Aranha et al. (ASIACRYPT'14)
- Idea: quantify the nonce bias by defining "bias function" $B_n(K) \in [0,1]$ and find the peak of it
  - $B_n(K) = 0$ if nonce is uniformly distributed over $\mathbb{Z}/n\mathbb{Z}$.
  - $B_n(K) \approx 1$ if nonce is biased.
- Most important & costly phase is so-called range reduction of integers h
  - Necessary to detect the bias peak correctly and efficiently

Range Reduction Problem

Given: S signatures $(h_1, \ldots, h_S)$

Range Reduction Problem

Given: S signatures $(h_1, \ldots, h_S)$

Find: sufficiently many (say L) linear combinations

$$h'_j = \omega_{j,1} h_1 + \ldots + \omega_{j,S} h_S \quad \text{for} \quad 1 \leq j \leq L$$

such that

Range Reduction Problem

Given: S signatures $(h_1, \ldots, h_S)$

Find: sufficiently many (say L) linear combinations

$$h'_j = \omega_{j,1} h_1 + \ldots + \omega_{j,S} h_S \quad \text{for} \quad 1 \le j \le L$$

such that

- Small $h'_j < L$

Range Reduction Problem

Given: S signatures $(h_1, \ldots, h_S)$

Find: sufficiently many (say L) linear combinations

$$h'_j = \omega_{j,1} h_1 + \ldots + \omega_{j,S} h_S \quad \text{for} \quad 1 \leq j \leq L$$

such that

- Small $h'_j < L$
- Sparse coefficients $\Omega := \sum_i |\omega_{j,i}|$ s.t. $|B_n(\boldsymbol{K})|^{\Omega} > 1/\sqrt{L}$

Range Reduction Problem

Given: S signatures $(h_1, \ldots, h_S)$

Find: sufficiently many (say L) linear combinations

$$h'_j = \omega_{j,1} h_1 + \ldots + \omega_{j,S} h_S \quad \text{for} \quad 1 \leq j \leq L$$

such that

- Small $h'_j < L$
- Sparse coefficients $\Omega := \sum_i |\omega_{j,i}|$ s.t. $|B_n(\boldsymbol{K})|^{\Omega} > 1/\sqrt{L}$

Looks like knapsack?

Difference: find many linear combinations instead of a single exact knapsack solution

Our Approach: Schroeppel-Shamir Algorithm

- Previous approaches are not optimal if the nonce bias is small:
  - ☹ BKZ (De Mulder et al.): Coefficients are not sparse enough.
  - ☹ S&D (Aranha et al.): Requires many inputs, huge memory space.

Our Approach: Schroeppel-Shamir Algorithm

- Previous approaches are not optimal if the nonce bias is small:
  - ☹ BKZ (De Mulder et al.): Coefficients are not sparse enough.
  - ☹ S&D (Aranha et al.): Requires many inputs, huge memory space.
- We applied Schroeppel-Shamir knapsack algorithm [SS81]

Our Approach: Schroeppel-Shamir Algorithm

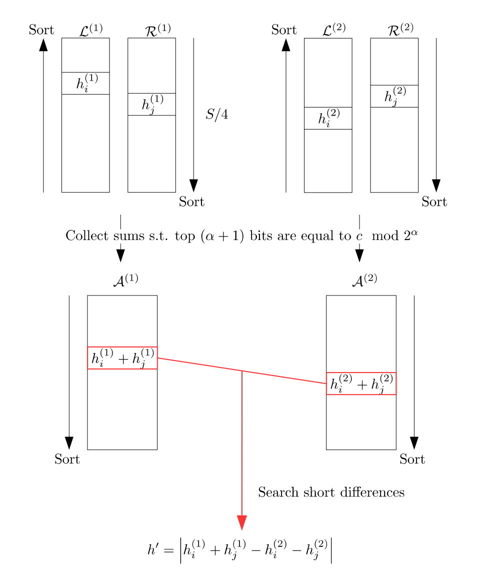
- Previous approaches are not optimal if the nonce bias is small:
  - ☹ BKZ (De Mulder et al.): Coefficients are not sparse enough.
  - ☹ S&D (Aranha et al.): Requires many inputs, huge memory space.
- We applied Schroeppel-Shamir knapsack algorithm [SS81]
- Mentioned by Bleichenbacher, but never examined in the literature

Our Approach: Schroeppel-Shamir Algorithm

- Previous approaches are not optimal if the nonce bias is small:
  - ☹ BKZ (De Mulder et al.): Coefficients are not sparse enough.
  - ☹ S&D (Aranha et al.): Requires many inputs, huge memory space.
- We applied Schroeppel-Shamir knapsack algorithm [SS81]
- Mentioned by Bleichenbacher, but never examined in the literature
- Advantages:
  - ☺ Highly space-efficient
  - ☺ Highly parallelizable with Howgrave-Graham–Joux's variant (EUROCRYPT'10, [HGJ10])

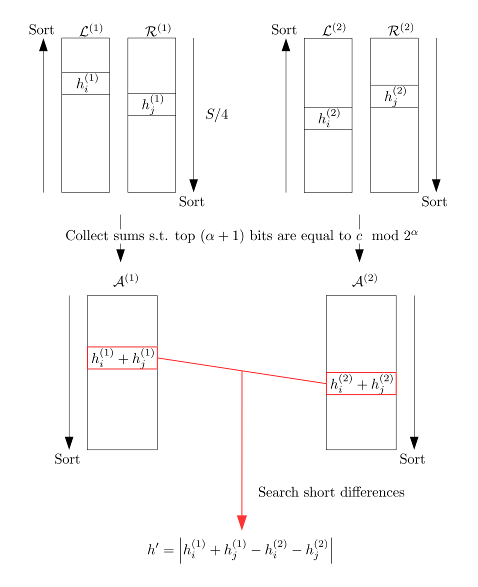
How HGJ–SS Helps

1. Split the inputs into four lists; sort.



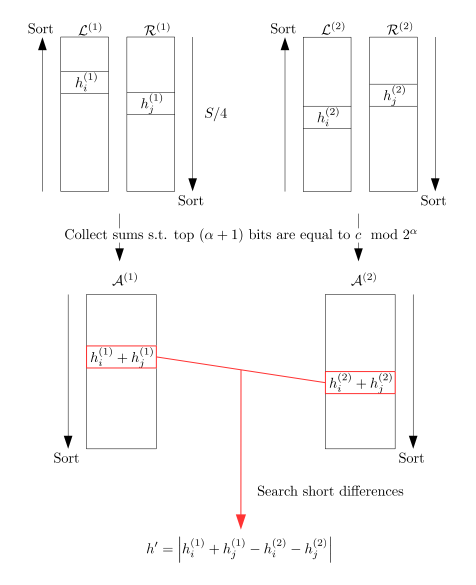
How HGJ–SS Helps

1. Split the inputs into four lists; sort.
2. Search for LC's of 2 whose top consecutive bits coincide with some fixed value; sort.



How HGJ–SS Helps

1. Split the inputs into four lists; sort.
2. Search for LC's of 2 whose top consecutive bits coincide with some fixed value; sort.
3. Take differences between values in two lists.
   → Get small LC's of 4 per round!



Complexity

Algorithm	Time	Space & # Sigs.
HGJ–SS (1 round)	$\widetilde{O}(S^{4/3})$	$O(S^{2/3})$
S&D (2 rounds)	$\widetilde{O}(S)$	$O(S)$

- Well-balanced time-space trade-offs

Complexity

Algorithm	Time	Space & # Sigs.
HGJ–SS (1 round)	$\widetilde{O}(S^{4/3})$	$O(S^{2/3})$
S&D (2 rounds)	$\widetilde{O}(S)$	$O(S)$

- Well-balanced time-space trade-offs
- HGJ–SS still terminates within a reasonable time frame due to parallelization

Fault Attacks on qDSA Signature

qDSA Signature over Curve25519

- qDSA: recent, high-speed variant of Schnorr signature by Renes and Smith (ASIACRYPT'17, [RS17])

qDSA Signature over Curve25519

- qDSA: recent, high-speed variant of Schnorr signature by Renes and Smith (ASIACRYPT'17, [RS17])
- Can be instantiated with Curve25519 Montgomery curve

qDSA Signature over Curve25519

- qDSA: recent, high-speed variant of Schnorr signature by Renes and Smith (ASIACRYPT'17, [RS17])
- Can be instantiated with Curve25519 Montgomery curve
- Signature generation computes

$$k \leftarrow H(M||d') \text{ // nonce}$$

$$\pm R \leftarrow \texttt{Ladder}(k, \pm P = (X : Z)) = \pm[k]P$$

qDSA Signature over Curve25519

- qDSA: recent, high-speed variant of Schnorr signature by Renes and Smith (ASIACRYPT'17, [RS17])
- Can be instantiated with Curve25519 Montgomery curve
- Signature generation computes

$$k \leftarrow H(M||d') \text{ // nonce}$$

$$\pm R \leftarrow \texttt{Ladder}(k, \pm P = (X : Z)) = \pm[k]P$$

- Attack idea:

qDSA Signature over Curve25519

- qDSA: recent, high-speed variant of Schnorr signature by Renes and Smith (ASIACRYPT'17, [RS17])
- Can be instantiated with Curve25519 Montgomery curve
- Signature generation computes

$$k \leftarrow H(M||d') \text{ // nonce}$$

$$\pm R \leftarrow \texttt{Ladder}(k, \pm P = (X : Z)) = \pm[k]P$$

- Attack idea:
  - Curve25519: $E(\mathbb{F}_p) \cong \mathbb{Z}/8\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$

qDSA Signature over Curve25519

- qDSA: recent, high-speed variant of Schnorr signature by Renes and Smith (ASIACRYPT'17, [RS17])
- Can be instantiated with Curve25519 Montgomery curve
- Signature generation computes

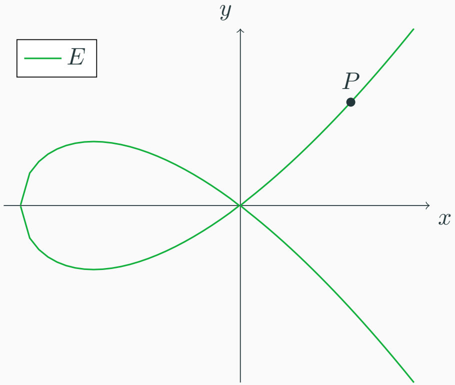
$$k \leftarrow H(M||d') \text{ // nonce}$$

$$\pm R \leftarrow \texttt{Ladder}(k, \pm P = (X : Z)) = \pm[k]P$$

- Attack idea:
  - Curve25519: $E(\mathbb{F}_p) \cong \mathbb{Z}/8\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$
  - By injecting a fault to the base point, we perturb it to non-prime/low-order points on Curve25519.

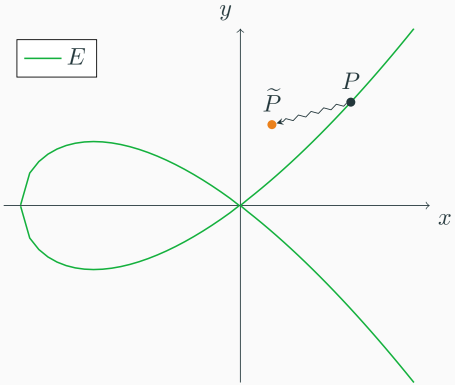
$$\pm\widetilde{R} \leftarrow \texttt{Ladder}(k, \pm\widetilde{P} = (\widetilde{X} : Z)) = \pm[k]\widetilde{P}$$

Observation



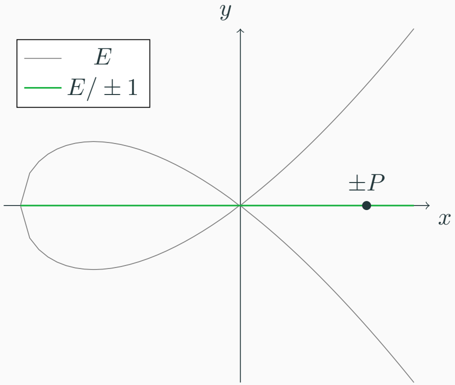
✗ EC-Schnorr/ECDSA uses y-coordinate $\rightsquigarrow$ perturbed point $\widetilde{P}$ is not likely on the original curve anymore

Observation



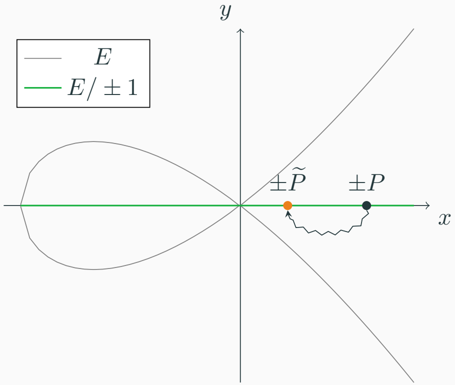
✗ EC-Schnorr/ECDSA uses y-coordinate $\rightsquigarrow$ perturbed point $\widetilde{P}$ is not likely on the original curve anymore

Observation



- ✗ EC-Schnorr/ECDSA uses y-coordinate $\rightsquigarrow$ perturbed point $\widetilde{P}$ is not likely on the original curve anymore
- ✓ qDSA makes use of x-only arithmetic $\rightsquigarrow$ perturbed point $\pm\widetilde{P}$ is necessarily on the curve or its twist!

Observation

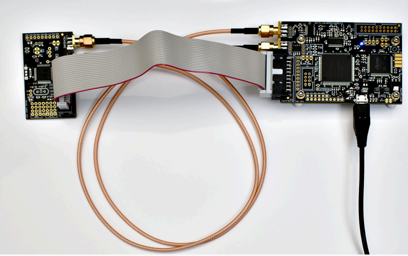


✗ EC-Schnorr/ECDSA uses y-coordinate $\rightsquigarrow$ perturbed point $\widetilde{P}$ is not likely on the original curve anymore

✓ qDSA makes use of x-only arithmetic $\rightsquigarrow$ perturbed point $\pm\widetilde{P}$ is necessarily on the curve or its twist!

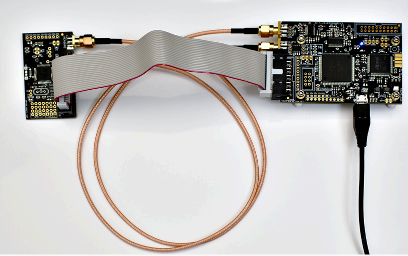
Fault Attacks on Curve25519 Base Point

1. Random semi-permanent fault against (program) memory
   ⇝ Can obtain 3-LSBs of nonce
2. Instruction skipping fault against base point initialization
   ⇝ Can obtain 2-LSBs of nonce
   - Verified using ChipWhisperer-Lite against AVR XMEGA



Fault Attacks on Curve25519 Base Point

1. Random semi-permanent fault against (program) memory
   $\rightsquigarrow$ Can obtain 3-LSBs of nonce
2. Instruction skipping fault against base point initialization
   $\rightsquigarrow$ Can obtain 2-LSBs of nonce
   - Verified using ChipWhisperer-Lite against AVR XMEGA



Countermeasure: multiply nonces by LCM of the the curve cofactor and the twist cofactor (i.e. "cofactor-killing")

$$\texttt{Ladder} : (8k, \pm\widetilde{P} = (\widetilde{X} : Z)) \mapsto \pm[8k]\widetilde{P}$$

Record-breaking Implementation of Nonce Attack

Result: Attack on 2-bit Leak

Wall clock time	CPU-time	Memory	# Sig	# MSB
16.7 days	11.7 years	15GB	2^{26}	26-bit

- Input: simulated 2^{45} faulty qDSA signatures, out of which 2^{26} instances (with $h <^{252-19}$) were fed into Bleichenbacher's attack

Result: Attack on 2-bit Leak

Wall clock time	CPU-time	Memory	# Sig	# MSB
16.7 days	11.7 years	15GB	2^{26}	26-bit

- Input: simulated 2^{45} faulty qDSA signatures, out of which 2^{26} instances (with $h <^{252-19}$) were fed into Bleichenbacher's attack
- Highly parallelized: 256 threads used (16 nodes $\times$ 16 vCPU)

Result: Attack on 2-bit Leak

Wall clock time	CPU-time	Memory	# Sig	# MSB
16.7 days	11.7 years	15GB	2^{26}	26-bit

- Input: simulated 2^{45} faulty qDSA signatures, out of which 2^{26} instances (with $h <^{252-19}$) were fed into Bleichenbacher's attack
- Highly parallelized: 256 threads used (16 nodes $\times$ 16 vCPU)

Result: Attack on 2-bit Leak

Wall clock time	CPU-time	Memory	# Sig	# MSB
16.7 days	11.7 years	15GB	2^{26}	26-bit

- Input: simulated 2^{45} faulty qDSA signatures, out of which 2^{26} instances (with $h <^{252-19}$) were fed into Bleichenbacher's attack
- Highly parallelized: 256 threads used (16 nodes $\times$ 16 vCPU)

Result: Attack on 2-bit Leak

Wall clock time	CPU-time	Memory	# Sig	# MSB
16.7 days	11.7 years	15GB	2^{26}	26-bit

- Input: simulated 2^{45} faulty qDSA signatures, out of which 2^{26} instances (with $h <^{252-19}$) were fed into Bleichenbacher's attack
- Highly parallelized: 256 threads used (16 nodes $\times$ 16 vCPU)

Result: Attack on 2-bit Leak

Wall clock time	CPU-time	Memory	# Sig	# MSB
16.7 days	11.7 years	15GB	2^{26}	26-bit

- Input: simulated 2^{45} faulty qDSA signatures, out of which 2^{26} instances (with $h <^{252-19}$) were fed into Bleichenbacher's attack
- Highly parallelized: 256 threads used (16 nodes $\times$ 16 vCPU)
- Recovered remaining bits of the secret key $<$ 6 hours

Result: Attack on 2-bit Leak

Wall clock time	CPU-time	Memory	# Sig	# MSB
16.7 days	11.7 years	15GB	2^{26}	26-bit

- Input: simulated 2^{45} faulty qDSA signatures, out of which 2^{26} instances (with $h <^{252-19}$) were fed into Bleichenbacher's attack
- Highly parallelized: 256 threads used (16 nodes $\times$ 16 vCPU)
- Recovered remaining bits of the secret key $<$ 6 hours
- Estimation shows S&D would require at least 2^{35} inputs ≈ 2TB RAM!

Result: Attack on 3-bit Leak

	Wall clock time	CPU-time	Memory	# Sig	# MSB
HGJ–SS	4.25 hours	238 hours	2.8GB	2^{23}	23-bit
S&D	0.75 hours	0.75 hours	128GB	2^{30}	21-bit

- 56 threads used (2 CPUs $\times$ 14 cores/CPU $\times$ 2 threads/core)

Result: Attack on 3-bit Leak

	Wall clock time	CPU-time	Memory	# Sig	# MSB
HGJ–SS	4.25 hours	238 hours	2.8GB	2^{23}	23-bit
S&D	0.75 hours	0.75 hours	128GB	2^{30}	21-bit

- 56 threads used (2 CPUs $\times$ 14 cores/CPU $\times$ 2 threads/core)

Result: Attack on 3-bit Leak

	Wall clock time	CPU-time	Memory	# Sig	# MSB
HGJ–SS	4.25 hours	238 hours	2.8GB	2^{23}	23-bit
S&D	0.75 hours	0.75 hours	128GB	2^{30}	21-bit

- 56 threads used (2 CPUs $\times$ 14 cores/CPU $\times$ 2 threads/core)
- The attack would be feasible using a small laptop!

Result: Attack on 3-bit Leak

	Wall clock time	CPU-time	Memory	# Sig	# MSB
HGJ–SS	4.25 hours	238 hours	2.8GB	2^{23}	23-bit
S&D	0.75 hours	0.75 hours	128GB	2^{30}	21-bit

- 56 threads used (2 CPUs $\times$ 14 cores/CPU $\times$ 2 threads/core)
- The attack would be feasible using a small laptop!
- Attacking with S&D is possible

Result: Attack on 3-bit Leak

	Wall clock time	CPU-time	Memory	# Sig	# MSB
HGJ–SS	4.25 hours	238 hours	2.8GB	2^{23}	23-bit
S&D	0.75 hours	0.75 hours	128GB	2^{30}	21-bit

- 56 threads used (2 CPUs $\times$ 14 cores/CPU $\times$ 2 threads/core)
- The attack would be feasible using a small laptop!
- Attacking with S&D is possible and faster

Result: Attack on 3-bit Leak

	Wall clock time	CPU-time	Memory	# Sig	# MSB
HGJ–SS	4.25 hours	238 hours	2.8GB	2^{23}	23-bit
S&D	0.75 hours	0.75 hours	128GB	2^{30}	21-bit

- 56 threads used (2 CPUs $\times$ 14 cores/CPU $\times$ 2 threads/core)
- The attack would be feasible using a small laptop!
- Attacking with S&D is possible and faster , but requires much more signatures and RAM

Wrap-up

Wrap-up

Contribution 1: Optimizing Bleichenbacher's attack

Wrap-up

Contribution 1: Optimizing Bleichenbacher’s attack

- Overcame the memory barrier of previous approach by applying knapsack algorithm.

Wrap-up

Contribution 1: Optimizing Bleichenbacher's attack

- Overcame the memory barrier of previous approach by applying knapsack algorithm.

Contribution 2: Fault attacks on qDSA over Curve25519

Wrap-up

Contribution 1: Optimizing Bleichenbacher's attack

- Overcame the memory barrier of previous approach by applying knapsack algorithm.

Contribution 2: Fault attacks on qDSA over Curve25519

- Discovered yet another situation where adversary could learn partial information of nonces.

Wrap-up

Contribution 1: Optimizing Bleichenbacher's attack

- Overcame the memory barrier of previous approach by applying knapsack algorithm.

Contribution 2: Fault attacks on qDSA over Curve25519

- Discovered yet another situation where adversary could learn partial information of nonces.
- Cofactor-killing is crucial when using x-only arithmetic.

Wrap-up

Contribution 1: Optimizing Bleichenbacher's attack

- Overcame the memory barrier of previous approach by applying knapsack algorithm.

Contribution 2: Fault attacks on qDSA over Curve25519

- Discovered yet another situation where adversary could learn partial information of nonces.
- Cofactor-killing is crucial when using x-only arithmetic.

Contribution 3: Implementation

Wrap-up

Contribution 1: Optimizing Bleichenbacher's attack

- Overcame the memory barrier of previous approach by applying knapsack algorithm.

Contribution 2: Fault attacks on qDSA over Curve25519

- Discovered yet another situation where adversary could learn partial information of nonces.
- Cofactor-killing is crucial when using x-only arithmetic.

Contribution 3: Implementation

- First large-scale parallelization of Bleichenbacher

Wrap-up

Contribution 1: Optimizing Bleichenbacher's attack

- Overcame the memory barrier of previous approach by applying knapsack algorithm.

Contribution 2: Fault attacks on qDSA over Curve25519

- Discovered yet another situation where adversary could learn partial information of nonces.
- Cofactor-killing is crucial when using x-only arithmetic.

Contribution 3: Implementation

- First large-scale parallelization of Bleichenbacher
- Set new records!

Thank you!

Dank je!

GitHub: `https://github.com/security-kouza/new-bleichenbacher-records`

By Akira Takahashi, Mehdi Tibouchi, and Masayuki Abe

References I

- Diego F. Aranha, Pierre-Alain Fouque, Benoit Gérard, Jean-Gabriel Kammerer, Mehdi Tibouchi, and Jean-Christophe Zapalowicz.
  GLV/GLS decomposition, power analysis, and attacks on ECDSA signatures with single-bit nonce bias.
  In T. Iwata and P. Sarkar, editors, *ASIACRYPT 2014*, volume 8873 of *LNCS*, pages 262–281. Springer, 2014.
- Daniel Bleichenbacher.
  On the generation of one-time keys in DL signature schemes.
  Presentation at IEEE P1363 working group meeting, 2000.

References II

Elke De Mulder, Michael Hutter, Mark E Marson, and Peter Pearson.
Using Bleichenbacher's solution to the hidden number problem to attack nonce leaks in 384-bit ECDSA: extended version.
Journal of Cryptographic Engineering, 4(1):33–45, 2014.

Freepik.
Icons made by Freepik from Flaticon.com is licensed by CC 3.0 BY.
`http://www.flaticon.com`.

Nick Howgrave-Graham and Antoine Joux.
New generic algorithms for hard knapsacks.
In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 235–256. Springer, 2010.

Mingjie Liu and Phong Q. Nguyen.
Solving BDD by enumeration: An update.
In *CT-RSA 2013*, volume 7779 of *LNCS*, pages 293–309. Springer, 2013.

Phong Q. Nguyen and Igor E. Shparlinski.
The insecurity of the digital signature algorithm with partially known nonces.
Journal of Cryptology, 15(3), 2002.

References IV

Joost Renes and Benjamin Smith.
qDSA: Small and secure digital signatures with curve-based Diffie-Hellman key pairs.
In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017*, volume 10625 of *LNCS*, pages 273–302. Springer, 2017.

Richard Schroeppel and Adi Shamir.
A $T = O(2^{n/2}), S = O(2^{n/4})$ algorithm for certain NP-complete problems.
SIAM Journal on Computing, 10(3):456–464, 1981.