LadderLeak

Breaking ECDSA with Less than One Bit of Nonce Leakage

ACM CCS '20

Diego F. Aranha[1] Felipe R. Novaes[2] Akira Takahashi[1] Mehdi Tibouchi[3] Yuval Yarom[4]

[1]DIGIT, Aarhus University, Denmark

[2]University of Campinas, Brazil

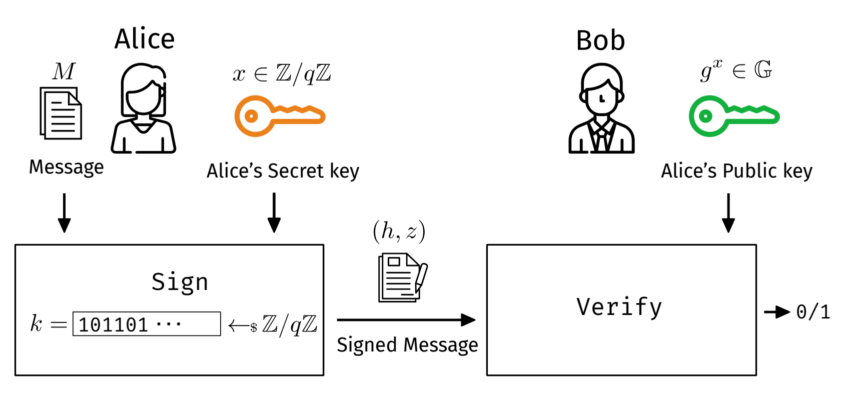
[3]NTT Corporation, Japan

[4]University of Adelaide and Data61, Australia

Attacks on ECDSA "nonce"

- ECDSA/Schnorr: Most popular signature schemes relying on the hardness of the (EC)DLP
- Signing operation involves **secret** randomness $k \in \mathbb{Z}_q$, sometimes called "**nonce**"
- Long history of research on the attacks against $k \ldots$

Randomness in ECDSA/Schnorr-type Schemes

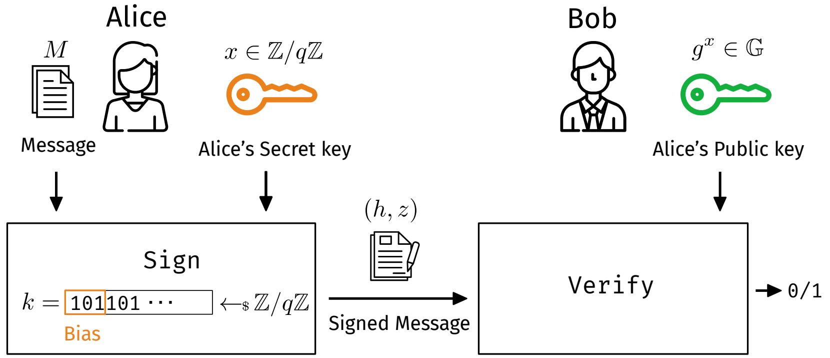


- k is a uniformly random value satisfying

$$k \equiv \underbrace{z}_{\text{public}} + \underbrace{h}_{\text{public}} \cdot x \mod q.$$

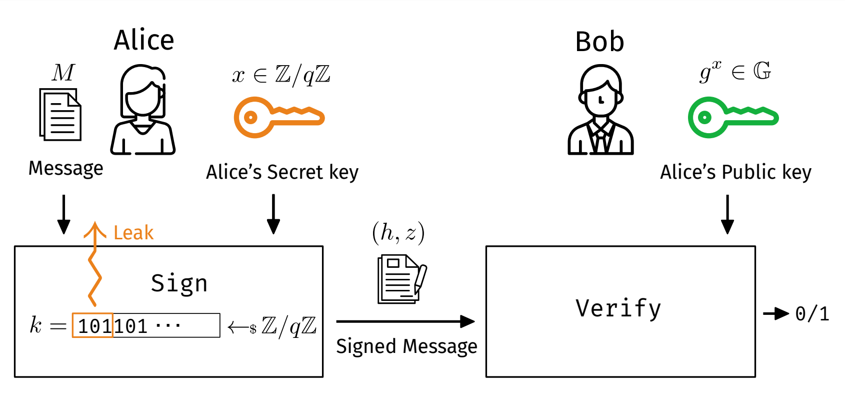
- k should **NEVER** be reused/exposed as $x = (z - z')/(h' - h) \mod q$

Risk of Biased/Leaky Randomness



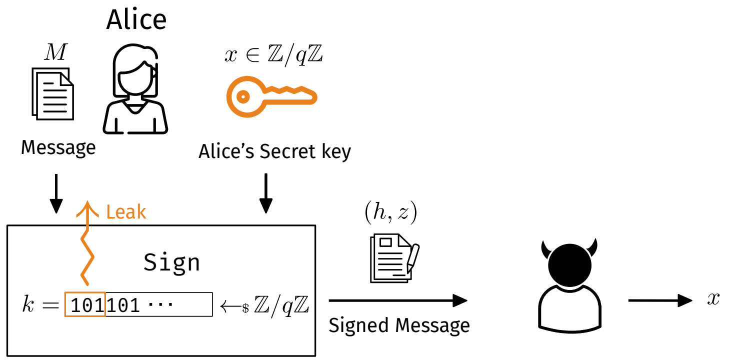
- What if k is slightly biased ?
- Secret key x is recovered by solving the **hidden number problem (HNP)**

Risk of Biased/Leaky Randomness



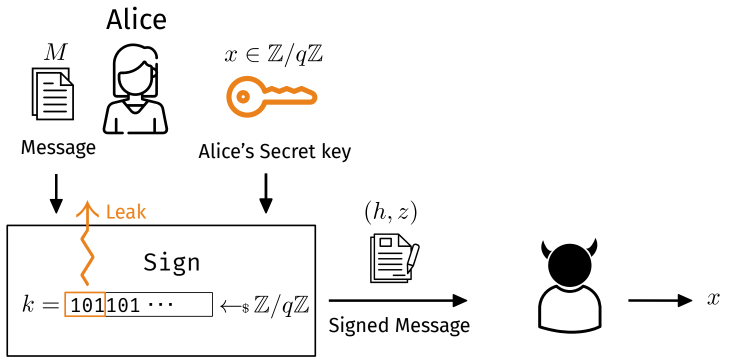
- What if k is slightly biased or partially leaked?
- Secret key x is recovered by solving the **hidden number problem (HNP)**

Risk of Biased/Leaky Randomness



- What if k is slightly biased or partially leaked? $\rightsquigarrow$ Attack!
- Secret key x is recovered by solving the **hidden number problem (HNP)**

Risk of Biased/Leaky Randomness



- What if k is slightly biased or partially leaked? $\rightsquigarrow$ Attack!
- Secret key x is recovered by solving the **hidden number problem (HNP)**

Randomness Failure in the Real World

- Poorly designed/implemented RNGs
- Predictable seed (`srand(time(0))`)
- VM resets ⇝ same snapshot will end up with the same seed
- Side-channel leakage
- and many more...

BBC | Sign in | News | Sport | Weather | Shop | Earth | Travel | Mo

NEWS

Home | UK | World | Business | Politics | Tech | Science | Health | Family & Education | E

Technology

iPhone hacker publishes secret Sony PlayStation 3 key

By Jonathan Fildes
Technology reporter, BBC News

6 January 2011

Share

The PlayStation 3's security has been broken by hackers, potentially allowing anyone to run any software - including pirated games - on the console.

A collective of hackers recently showed off a method that could force the system to reveal secret keys used to load

BBC news. 2011. https://www.bbc.com/news/technology-12116051

Randomness Failure in the Real World

- Poorly designed/implemented RNGs
- Predictable seed (`srand(time(0))`)
- VM resets ⇝ same snapshot will end up with the same seed
- Side-channel leakage
- and many more...


BBC Sign in News Sport Weather Shop Earth Travel

NEWS

Home UK World Business Politics Tech Science Health Family & Education

Technology

iPhone hacker publishes secret Sony PlayStation 3 key

By Jonathan Fildes
Technology reporter, BBC News

6 January 2011

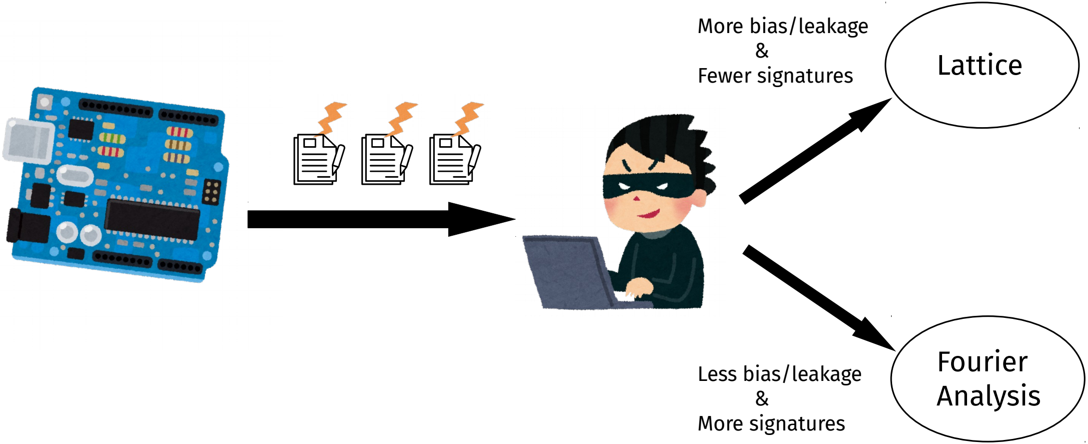
Share

The PlayStation 3's security has been broken by hackers, potentially allowing anyone to run any software - including pirated games - on the console.

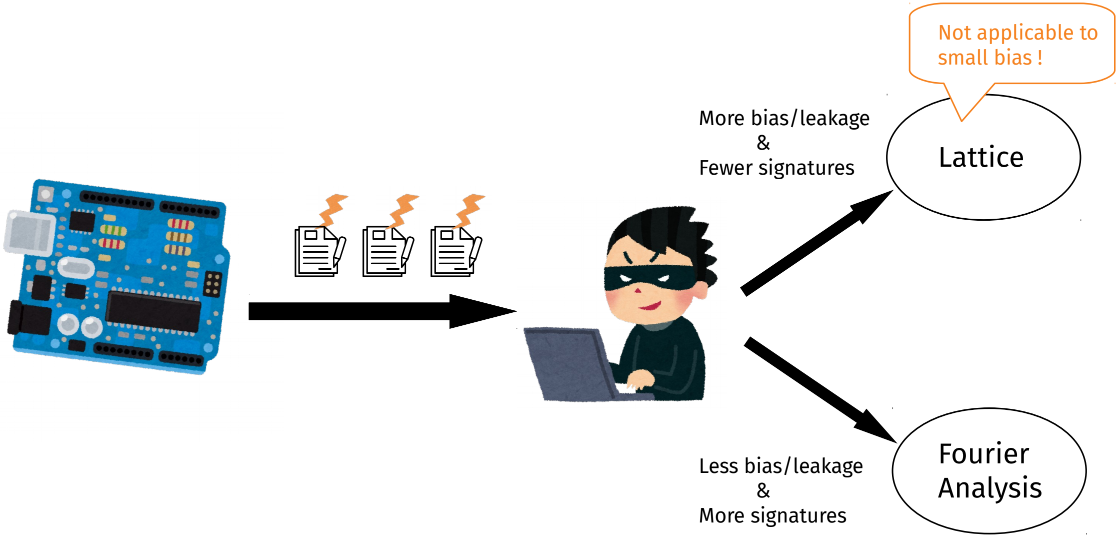
A collective of hackers recently showed off a method that could force the system to reveal secret keys used to load

BBC news. 2011. https://www.bbc.com/news/technology-12116051

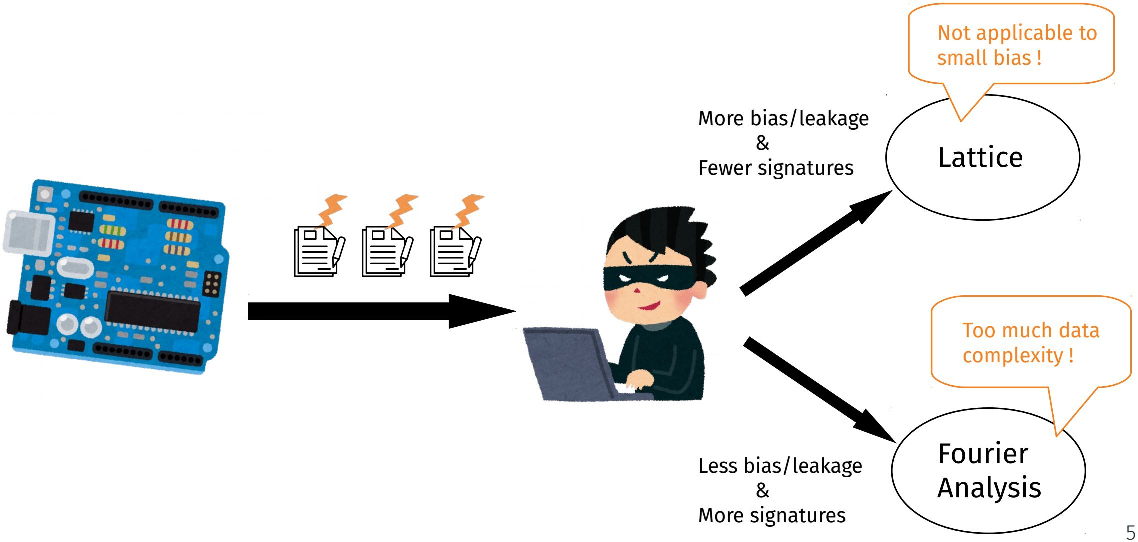
How to solve the HNP



How to solve the HNP



How to solve the HNP



Questions

- Can we reduce the data complexity of Fourier analysis-based attack?
- Can we attack even **less than 1-bit of nonce leakage** (= MSB is only leaked with prob. < 1)?
- Can we obtain such a small leakage from practical ECDSA implementations?

YES!

Questions

- Can we reduce the data complexity of Fourier analysis-based attack?
- Can we attack even **less than 1-bit of nonce leakage** (= MSB is only leaked with prob. < 1)?
- Can we obtain such a small leakage from practical ECDSA implementations?

YES!

Questions

- Can we reduce the data complexity of Fourier analysis-based attack?
- Can we attack even **less than 1-bit of nonce leakage** (= MSB is only leaked with prob. < 1)?
- Can we obtain such a small leakage from practical ECDSA implementations?

YES!

Questions

- Can we reduce the data complexity of Fourier analysis-based attack?
- Can we attack even **less than 1-bit of nonce leakage** (= MSB is only leaked with prob. < 1)?
- Can we obtain such a small leakage from practical ECDSA implementations?

YES!

Summary of results

1. Novel class of cache attacks against the Montgomery ladder scalar multiplication in OpenSSL `1.0.2u` and `1.1.0l`, and RELIC 0.4.0.
   - **Affected curves:** NIST P-192, P-224, P-256 (not by default in OpenSSL), P-384, P-521, B-283, K-283, K-409, B-571, `sect163r1`, `secp192k1`, `secp256k1`
2. Improved theoretical analysis of the Fourier analysis-based attack on the HNP (originally by Bleichenbacher)
   - Significantly reduced the required input data
   - Analysis in the presence of erroneous leakage information
3. Implemented a full secret key recovery attack against OpenSSL ECDSA over `sect163r1` and NIST P-192.

Summary of results

1. Novel class of cache attacks against the Montgomery ladder scalar multiplication in OpenSSL `1.0.2u` and `1.1.0l`, and RELIC 0.4.0.
    - **Affected curves:** NIST P-192, P-224, P-256 (not by default in OpenSSL), P-384, P-521, B-283, K-283, K-409, B-571, `sect163r1`, `secp192k1`, `secp256k1`
2. Improved theoretical analysis of the Fourier analysis-based attack on the HNP (originally by Bleichenbacher)
    - Significantly reduced the required input data
    - Analysis in the presence of erroneous leakage information
3. Implemented a full secret key recovery attack against OpenSSL ECDSA over `sect163r1` and NIST P-192.

Summary of results

1. Novel class of cache attacks against the Montgomery ladder scalar multiplication in OpenSSL `1.0.2u` and `1.1.0l`, and RELIC 0.4.0.
    - **Affected curves:** NIST P-192, P-224, P-256 (not by default in OpenSSL), P-384, P-521, B-283, K-283, K-409, B-571, `sect163r1`, `secp192k1`, `secp256k1`
2. Improved theoretical analysis of the Fourier analysis-based attack on the HNP (originally by Bleichenbacher)
    - Significantly reduced the required input data
    - Analysis in the presence of erroneous leakage information
3. Implemented a full secret key recovery attack against OpenSSL ECDSA over `sect163r1` and NIST P-192.

New attack records for the HNP!

Comparison with the previous records of solutions to the HNP: Fourier analysis vs Lattice

	< 1	1	2	3	4
256-bit	—	—	[TTA18]	[TTA18]	[Rya18, Rya19, MSEH19, WSBS20]
192-bit	This work	This work	—	—	—
160-bit	This work	This work (less data), [AFG+14, Ble05]	[Ble00][LN13]	[NS02]	—

- Require fewer input signatures to attack 160-bit HNP with 1-bit leak!
- First attack records for 192-bit HNP with (less than) 1-bit leak!

How to acquire ECDSA nonce

ECDSA signing

Scalar multiplication is critical for performance/security of ECC.

Algorithm 1 ECDSA signature generation

Input: $sk \in \mathbb{Z}_q$, $\texttt{msg} \in \{0,1\}^*$

Output: A valid signature (r, s)

1: $k \leftarrow_\$ \mathbb{Z}_q^*$
2: $R = (r_x, r_y) \leftarrow [k]P$
3: $r \leftarrow r_x \mod q$
4: $s \leftarrow (H(\texttt{msg}) + r \cdot sk)/k \mod q$
5: **return** (r, s)

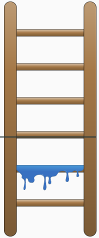
Critical: $[k]P$ should be **constant time** to avoid timing leakage about k.

LadderLeak: Tiny timing leakage from the Montgomery ladder

Algorithm 2 Montgomery ladder

Input: $P = (x, y)$, $k = (1, k_{t-2}, \ldots, k_1, k_0)$
Output: $Q = [k]P$

```
k' ← Select (k + q, k + 2q)
R_0 ← P, R_1 ← [2]P
for i ← lg(q) − 1 downto 0 do
    Swap (R_0, R_1) if k'_i = 0
    R_0 ← R_0 ⊕ R_1; R_1 ← 2R_1
    Swap (R_0, R_1) if k'_i = 0
end for
return Q = R_0
```



Conditions for the attack to work:

- Accumulators (R_0, R_1) are in **projective coordinates**, but initialized with the base point in **affine coordinates**.
- Group order is $2^n - \delta$
- Group law is non-constant time wrt handling Z coordinates $\rightsquigarrow$ **Weierstrass model**

Experiments were carried out with Flush+Reload cache attack technique

$\rightsquigarrow$ MSB of k was detected with > 99 % accuracy.

Software countermeasures & coordinated disclosure

There are **at least** three possible fixes:

1. **Randomize** Z coordinates at the beginning of scalar multiplication.
2. Implement group law in constant time, for example using **complete addition formulas** (no branches).
3. Implement ladder over co-Z arithmetic to **not handle** Z directly.

Coordinated disclosure: reported in December 2019 (before EOL of OpenSSL `1.0.2`), fixed in April 2020 with the first countermeasure.

How to exploit ECDSA nonce bias

Bleichenbacher's Attack: High-level Overview

- Step 1. Quantify the modular bias of randomness $k \leftarrow K$
  - $\text{Bias}_q(K) \approx 0$ if k is uniform in $\mathbb{Z}_q$
  - $\text{Bias}_q(K) \approx 1$ if k is biased in $\mathbb{Z}_q$
  - Contribution-1 Analyzed the behavior $\text{Bias}_q(K)$ when k's MSB is biased with probability < 1!
- Step 2. Find a candidate secret key which leads to the peak of $\text{Bias}_q(K)$ (by computing FFT)
- Critical intermediate step: collision search of integers h
  - Detect the bias peak correctly and efficiently
  - Contribution-2 Established **unified time-memory-data tradeoffs** by applying $\mathcal{K}$-list sum algorithm for the GBP!

Bleichenbacher's Attack: High-level Overview

- Step 1. Quantify the modular bias of randomness $k \leftarrow K$
  - $\text{Bias}_q(K) \approx 0$ if k is uniform in $\mathbb{Z}_q$
  - $\text{Bias}_q(K) \approx 1$ if k is biased in $\mathbb{Z}_q$
  - Contribution-1 Analyzed the behavior $\text{Bias}_q(K)$ when k's MSB is biased with probability < 1!

- Step 2. Find a candidate secret key which leads to the peak of $\text{Bias}_q(K)$ (by computing FFT)

- Critical intermediate step: collision search of integers h
  - Detect the bias peak correctly and efficiently
  - Contribution-2 Established **unified time-memory-data tradeoffs** by applying $\mathcal{K}$-list sum algorithm for the GBP!

Bleichenbacher's Attack: High-level Overview

- Step 1. Quantify the modular bias of randomness $k \leftarrow K$
  - $\text{Bias}_q(K) \approx 0$ if k is uniform in $\mathbb{Z}_q$
  - $\text{Bias}_q(K) \approx 1$ if k is biased in $\mathbb{Z}_q$
  - Contribution-1 Analyzed the behavior $\text{Bias}_q(K)$ when k's MSB is biased with probability < 1!
- Step 2. Find a candidate secret key which leads to the peak of $\text{Bias}_q(K)$ (by computing FFT)
- Critical intermediate step: collision search of integers h
  - Detect the bias peak correctly and efficiently
  - Contribution-2 Established **unified time-memory-data tradeoffs** by applying $\mathcal{K}$-list sum algorithm for the GBP!

Tradeoff Graphs for 1-bit Bias

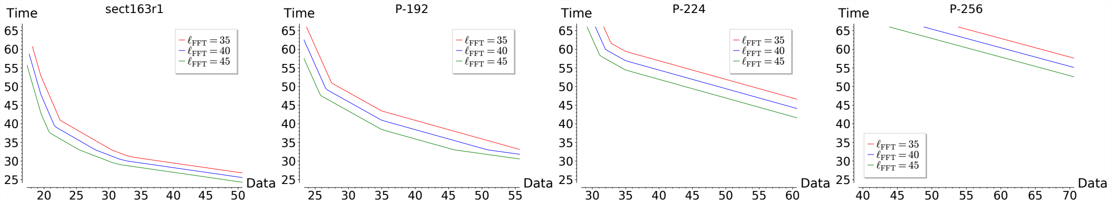


Figure 1: Time–Data tradeoffs when memory is fixed to 2^{35}.

* Optimized data complexity by solving the linear programming problem
* Paper has various tradeoff graphs and improved complexity estimates for 2-3 bits bias

Experimental Results on Full Key Recovery

Target	Facility	Error rate	Input	Output	Thread (Collision)	Time (Collision)	RAM (Collision)	L_{FFT}	Recovered MSBs
NIST P-192	AWS EC2	0	2^{29}	2^{29}	96×24	113h	492GB	2^{38}	39
NIST P-192	AWS EC2	1%	2^{35}	2^{30}	96×24	52h	492GB	2^{37}	39
`sect163r1`	Cluster	0	2^{23}	2^{27}	16×16	7h	80GB	2^{35}	36
`sect163r1`	Workstation	2.7%	2^{24}	2^{29}	48	42h	250GB	2^{34}	35

- Attack on **P-192** is made possible by our highly optimized parallel implementation.
- Attack on `sect163r1` is even feasible with a laptop.
- Recovering remaining bits is much cheaper in Bleichenbacher's framework.
- Attacks on P-224 with 1-bit bias or P-256 with 2-bit bias are also tractable.

Main takeaways

- Securely implementing brittle cryptographic algorithms is still **hard**.
- **Don't** underestimate even less than 1-bit of nonce leakage!
- Interesting connection between the HNP and GBP (from symmetric key crypto)
- Open questions:
    - More list sum algorithms and tradeoffs?
    - Improvements to FFT computation?
    - Other sources of small leakage?

Thank you! & Questions?

More details at `https://ia.cr/2020/615`

Main takeaways

- Securely implementing brittle cryptographic algorithms is still **hard**.
- **Don't** underestimate even less than 1-bit of nonce leakage!
- Interesting connection between the HNP and GBP (from symmetric key crypto)
- Open questions:
  - More list sum algorithms and tradeoffs?
  - Improvements to FFT computation?
  - Other sources of small leakage?

Thank you! & Questions?

More details at `https://ia.cr/2020/615`

Main takeaways

- Securely implementing brittle cryptographic algorithms is still **hard**.
- **Don't** underestimate even less than 1-bit of nonce leakage!
- Interesting connection between the HNP and GBP (from symmetric key crypto)
- Open questions:
  - More list sum algorithms and tradeoffs?
  - Improvements to FFT computation?
  - Other sources of small leakage?

Thank you! & Questions?

More details at `https://ia.cr/2020/615`

Main takeaways

- Securely implementing brittle cryptographic algorithms is still **hard**.
- **Don't** underestimate even less than 1-bit of nonce leakage!
- Interesting connection between the HNP and GBP (from symmetric key crypto)
- Open questions:
  - More list sum algorithms and tradeoffs?
  - Improvements to FFT computation?
  - Other sources of small leakage?

Thank you! & Questions?

More details at `https://ia.cr/2020/615`

Main takeaways

- Securely implementing brittle cryptographic algorithms is still **hard**.
- **Don't** underestimate even less than 1-bit of nonce leakage!
- Interesting connection between the HNP and GBP (from symmetric key crypto)
- Open questions:
  - More list sum algorithms and tradeoffs?
  - Improvements to FFT computation?
  - Other sources of small leakage?

Thank you! & Questions?

More details at `https://ia.cr/2020/615`

Diego F. Aranha, Pierre-Alain Fouque, Benoît 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 Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 262–281. Springer, Heidelberg, December 2014.

Daniel Bleichenbacher.
On the generation of one-time keys in DL signature schemes.
Presentation at IEEE P1363 working group meeting, 2000.

References ii

Daniel Bleichenbacher.
Experiments with DSA.
Rump session at CRYPTO 2005, 2005.
Available from https://www.iacr.org/conferences/crypto2005/r/3.pdf.

Freepik.
Icons made by Freepik from Flaticon.com.
http://www.flaticon.com.

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

Daniel Moghimi, Berk Sunar, Thomas Eisenbarth, and Nadia Heninger.
TPM-FAIL: TPM meets timing and lattice attacks.
CoRR, abs/1911.05673, 2019.
To appear at USENIX Security 2020.

Phong Q. Nguyen and Igor Shparlinski.
The insecurity of the digital signature algorithm with partially known nonces.
Journal of Cryptology, 15(3):151–176, June 2002.

Keegan Ryan.
Return of the hidden number problem.
IACR TCHES, 2019(1):146–168, 2018.
https://tches.iacr.org/index.php/TCHES/article/view/7337.

Keegan Ryan.
Hardware-backed heist: Extracting ECDSA keys from qualcomm's TrustZone.
In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 181–194. ACM Press, November 2019.

References v

- Akira Takahashi, Mehdi Tibouchi, and Masayuki Abe.
  New Bleichenbacher records: Fault attacks on qDSA signatures.
  IACR TCHES, 2018(3):331–371, 2018.
  https://tches.iacr.org/index.php/TCHES/article/view/7278.
- Samuel Weiser, David Schrammel, Lukas Bodner, and Raphael Spreitzer.
  Big Numbers - Big Troubles: Systematically analyzing nonce leakage in (EC)DSA implementations.
  In *USENIX Security 2020)*, Boston, MA, August 2020. USENIX Association.