AN EFFICIENT LATTICE-BASED SIGNATURE SCHEME WITH PROVABLY SECURE INSTANTIATION

TECHNISCHE UNIVERSITÄT DARMSTADT

CROSSING

AfricaCrypt 2016
International Conference on Cryptology
Fez, Morocco
04/13/2016

Sedat Akleylek
Nina Bindel
Johannes Buchmann
Juliane Krämer
Giorgia A. Marson

OUTLINE

- Security Reduction and Provably Secure Instantiation
- Description of the Signature Scheme
- Parameter Selection
- Comparison with State-of-the-Art
- Conclusion

LATTICE-BASED SIGNATURES

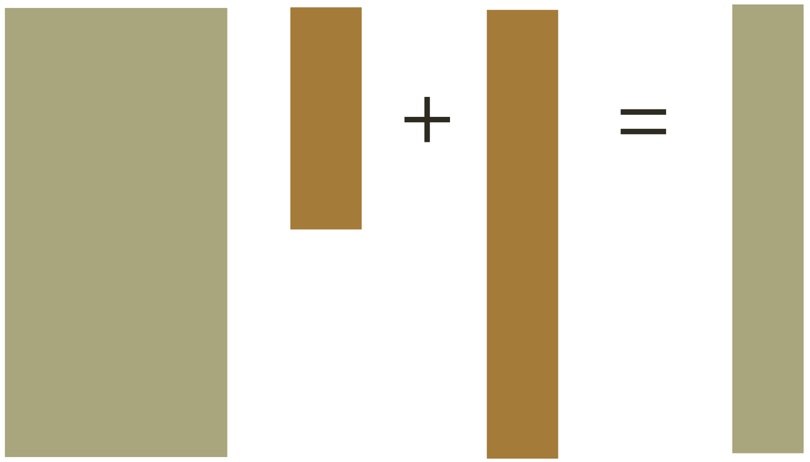
Signature Scheme	Bit Security	Sign. Size [Byte]	Sign Cycles	Verify Cycles	Comp. Assumption
GLP* **Güneysu, Lyubashevsky, Pöppelmann**	75-80	1 186	570 000	46 000	DCK
BLISS* **Ducas, Durmus, Lepoint, Lyubashevsky**	128	1 559	351 000	102 000	R-SIS, NTRU

* Sizes of uncompressed elements from the implementation given

LEARNING WITH ERRORS PROBLEM (LWE)

LEARNING WITH ERRORS PROBLEM (LWE)

LWE



$$A \cdot s + e = b \mod q$$

RING-LEARNING WITH ERRORS PROBLEM (R-LWE)

R-LWE

$$a \cdot s + e = b \mod q$$

$$a \xleftarrow{\$} \mathbb{Z}_q[x]/(x^n + 1)$$

$$s, e \leftarrow D_\sigma$$

RING-LEARNING WITH ERRORS PROBLEM (R-LWE)

R-LWE

$$a \cdot s + e = b \bmod q$$

$$a \xleftarrow{\$} \mathbb{Z}_q[x]/(x^n + 1)$$

$$s, e \longleftarrow D_\sigma$$

DCK

$$a \cdot s + e = b \bmod q$$

$$a \xleftarrow{\$} \mathbb{Z}_q[x]/(x^n + 1)$$

RING-LEARNING WITH ERRORS PROBLEM (R-LWE)

R-LWE

$a \cdot s + e = b \bmod q$

$a \xleftarrow{\$} \mathbb{Z}_q[x]/(x^n + 1)$

$s, e \leftarrow D_\sigma$

DCK

$a \cdot s + e = b \bmod q$

$a \xleftarrow{\$} \mathbb{Z}_q[x]/(x^n + 1)$

$s_i, e_i \leftarrow [-1,0,1]$

LATTICE-BASED SIGNATURES

Signature Scheme	Bit Security	Sign. Size [Byte]	Sign Cycles	Verify Cycles	Comp. Assumption
GLP* **Güneysu, Lyubashevsky, Pöppelmann**	75-80	1 186	570 000	46 000	DCK
BLISS* **Ducas, Durmus, Lepoint, Lyubashevsky**	128	1 559	351 000	102 000	R-SIS, NTRU

* Sizes of uncompressed elements from the implementation given

LATTICE-BASED SIGNATURES

- Good performance
- Provable Secure

LATTICE-BASED SIGNATURES

- Good performance
- Provable Secure
- Provably Secure Instantiation

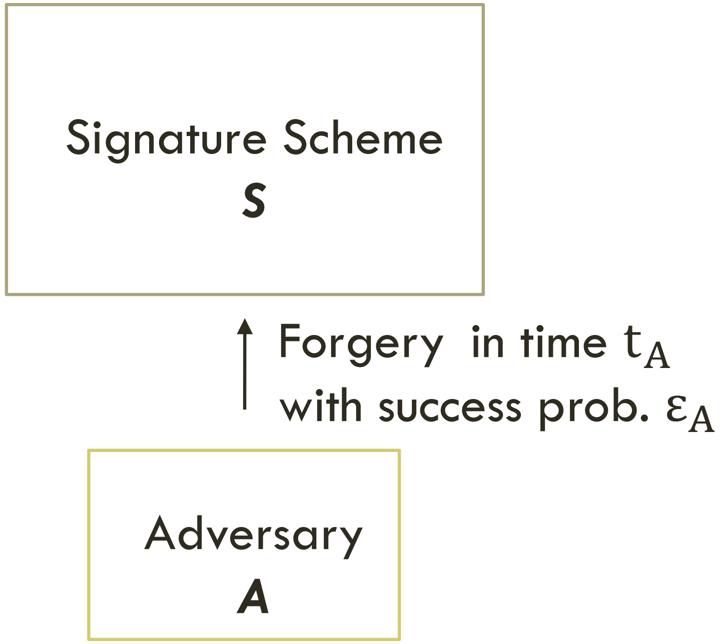
LATTICE-BASED SIGNATURES

- Good performance
- Provable Secure
- Provably Secure Instantiation

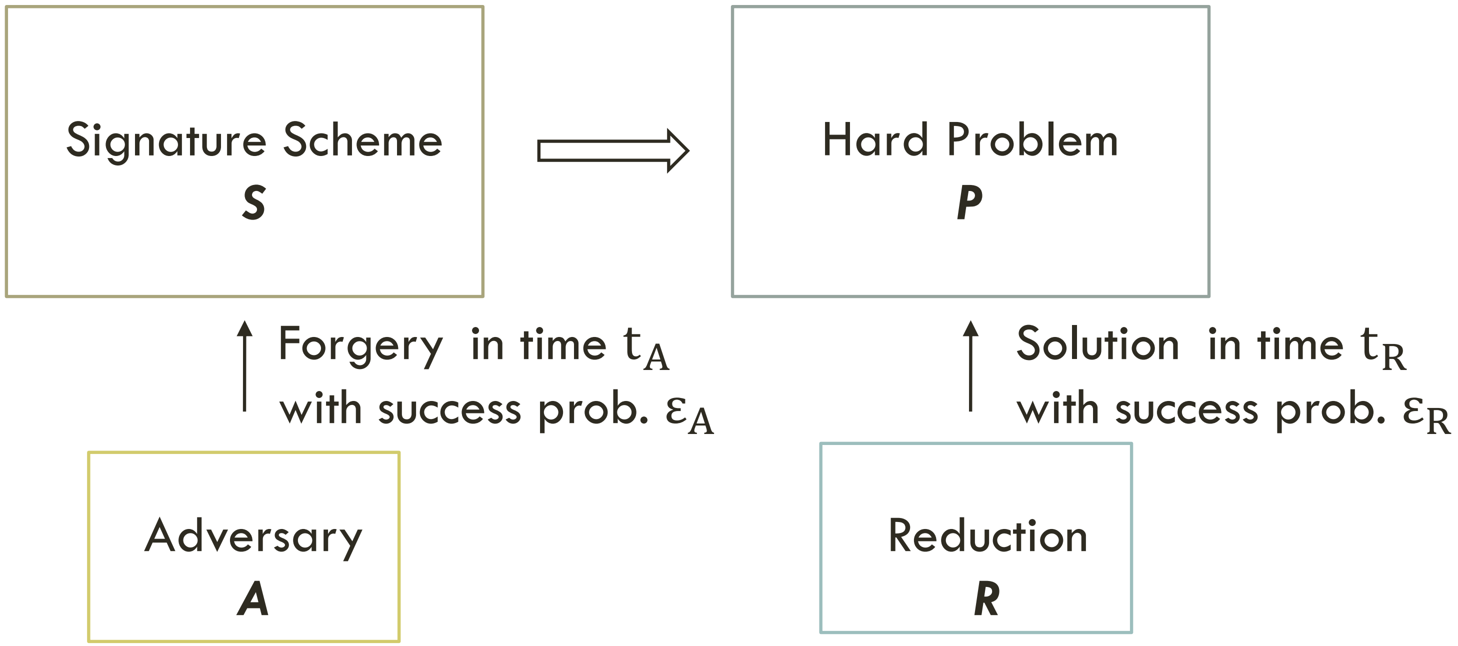
Signature Scheme	Bit Security	Sign. Size [Byte]	Sign Cycles	Verify Cycles	Comp. Assumption	Provably Secure Instantiation
GLP* **Güneysu, Lyubashevsky, Pöppelmann**	75-80	1 186	570 000	46 000	DCK	no
BLISS* **Ducas, Durmus, Lepoint, Lyubashevsky**	128	1 559	351 000	102 000	R-SIS, NTRU	no

* Sizes of uncompressed elements from the implementation given

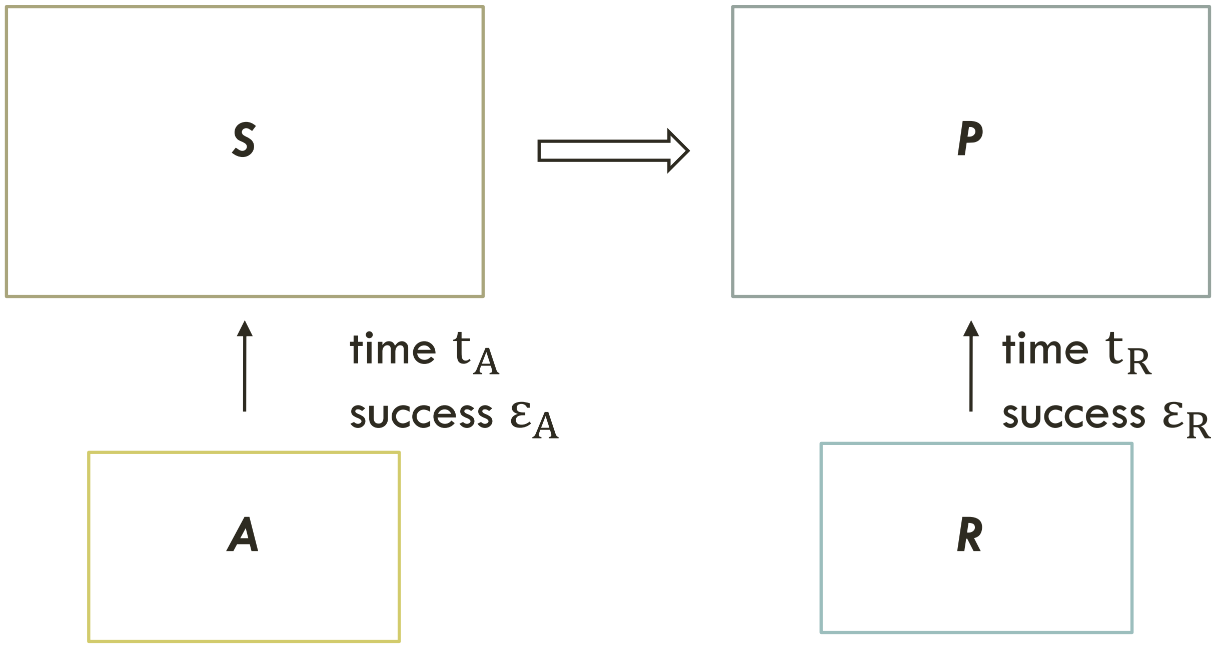
SECURITY REDUCTION



SECURITY REDUCTION



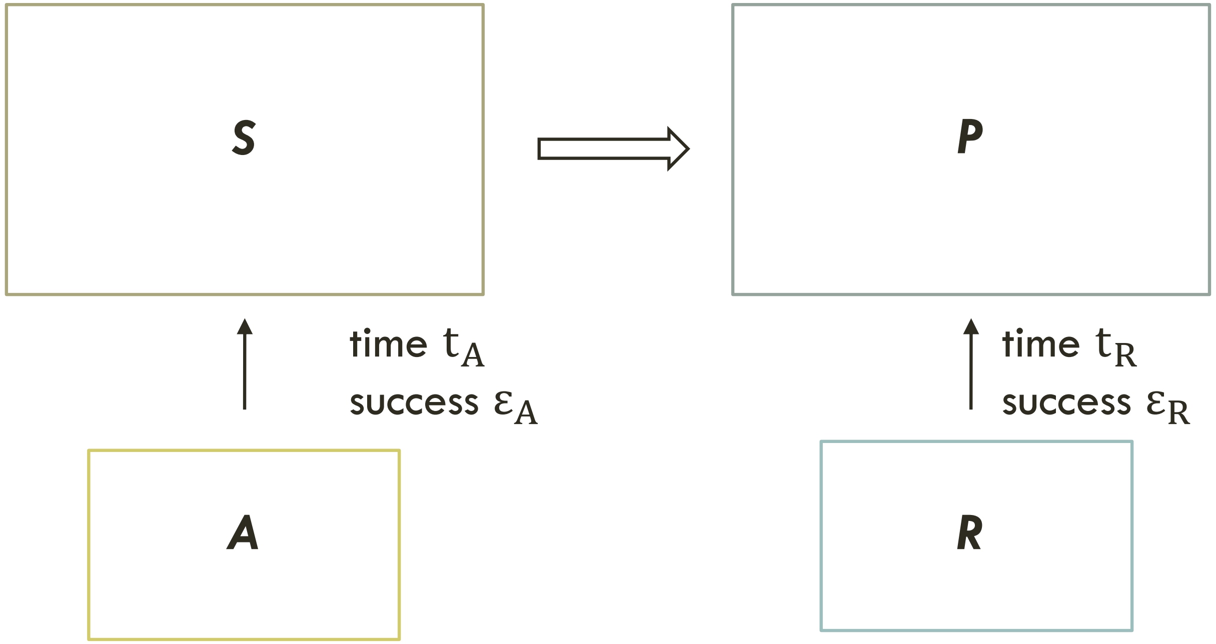
HARDNESS AND SECURITY



Bit-security: $\frac{t_A}{\varepsilon_A}$

Bit-hardness: $\frac{t_R}{\varepsilon_R}$

PROVABLY SECURE INSTANTIATION



Example:

- $t_R \approx t_A$
- $\varepsilon_R \approx \varepsilon_A$

Bit-security: $\frac{t_A}{\varepsilon_A}$

Bit-hardness: $\frac{t_R}{\varepsilon_R}$

HARDNESS AND SECURITY - EXAMPLE

Example:

- $t_R \approx t_A$
- $\varepsilon_R \approx \varepsilon_A$

$\longrightarrow$ ***P*** bit-hardness: 100 bit
S bit-security: $\approx$ 100 bit

HARDNESS AND SECURITY - EXAMPLE

Example:

- $t_R \approx t_A$
- $\varepsilon_R \approx \varepsilon_A$

$\longrightarrow$ ***P*** bit-hardness: 100 bit
S bit-security: $\approx$ 100 bit

- $t_R \approx t_A$
- $\varepsilon_R \approx \varepsilon_A^2$

HARDNESS AND SECURITY - EXAMPLE

Example:

- $t_R \approx t_A$
- $\varepsilon_R \approx \varepsilon_A$

$\longrightarrow$ ***P*** bit-hardness: 100 bit
S bit-security: $\approx$ 100 bit

- $t_R \approx t_A$
- $\varepsilon_R \approx \varepsilon_A^2$

$\longrightarrow$ ***P*** bit-hardness: 100 bit
S bit-security: ? $\geq$ 50 bit

PROVABLY SECURE INSTANTIATION

To choose instantiation of ***P*** s. th. security of ***S*** gives desired security level, e.g., 100 bit

PROVABLY SECURE INSTANTIATION

To choose instantiation of ***P*** s. th. security of ***S*** gives desired security level, e.g., 100 bit

Example:

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A$

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A^2$

PROVABLY SECURE INSTANTIATION

To choose instantiation of ***P*** s. th. security of ***S*** gives desired security level, e.g., 100 bit

Example:

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A$

bit-security of ***S*** = bit-hardness of ***P*** = 100 bit

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A^2$

PROVABLY SECURE INSTANTIATION

To choose instantiation of ***P*** s. th. security of ***S*** gives desired security level, e.g., 100 bit

Example:

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A$

bit-security of ***S*** = bit-hardness of ***P*** = 100 bit

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A^2$

bit-security of ***S*** ≥ ½ bit-hardness of ***P***

PROVABLY SECURE INSTANTIATION

To choose instantiation of ***P*** s. th. security of ***S*** gives desired security level, e.g., 100 bit

Example:

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A$

bit-security of ***S*** = bit-hardness of ***P*** = 100 bit

- $t_R = t_A$
- $\varepsilon_R = \varepsilon_A^2$

bit-security of ***S*** ≥ ½ bit-hardness of ***P***

- choose bit-hardness of P = 200 bit
- to get bit-security of ***S*** ≥ 100 bit

LATTICE-BASED SIGNATURES

- Good performance
- Provable Secure
- Provably Secure Instantiation

Signature Scheme	Bit Security	Sign. Size [Byte]	Sign Cycles	Verify Cycles	Comp. Assumption	Provably Secure Instantiation
GLP* **Güneysu, Lyubashevsky, Pöppelmann**	75-80	1 186	570 000	46 000	DCK	no
BLISS* **Ducas, Durmus, Lepoint, Lyubashevsky**	128	1 559	351 000	102 000	R-SIS, NTRU	no

* Sizes of uncompressed elements from the implementation given

RING-TESLA

RING-TESLA

- Construction by Bai and Galbraith
  - Standard lattices
  - LWE, SIS

RING-TESLA

- Construction by Bai and Galbraith
  - Standard lattices
  - LWE, SIS
- Improvements by Dagdelen, El Bansarkani, Göpfert, Güneysu, Oder, Pöppelmann, Sánchez, Schwabe
  - Parameters for high-speed implementation

RING-TESLA

- Construction by Bai and Galbraith
  - Standard lattices
  - LWE, SIS
- Improvements by Dagdelen, El Bansarkani, Göpfert, Güneysu, Oder, Pöppelmann, Sánchez, Schwabe
  - Parameters for high-speed implementation
- TESLA by Alkim, Bindel, Buchmann, Dagdelen, Schwabe
  - Standard lattices
  - Tight reduction from LWE

  ⟶ Provably Secure Instantiation

RING-TESLA

- Ideal lattices
- R-LWE
- Tight security reduction
  → Provably Secure Instantiation

DESCRIPTION OF RING-TESLA

Sign

Verify

DESCRIPTION OF RING-TESLA

$\mathrm{pk} = (\mathrm{a}_1, \mathrm{b}_1, \mathrm{a}_2, \mathrm{b}_2)$
$\mathrm{sk} = (\mathrm{s}, \mathrm{e}_1, \mathrm{e}_2)$

Sign

Verify

DESCRIPTION OF RING-TESLA

$pk = (a_1, b_1, a_2, b_2)$
$sk = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$

Verify

DESCRIPTION OF RING-TESLA

$\mathrm{pk} = (a_1, b_1, a_2, b_2)$
$\mathrm{sk} = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$
2. $c \leftarrow H(\lfloor a_1 y \rceil, \lfloor a_2 y \rceil, \mu)$
3. $z = y + sc$

Verify

DESCRIPTION OF RING-TESLA

$\mathrm{pk} = (a_1, b_1, a_2, b_2)$
$\mathrm{sk} = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$
2. $c \leftarrow H(\lfloor a_1 y \rceil, \lfloor a_2 y \rceil, \mu)$
3. $z = y + sc$
4. if $\|a_i y - e_i c\|_2$ small $\wedge$ $\|z\|_\infty$ small:
   return (z, c)
5. else: restart

Verify

DESCRIPTION OF RING-TESLA

$\mathrm{pk} = (a_1, b_1, a_2, b_2)$
$\mathrm{sk} = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$
2. $c \leftarrow H(\lfloor a_1 y \rceil, \lfloor a_2 y \rceil, \mu)$
3. $z = y + sc$
4. if $\|a_i y - e_i c\|_2$ small $\wedge$ $\|z\|_\infty$ small:

Correctness Security

Verify

DESCRIPTION OF RING-TESLA

$pk = (a_1, b_1, a_2, b_2)$
$sk = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$
2. $c \leftarrow H(\lfloor a_1 y \rceil, \lfloor a_2 y \rceil, \mu)$
3. $z = y + sc$
4. if $\|a_i y - e_i c\|_2$ small $\wedge$ $\|z\|_\infty$ small :
   return (z, c)
5. else: restart

Verify

Input: pk, μ, σ

Output: $\{0,1\}$

DESCRIPTION OF RING-TESLA

$\mathrm{pk} = (a_1, b_1, a_2, b_2)$
$\mathrm{sk} = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$
2. $c \leftarrow H(\lfloor a_1 y \rceil, \lfloor a_2 y \rceil, \mu)$
3. $z = y + sc$
4. if $\|a_i y - e_i c\|_2$ small $\wedge \|z\|_\infty$ small:
   return (z, c)
5. else: restart

Verify

Input: pk, μ, σ

Output: $\{0,1\}$

1. if $c = H(\lfloor a_1 z - b_1 c \rceil, \lfloor a_2 z - b_2 c \rceil, \mu)$
   $\wedge \|z\|_\infty$ small:
   return 1
2. return 0

DESCRIPTION OF RING-TESLA

$pk = (a_1, b_1, a_2, b_2)$
$sk = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$
2. $c \leftarrow H(\lfloor a_1 y \rceil, \lfloor a_2 y \rceil, \mu)$
3. $z = y + sc$
4. if $\|a_i y - e_i c\|_2$ small $\wedge$ $\|z\|_\infty$ small :
   return (z, c)
5. else: restart

Verify

Input: pk, μ, σ

Output: $\{0,1\}$

1. if $c = H(\lfloor a_1 z - b_1 c \rceil, \lfloor a_2 z - b_2 c \rceil, \mu)$
   $\wedge$ $\|z\|_\infty$small:
   return 1
2. return 0

UNIFORM VS. GAUSSIAN SAMPLING

Uniform Sampling

- timing-constant implementation
- large signature size

Gaussian Sampling

- no (efficient) timing-constant implementation
- small signature size

PARAMETER SELECTION (GENERAL)

General case:

1. Choose security level

PARAMETER SELECTION (GENERAL)

General case:

1. Choose security level
2. Select problem instance with assumption Hardness = Security

$$\frac{t_A}{\varepsilon_A} \sim \frac{t_R}{\varepsilon_R}$$

PARAMETER SELECTION (GENERAL)

General case:

1. Choose security level
2. Select problem instance with assumption Hardness = Security
3. Select system parameters

$$\frac{t_A}{\varepsilon_A} \sim \frac{t_R}{\varepsilon_R}$$

PARAMETER SELECTION (OUR CASE)

Our case:

1. Security level: 128 bit

PARAMETER SELECTION (OUR CASE)

Our case:

1. Security level: 128 bit
2. Tight security reduction

→ Hardness = Security + 2 bit

→ Choose 130-bit ring-LWE instance: σ, q, n

3. Compute system parameters

$$a \cdot s + e = b \bmod q$$

$$\frac{t_A}{\varepsilon_A} \sim \frac{t_R}{\varepsilon_R}$$

SYSTEM PARAMETERS RING-TESLA

$pk = (a_1, b_1, a_2, b_2)$
$sk = (s, e_1, e_2)$

Sign

Input: sk, μ

Output: $\sigma = (z, c)$

1. $y \leftarrow R_B$
2. $c \leftarrow H(\lfloor a_1 y \rceil, \lfloor a_2 y \rceil, \mu)$
3. $z = y + sc$
4. if $\|a_i y - e_i c\|_2$ small $\wedge$ $\|z\|_\infty$ small :
   return (z, c)
5. else: restart

Verify

Input: pk, μ, σ

Output: $\{0,1\}$

1. if $c = H(\lfloor a_1 z - b_1 c \rceil, \lfloor a_2 z - b_2 c \rceil, \mu)$
   $\wedge$ $\|z\|_\infty$ small:
   return 1
2. return 0

COMPARISON (SPACE)

Signature Scheme	Bit Security	Sign. Size [Byte]	pk Size [byte]	sk Size [byte]	Provably Sec. Instantiation
GLP* **Güneysu, Lyubashevsky, Pöppelmann**	75-80	1 186	1 536	256	no
ring-TESLA* **(this work)**	80	1 728	3 072	1 728	yes
BLISS* **Ducas, Durmus, Lepoint, Lyubashevsky**	128	1 559	7 168	2 048	no
ring-TESLA* **(this work)**	128	1 568	3 328	1 920	yes

* Sizes of uncompressed elements from the implementation given

COMPARISON (RUNTIME)

Signature Scheme	Bit Security	Sign Cycles	Verify Cycles	Provably Sec. Instantiation
GLP **Güneysu, Lyubashevsky, Pöppelmann**	75-80	570 000	46 000	no
ring-TESLA **(this work)**	80	371 000	94 000	yes
BLISS **Ducas, Durmus, Lepoint, Lyubashevsky**	128	351 000	102 000	no
ring-TESLA **(this work)**	128	511 000	168 000	yes

CONCLUSION

CONCLUSION

- ideal-lattice based signature scheme from R-LWE

CONCLUSION

- ideal-lattice based signature scheme from R-LWE
- provably secure instantiations

CONCLUSION

- ideal-lattice based signature scheme from R-LWE
- provably secure instantiations
- sizes and runtimes similar to GLP and BLISS

CONCLUSION

- ideal-lattice based signature scheme from R-LWE
- provably secure instantiations
- sizes and runtimes similar to GLP and BLISS
- no Gaussian sampling during sign algorithm

TECHNISCHE UNIVERSITÄT DARMSTADT

CROSSING

THANKS

Questions or Comments ?