

Fault Attacks Against Lattice-Based Signatures

Thomas Espitau[†]
Pierre-Alain Fouque
Benoit Gérard
Mehdi Tibouchi

[†]Lip6, Sorbonne Universités, a.k.a. [Here](#)

July 5, 2016
Workshop HEAT

Towards postquantum cryptography

- ▶ Quantum computers **would** break all currently deployed public-key crypto: RSA, discrete logs, elliptic curves
- ▶ Agencies warnings
 - ▶ NSA deprecating Suite B (elliptic curves)
 - ▶ NIST starting postquantum competition

Towards postquantum cryptography

- ▶ **In theory**, plenty of schemes quantum-resistant
 - ▶ Code-based, hash trees, multivariate crypto, isogenies...
 - ▶ Almost everything possible with **lattices**

- ▶ **In practice**, very few actual implementations
 - ▶ Secure parameters often unclear
 - ▶ Concrete software/hardware implementation papers quite rare
 - ▶ Almost no consideration for **implementation attacks**

- ▶ Serious issue for practical postquantum crypto

Implementations of lattice-based schemes (I)

- ▶ Implementation of lattice-based crypto:

Limited and mostly academic

- ▶ One scheme has “industry” backing and quite a bit of code:
NTRU
 - ▶ **NTRUEncrypt**, ANSI standard, believed to be okay
 - ▶ **NTRUSign** is a trainwreck that has been patched and broken

Implementations of lattice-based schemes (II)

- ▶ In terms of practical schemes, other than NTRU, main efforts on signatures
 - ▶ **GLP**: improvement of Lyubashevsky signatures, efficient in SW and HW (**CHES'12**)
 - ▶ **BLISS**: improvement of GLP, even better (**CRYPTO'13**, **CHES'14**)
 - ▶ **GPV**: in particular variant of Ducas, Lyubashevsky, Prest NTRU-based IBE (**AC'14**),
 - ▶ **PASSSign** (**ACNS'14**), **TESLA** (**LATINCRYPT 14**),...

Implementation attacks

Break a provably security cryptographic scheme:

solve a hard computational problem



Break an **implementation**

potentially bypass security proof

“PROBLEM EXISTS BETWEEN KEYBOARD AND CHAIR”

Implementation attacks

- ▶ **Side-channel attacks:** *Passive physical attacks, exploiting information leakage*
 - ▶ Timing attacks, power analysis, EM attacks, cache attacks, acoustic attacks...

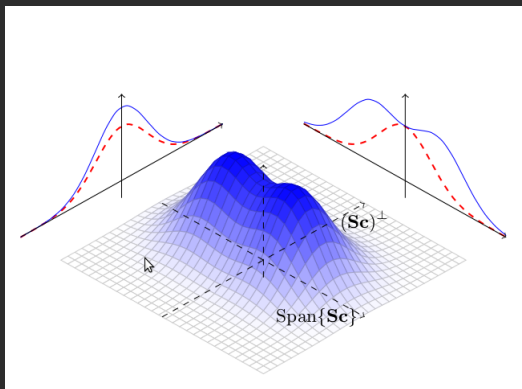
- ▶ **Fault attacks:** *Active physical attacks, extract secret information by tampering with the device to cause errors*
 - ▶ Faults on memory: lasers, x-rays...
 - ▶ Faults on computation: variations in supply voltage, external clock, temperature...

BLISS: the basics

- ▶ Introduced by *Ducas, Durmus, Lepoint and Lyubashevsky* at **CRYPTO'13**
- ▶ Improvement of Ring-SIS-based scheme of Lyubashevsky
- ▶ Still kind of “Fiat–Shamir signatures”

BLISS: the basics

- ▶ Defined over $\mathcal{R} = \mathbb{Z}[\mathbf{x}]/(\mathbf{x}^n + 1)$
- ▶ Main improvement: Reduce the size of parameters by **Bimodal Gaussian** distributions



BLISS: the basics

- ▶ Defined over $\mathcal{R} = \mathbb{Z}[\mathbf{x}]/(\mathbf{x}^n + 1)$
- ▶ Main improvement: Reduce the size of parameters by **Bimodal Gaussian** distributions



Distributio Camelus bactrianus

BLISS: key generation

- 1: **function** KEYGEN()
- 2: choose \mathbf{f}, \mathbf{g} as uniform polynomials with exactly $d_1 = \lceil \delta_1 n \rceil$ entries in $\{\pm 1\}$ and $d_2 = \lceil \delta_2 n \rceil$ entries in $\{\pm 2\}$
- 3: $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2)^T \leftarrow (\mathbf{f}, 2\mathbf{g} + \mathbf{1})^T$
- 4: **if** $N_\kappa(\mathbf{S}) \geq C^2 \cdot 5 \cdot (\lceil \delta_1 n \rceil + 4\lceil \delta_2 n \rceil) \cdot \kappa$ **then restart**
- 5: **if** \mathbf{f} is not invertible **then restart**
- 6: $\mathbf{a}_q = (2\mathbf{g} + \mathbf{1})/\mathbf{f} \bmod q$
- 7: **return** $(pk = \mathbf{A}, sk = \mathbf{S})$ where
 $\mathbf{A} = (\mathbf{a}_1 = 2\mathbf{a}_q, q - 2) \bmod 2q$
- 8: **end function**

BLISS: signature

- 1: **function** SIGN($\mu, pk = \mathbf{A}, sk = \mathbf{S}$)
- 2: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_{\mathbb{Z}, \sigma}^n$ ▷ Gaussian sampling
- 3: $\mathbf{u} = \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q$ ▷ $\zeta = 1/(q - 2)$
- 4: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rfloor_d \bmod p, \mu)$ ▷ special hashing
- 5: choose a random bit b
- 6: $\mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^{b_{s_1}} \mathbf{c}$
- 7: $\mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^{b_{s_2}} \mathbf{c}$
- 8: **continue** with probability
 $1 / (M \exp(-\|\mathbf{S}\mathbf{c}\| / (2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{S}\mathbf{c} \rangle / \sigma^2))$ otherwise **restart**
- 9: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rfloor_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rfloor_d) \bmod p$
- 10: **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
- 11: **end function**

BLISS: verification

- 1: **function** **VERIFY**($\mu, \mathbf{A}, (\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$)
- 2: **if** $\|(\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger)\|_2 > B_2$ **then** reject
- 3: **if** $\|(\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger)\|_\infty > B_\infty$ **then** reject
- 4: **accept iff** $\mathbf{c} = H(\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot \mathbf{q} \cdot \mathbf{c} \rfloor_d + \mathbf{z}_2^\dagger \bmod p, \mu)$
- 5: **end function**

Let's break things!



BLISS: signature

- 1: **function** **SIGN**($\mu, pk = \mathbf{A}, sk = \mathbf{S}$)
- 2: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_{\mathbb{Z}, \sigma}^n$ ▷ Gaussian sampling
- 3: $\mathbf{u} = \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q$ ▷ $\zeta = 1/(q - 2)$
- 4: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rfloor_d \bmod p, \mu)$ ▷ special hashing
- 5: choose a random bit b
- 6: $\mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^{b_{s_1}} \mathbf{c}$
- 7: $\mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^{b_{s_2}} \mathbf{c}$
- 8: **continue** with probability
 $1 / (M \exp(-\|\mathbf{S}\mathbf{c}\| / (2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{S}\mathbf{c} \rangle / \sigma^2))$ otherwise **restart**
- 9: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rfloor_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rfloor_d) \bmod p$
- 10: **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
- 11: **end function**

BLISS: signature

- 1: **function** SIGN($\mu, pk = \mathbf{A}, sk = \mathbf{S}$)
- 2: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_{\mathbb{Z}, \sigma}^n$ ▷ Gaussian sampling
- 3: $\mathbf{u} = \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q$ ▷ $\zeta = 1/(q - 2)$
- 4: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rfloor_d \bmod p, \mu)$ ▷ special hashing
- 5: **choose a random bit** b
- 6: $\mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$
- 7: $\mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
- 8: **continue** with probability
 $1 / (M \exp(-\|\mathbf{S}\mathbf{c}\| / (2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{S}\mathbf{c} \rangle / \sigma^2))$ otherwise **restart**
- 9: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rfloor_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rfloor_d) \bmod p$
- 10: **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
- 11: **end function**

BLISS: signature

- 1: **function** SIGN($\mu, pk = \mathbf{A}, sk = \mathbf{S}$)
 - 2: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_{\mathbb{Z}, \sigma}^n$
 - 3: $\mathbf{u} = \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q$
 - 4: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rfloor_d \bmod p, \mu)$
 - 5: choose a random bit b
 - 6: $\mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^{b_{s_1}} \mathbf{c}$
 - 7: $\mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^{b_{s_2}} \mathbf{c}$
 - 8: **continue** with probability
 $1 / (M \exp(-\|\mathbf{S}\mathbf{c}\| / (2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{S}\mathbf{c} \rangle / \sigma^2))$ otherwise **restart**
 - 9: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rfloor_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rfloor_d) \bmod p$
 - 10: **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
 - 11: **end function**
- ▷ Gaussian sampling
 - ▷ $\zeta = 1/(q - 2)$
 - ▷ special hashing

Attacking y

- ▶ y_1 (\equiv discrete Gaussian) \approx additive mask in

$$z_1 \equiv y_1 + (-1)^b s_1 c \pmod{q}$$

- ▶ Sampling: coefficient by coefficient
- ▶ Use fault injection to abort the sampling early \implies faulty signature with a low-degree y_1
- ▶ Done by attacking:
 - ▶ Branching test of the loop (voltage spike, clock variation...)
 - ▶ Contents of the loop counter (lasers, x-rays...)

Attacking y

- ▶ y_1 (\equiv discrete Gaussian) \approx additive mask in

$$z_1 \equiv y_1 + (-1)^b s_1 c \pmod{q}$$

- ▶ Sampling: coefficient by coefficient
- ▶ Use fault injection to abort the sampling early \implies faulty signature with a low-degree y_1
- ▶ Done by attacking:
 - ▶ Branching test of the loop (voltage spike, clock variation...)
 - ▶ Contents of the loop counter (lasers, x-rays...)

Attacking y

- ▶ y_1 (\equiv discrete Gaussian) \approx additive mask in

$$z_1 \equiv y_1 + (-1)^b s_1 c \pmod{q}$$

- ▶ Sampling: coefficient by coefficient
- ▶ Use fault injection to abort the sampling early \implies faulty signature with a low-degree y_1
- ▶ Done by attacking:
 - ▶ Branching test of the loop (voltage spike, clock variation...)
 - ▶ Contents of the loop counter (lasers, x-rays...)

Attacking y

- ▶ y_1 (\equiv discrete Gaussian) \approx additive mask in

$$z_1 \equiv y_1 + (-1)^b s_1 c \pmod{q}$$

- ▶ Sampling: coefficient by coefficient
- ▶ Use fault injection to abort the sampling early \implies faulty signature with a low-degree y_1
- ▶ Done by attacking:
 - ▶ Branching test of the loop (voltage spike, clock variation...)
 - ▶ Contents of the loop counter (lasers, x-rays...)

Attack details (I)

- ▶ Signature generated with \mathbf{y}_1 of degree $m \ll n$
- ▶ If \mathbf{c} invertible (probability $(1 - 1/q)^n \approx 96\%$):

$$\mathbf{v} = \mathbf{c}^{-1} \mathbf{z}_1 \equiv \mathbf{c}^{-1} \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \pmod{q}$$

WLOG, $b = 0$ (equivalent keys)

- ▶ \mathbf{s}_1 is short $\implies \mathbf{v}$ very close to lattice

$$L = \text{Span}(q\mathbb{Z}^n, \mathbf{w}_i = \mathbf{c}^{-1} \mathbf{x}^i)$$

- ▶ $\dim(L) = n$ too large to apply lattice reduction

Same relation on subset of coefficients: $(i_1, \dots, i_m) \in \{1, \dots, n\}^m$

Attack details (I)

- ▶ Signature generated with \mathbf{y}_1 of degree $m \ll n$
- ▶ If \mathbf{c} invertible (probability $(1 - 1/q)^n \approx 96\%$):

$$\mathbf{v} = \mathbf{c}^{-1}\mathbf{z}_1 \equiv \mathbf{c}^{-1}\mathbf{y}_1 + (-1)^b\mathbf{s}_1 \pmod{q}$$

WLOG, $b = 0$ (equivalent keys)

- ▶ \mathbf{s}_1 is short $\implies \mathbf{v}$ very close to lattice

$$L = \text{Span}(q\mathbb{Z}^n, \mathbf{w}_j = \mathbf{c}^{-1}\mathbf{x}^j)$$

- ▶ $\dim(L) = n$ too large to apply lattice reduction

Same relation on subset of coefficients: (i_1, \dots, i_m)

Attack details (I)

- ▶ Signature generated with \mathbf{y}_1 of degree $m \ll n$
- ▶ If \mathbf{c} invertible (probability $(1 - 1/q)^n \approx 96\%$):

$$\mathbf{v} = \mathbf{c}^{-1}\mathbf{z}_1 \equiv \mathbf{c}^{-1}\mathbf{y}_1 + (-1)^b\mathbf{s}_1 \pmod{q}$$

WLOG, $b = 0$ (equivalent keys)

- ▶ \mathbf{s}_1 is short $\implies \mathbf{v}$ very close to lattice

$$L = \text{Span}(q\mathbb{Z}^n, \mathbf{w}_j = \mathbf{c}^{-1}\mathbf{x}^j)$$

- ▶ $\dim(L) = n$ too large to apply lattice reduction

Same relation on subset of coefficients: $\mathbf{w}_j = (\mathbf{c}^{-1}\mathbf{x}^j)_1$

Attack details (I)

- ▶ Signature generated with \mathbf{y}_1 of degree $m \ll n$
- ▶ If \mathbf{c} invertible (probability $(1 - 1/q)^n \approx 96\%$):

$$\mathbf{v} = \mathbf{c}^{-1}\mathbf{z}_1 \equiv \mathbf{c}^{-1}\mathbf{y}_1 + (-1)^b \mathbf{s}_1 \pmod{q}$$

WLOG, $b = 0$ (equivalent keys)

- ▶ \mathbf{s}_1 is short $\implies \mathbf{v}$ very close to lattice

$$L = \text{Span}(q\mathbb{Z}^n, \mathbf{w}_i = \mathbf{c}^{-1}\mathbf{x}^i)$$

- ▶ $\dim(L) = n$ too large to apply lattice reduction

Same relation on subset of coefficients: REDUCE THE DIM

Attack details (I)

- ▶ Signature generated with \mathbf{y}_1 of degree $m \ll n$
- ▶ If \mathbf{c} invertible (probability $(1 - 1/q)^n \approx 96\%$):

$$\mathbf{v} = \mathbf{c}^{-1}\mathbf{z}_1 \equiv \mathbf{c}^{-1}\mathbf{y}_1 + (-1)^b \mathbf{s}_1 \pmod{q}$$

WLOG, $b = 0$ (equivalent keys)

- ▶ \mathbf{s}_1 is short $\implies \mathbf{v}$ very close to lattice

$$L = \text{Span}(q\mathbb{Z}^n, \mathbf{w}_i = \mathbf{c}^{-1}\mathbf{x}^i)$$

- ▶ $\dim(L) = n$ too large to apply lattice reduction

Same relation on subset of coefficients: REDUCE THE DIM

Attack details (I)

- ▶ Signature generated with \mathbf{y}_1 of degree $m \ll n$
- ▶ If \mathbf{c} invertible (probability $(1 - 1/q)^n \approx 96\%$):

$$\mathbf{v} = \mathbf{c}^{-1}\mathbf{z}_1 \equiv \mathbf{c}^{-1}\mathbf{y}_1 + (-1)^b \mathbf{s}_1 \pmod{q}$$

WLOG, $b = 0$ (equivalent keys)

- ▶ \mathbf{s}_1 is short $\implies \mathbf{v}$ very close to lattice

$$L = \text{Span}(q\mathbb{Z}^n, \mathbf{w}_i = \mathbf{c}^{-1}\mathbf{x}^i)$$

- ▶ $\dim(L) = n$ too large to apply lattice reduction

Same relation on subset of coefficients: **REDUCE THE DIM**

Attack details (II)

- ▶ Subset $I \subset \{0, \dots, n-1\}$ of cardinal ℓ $\varphi_I: \mathbb{Z}^n \rightarrow \mathbb{Z}^I$ projection
- ▶ $\varphi_I(\mathbf{v})$ close to the lattice generated by $\varphi_I(\mathbf{w}_j)$ and $q\mathbb{Z}^I$
If ℓ large enough, difference should be $\varphi_I(\mathbf{s}_1)$.
- ▶ CVP using Babai nearest plane algorithm. Condition on ℓ to recover $\varphi_I(\mathbf{s}_1)$:

$$\ell + 1 \gtrsim \frac{m + 2 + \frac{\log \sqrt{\delta_1 + 4\delta_2}}{\log q}}{1 - \frac{\log \sqrt{2\pi e(\delta_1 + 4\delta_2)}}{\log q}}$$

- ▶ For BLISS-I and BLISS-II, $\ell \approx 1.09 \cdot m$

Attack details (II)

- ▶ Subset $I \subset \{0, \dots, n-1\}$ of cardinal ℓ $\varphi_I: \mathbb{Z}^n \rightarrow \mathbb{Z}^I$ projection
- ▶ $\varphi_I(\mathbf{v})$ close to the lattice generated by $\varphi_I(\mathbf{w}_i)$ and $q\mathbb{Z}^I$
If ℓ large enough, difference should be $\varphi_I(\mathbf{s}_1)$.
- ▶ CVP using Babai nearest plane algorithm. Condition on ℓ to recover $\varphi_I(\mathbf{s}_1)$:

$$\ell + 1 \gtrsim \frac{m + 2 + \frac{\log \sqrt{\delta_1 + 4\delta_2}}{\log q}}{1 - \frac{\log \sqrt{2\pi e(\delta_1 + 4\delta_2)}}{\log q}}$$

- ▶ For BLISS-I and BLISS-II, $\ell \approx 1.09 \cdot m$

Attack details (II)

- ▶ Subset $I \subset \{0, \dots, n-1\}$ of cardinal ℓ $\varphi_I: \mathbb{Z}^n \rightarrow \mathbb{Z}^I$ projection
- ▶ $\varphi_I(\mathbf{v})$ close to the lattice generated by $\varphi_I(\mathbf{w}_i)$ and $q\mathbb{Z}^I$
If ℓ large enough, difference should be $\varphi_I(\mathbf{s}_1)$.
- ▶ CVP using Babai nearest plane algorithm. Condition on ℓ to recover $\varphi_I(\mathbf{s}_1)$:

$$\ell + 1 \gtrsim \frac{m + 2 + \frac{\log \sqrt{\delta_1 + 4\delta_2}}{\log q}}{1 - \frac{\log \sqrt{2\pi e(\delta_1 + 4\delta_2)}}{\log q}}$$

- ▶ For BLISS-I and BLISS-II, $\ell \approx 1.09 \cdot m$

Attack details (II)

- ▶ Subset $I \subset \{0, \dots, n-1\}$ of cardinal ℓ $\varphi_I: \mathbb{Z}^n \rightarrow \mathbb{Z}^I$ projection
- ▶ $\varphi_I(\mathbf{v})$ close to the lattice generated by $\varphi_I(\mathbf{w}_i)$ and $q\mathbb{Z}^I$
If ℓ large enough, difference should be $\varphi_I(\mathbf{s}_1)$.
- ▶ CVP using Babai nearest plane algorithm. Condition on ℓ to recover $\varphi_I(\mathbf{s}_1)$:

$$\ell + 1 \gtrsim \frac{m + 2 + \frac{\log \sqrt{\delta_1 + 4\delta_2}}{\log q}}{1 - \frac{\log \sqrt{2\pi e(\delta_1 + 4\delta_2)}}{\log q}}$$

- ▶ For BLISS-I and BLISS-II, $\ell \approx 1.09 \cdot m$

Attack details (III)

- ▶ In practice: Works fine with LLL for $m \lesssim 60$ and with BKZ with $m \lesssim 100$
- ▶ Apply the attack for several choices of l to recover all of s_1 , and subsequently s_2 : full key recovery with one faulty signature!

Fault iteration $m =$	2	5	10	20	40	60	80	100
Theoretical min dim l_{\min}	3	6	11	22	44	66	88	110
Dim l (experimental)	3	6	12	24	50	80	110	140
Reduction algorithm	LLL	LLL	LLL	LLL	BKZ-20	BKZ-25	BKZ-25	BKZ-25
Success probab. (%)	100	99	100	100	100	100	100	—
Time recovery l coeffs. (s)	0.002	0.005	0.022	0.23	7.3	119	941	10500
Time full key recovery	0.5 s	0.5 s	1 s	5 s	80 s	14 min	80 min	12 h

Attack details (III)

- ▶ In practice: Works fine with LLL for $m \lesssim 60$ and with BKZ with $m \lesssim 100$
- ▶ Apply the attack for several choices of l to recover all of s_1 , and subsequently s_2 : full key recovery with one faulty signature!

Fault iteration $m =$	2	5	10	20	40	60	80	100
Theoretical min dim ℓ_{\min}	3	6	11	22	44	66	88	110
Dim ℓ (experimental)	3	6	12	24	50	80	110	140
Reduction algorithm	LLL	LLL	LLL	LLL	BKZ-20	BKZ-25	BKZ-25	BKZ-25
Success proba. (%)	100	99	100	100	100	100	100	—
Time recovery ℓ coeffs. (s)	0.002	0.005	0.022	0.23	7.3	119	941	10500
Time full key recovery	0.5 s	0.5 s	1 s	5 s	80 s	14 min	80 min	12 h

GPV-Based scheme

- ▶ Variant of Ducas-Lyubashevsky-Prest based on GPV-style lattice trapdoors.
- ▶ Defined once again over $\mathcal{R} = \mathbb{Z}[\mathbf{x}]/(\mathbf{x}^n + 1)$
- ▶ Secret key:

$$\mathbf{B} \leftarrow \begin{pmatrix} \mathbf{M}_g & -\mathbf{M}_f \\ \mathbf{M}_G & -\mathbf{M}_F \end{pmatrix} \in \mathbb{Z}^{2n \times 2n}$$

$$\text{for } \mathbf{f} \leftarrow D_{\sigma_0}^n, \mathbf{g} \leftarrow D_{\sigma_0}^n$$

$$f \cdot G - g \cdot F = q$$

Sign and Verify

```
1: function SIGN( $\mu$ ,  $sk = \mathbf{B}$ )
2:    $\mathbf{c} \leftarrow H(\mu) \in \mathbb{Z}_q^n$ 
3:    $(\mathbf{y}, \mathbf{z}) \leftarrow (\mathbf{c}, \mathbf{0}) - \text{GAUSSIAN SAMPLER}(\mathbf{B}, \sigma, (\mathbf{c}, \mathbf{0})) \triangleright \mathbf{y}, \mathbf{z}$   
   are short and satisfy  $\mathbf{y} + \mathbf{z} \cdot \mathbf{h} = \mathbf{c} \pmod q$ 
4:   return  $\mathbf{z}$ 
5: end function
```

```
1: function VERIFY( $\mu$ ,  $pk = \mathbf{h}, \mathbf{z}$ )
2:   accept iff  $\|\mathbf{z}\|_2 + \|H(\mu) - \mathbf{z} \cdot \mathbf{h}\|_2 \leq \sigma\sqrt{2n}$ 
3: end function
```

Sign and Verify

```
1: function SIGN( $\mu$ ,  $sk = \mathbf{B}$ )
2:    $\mathbf{c} \leftarrow H(\mu) \in \mathbb{Z}_q^n$ 
3:    $(\mathbf{y}, \mathbf{z}) \leftarrow (\mathbf{c}, \mathbf{0}) - \text{GAUSSIAN SAMPLER}(\mathbf{B}, \sigma, (\mathbf{c}, \mathbf{0})) \triangleright \mathbf{y}, \mathbf{z}$   
   are short and satisfy  $\mathbf{y} + \mathbf{z} \cdot \mathbf{h} = \mathbf{c} \pmod q$ 
4:   return  $\mathbf{z}$ 
5: end function
```

```
1: function VERIFY( $\mu$ ,  $pk = \mathbf{h}, \mathbf{z}$ )
2:   accept iff  $\|\mathbf{z}\|_2 + \|H(\mu) - \mathbf{z} \cdot \mathbf{h}\|_2 \leq \sigma\sqrt{2n}$ 
3: end function
```

Gaussian Sampling

- 1: **function** GAUSSIAN_SAMPLER($\mathbf{B}, \sigma, \mathbf{c}$) \triangleright \mathbf{b}_i (resp. $\tilde{\mathbf{b}}_i$) are the rows of \mathbf{B} (resp. of its Gram–Schmidt matrix $\tilde{\mathbf{B}}$)
- 2: $\mathbf{v} \leftarrow \mathbf{0}$
- 3: **for** $i = 2n$ **down to** 1 **do**
- 4: $c' \leftarrow \langle \mathbf{c}, \tilde{\mathbf{b}}_i \rangle / \|\tilde{\mathbf{b}}_i\|_2^2$
- 5: $\sigma' \leftarrow \sigma / \|\tilde{\mathbf{b}}_i\|_2$
- 6: $r \leftarrow D_{\mathbb{Z}, \sigma', c'}$
- 7: $\mathbf{c} \leftarrow \mathbf{c} - r\mathbf{b}_i$ and $\mathbf{v} \leftarrow \mathbf{v} + r\mathbf{b}_i$
- 8: **end for**
- 9: **return** \mathbf{v} \triangleright \mathbf{v} sampled according to the lattice Gaussian distribution $D_{\Lambda, \sigma, \mathbf{c}}$
- 10: **end function**

Gaussian Sampling

- 1: **function** GAUSSIAN_SAMPLER($\mathbf{B}, \sigma, \mathbf{c}$) $\triangleright \mathbf{b}_i$ (resp. $\tilde{\mathbf{b}}_i$) are the rows of \mathbf{B} (resp. of its Gram–Schmidt matrix $\tilde{\mathbf{B}}$)
- 2: $\mathbf{v} \leftarrow \mathbf{0}$
- 3: **for** $i = 2n$ **down to** 1 **do**
- 4: $c' \leftarrow \langle \mathbf{c}, \tilde{\mathbf{b}}_i \rangle / \|\tilde{\mathbf{b}}_i\|_2^2$
- 5: $\sigma' \leftarrow \sigma / \|\tilde{\mathbf{b}}_i\|_2$
- 6: $r \leftarrow D_{\mathbb{Z}, \sigma', c'}$
- 7: $\mathbf{c} \leftarrow \mathbf{c} - r\mathbf{b}_i$ and $\mathbf{v} \leftarrow \mathbf{v} + r\mathbf{b}_i$
- 8: **end for**
- 9: **return** \mathbf{v} $\triangleright \mathbf{v}$ sampled according to the lattice Gaussian distribution $D_{\Lambda, \sigma, \mathbf{c}}$
- 10: **end function**

Attacking the Gaussian sampler

- ▶ Correctly generated signature: element of the form

$$\mathbf{z} = \mathbf{R} \cdot \mathbf{f} + \mathbf{r} \cdot \mathbf{F} \in \mathbb{Z}[\mathbf{x}]/(\mathbf{x}^n + 1)$$

- ▶ Faults introduced after m iterations of the generation of r, R :

$$\mathbf{z} = r_0 \mathbf{x}^{n-1} \mathbf{F} + r_1 \mathbf{x}^{n-2} \mathbf{F} + \dots + r_{m-1} \mathbf{x}^{n-m} \mathbf{F}.$$

- ▶ Belongs to lattice :

$$L = \text{Span}(\mathbf{x}^{n-i} \mathbf{F})$$

for $1 \leq i \leq m$.

Attacking the Gaussian sampler

- ▶ Correctly generated signature: element of the form

$$\mathbf{z} = \mathbf{R} \cdot \mathbf{f} + \mathbf{r} \cdot \mathbf{F} \in \mathbb{Z}[\mathbf{x}] / (\mathbf{x}^n + 1)$$

- ▶ Faults introduced after m iterations of the generation of r, R :

$$\mathbf{z} = r_0 \mathbf{x}^{n-1} \mathbf{F} + r_1 \mathbf{x}^{n-2} \mathbf{F} + \dots + r_{m-1} \mathbf{x}^{n-m} \mathbf{F}.$$

- ▶ Belongs to lattice :

$$L = \text{Span}(\mathbf{x}^{n-i} \mathbf{F})$$

for $1 \leq i \leq m$.

Attacking the Gaussian sampler

- ▶ Correctly generated signature: element of the form

$$\mathbf{z} = \mathbf{R} \cdot \mathbf{f} + \mathbf{r} \cdot \mathbf{F} \in \mathbb{Z}[\mathbf{x}]/(\mathbf{x}^n + 1)$$

- ▶ Faults introduced after m iterations of the generation of r, R :

$$\mathbf{z} = r_0 \mathbf{x}^{n-1} \mathbf{F} + r_1 \mathbf{x}^{n-2} \mathbf{F} + \dots + r_{m-1} \mathbf{x}^{n-m} \mathbf{F}.$$

- ▶ Belongs to lattice :

$$L = \text{Span}(\mathbf{x}^{n-i} \mathbf{F})$$

for $1 \leq i \leq m$.

Multiple faulted signatures?

- ▶ $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(\ell)}$ faulty signatures.
- ▶ With probability $\geq \prod_{k=l-m+1}^{+\infty} \frac{1}{\zeta(k)}$ generates L . [*Maze, Rosenthal, Wagner*]
- ▶ SVP of L should be one of the $\mathbf{x}^{n-i}\mathbf{F}$ for $1 \leq i \leq m$.
 - \implies Full recovery of a basis $(\zeta\mathbf{F}, \zeta\mathbf{g}, \zeta\mathbf{F}, \zeta\mathbf{G})$ for a $\zeta = \pm \mathbf{x}^n$.

Multiple faulted signatures?

- ▶ $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(\ell)}$ faulty signatures.
- ▶ With probability $\geq \prod_{k=l-m+1}^{+\infty} \frac{1}{\zeta(k)}$ generates L . [*Maze, Rosenthal, Wagner*]
- ▶ SVP of L should be one of the $\mathbf{x}^{n-i}\mathbf{F}$ for $1 \leq i \leq m$.
 \implies Full recovery of a basis $(\zeta f, \zeta g, \zeta F, \zeta G)$ for a $\zeta = \pm \mathbf{x}^n$.

Multiple faulted signatures?

- ▶ $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(\ell)}$ faulty signatures.
- ▶ With probability $\geq \prod_{k=l-m+1}^{+\infty} \frac{1}{\zeta(k)}$ generates L . [*Maze, Rosenthal, Wagner*]
- ▶ SVP of L should be one of the $\mathbf{x}^{n-i}\mathbf{F}$ for $1 \leq i \leq m$.

\implies Full recovery of a basis $(\zeta f, \zeta g, \zeta F, \zeta G)$ for a $\zeta = \pm x^a$.

Multiple faulted signatures?

- ▶ $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(\ell)}$ faulty signatures.
 - ▶ With probability $\geq \prod_{k=l-m+1}^{+\infty} \frac{1}{\zeta(k)}$ generates L . [*Maze, Rosenthal, Wagner*]
 - ▶ SVP of L should be one of the $\mathbf{x}^{n-i}\mathbf{F}$ for $1 \leq i \leq m$.
- \implies Full recovery of a basis $(\zeta\mathbf{f}, \zeta\mathbf{g}, \zeta\mathbf{F}, \zeta\mathbf{G})$ for a $\zeta = \pm\mathbf{x}^\alpha$.

In practice

Fault after iteration number $m =$	2	5	10	20	40	60	80	100
Lattice reduction algorithm	LLL	LLL	LLL	LLL	LLL	LLL	BKZ-20	BKZ-20
Success probability for $\ell = m + 1$ (%)	75	77	90	93	94	94	95	95
Avg. CPU time for $\ell = m + 1$ (s)	0.001	0.003	0.016	0.19	2.1	8.1	21.7	104
Success probability for $\ell = m + 2$ (%)	89	95	100	100	99	99	100	100
Avg. CPU time for $\ell = m + 2$ (s)	0.001	0.003	0.017	0.19	2.1	8.2	21.6	146

Conclusion and countermeasures

- ▶ Important to investigate implementation attacks on lattice schemes
- ▶ **Faults against Fiat-Shamir and Hash-And-Sign** signatures
 - ▶ Among **first fault attacks** against **non-broken** lattice signatures
 - ▶ Both based on early loop abort
 - ▶ One of them recovers **the full key with a single faulty sig.**
 - ▶ Other one: multiple faulty sig., but still **on fault per sig.**

Conclusion and countermeasures

- ▶ Check that the loop ran completely (**two** loop counters)
- ▶ For \mathbf{y}_1 : check that the result has $> (1 - \varepsilon) \cdot n$ non zero coeffs.
- ▶ Alternatively: randomize the order of generation of the coefficients (still a bit risky)

Thank you for your attention!

