LWE over the Integers and Improved Side-Channel Attacks against BLISS

J. Bootle, C. Delaplace, T. Espitau,
P.-A. Fouque and M. Tibouchi

NTT Secure Platform Laboratories

ASIACRYPT 2018, Brisbane, 2018–12–03

Outline

Outline

Outline

Introduction

- Espitau, Fouque, Gérard and T. (CCS 2017): the rejection sampling step in BLISS leaks secret key info through timing side-channels
- More precisely, leakage of two functions of the secret key
  - exact leakage of a quadratic function of the key
  - noisy leakage of a linear function of the key
- In the CCS paper: exploit the quadratic leakage
  - requires relatively few side-channel traces
  - heavy-weight, expensive algebraic number theory
  - can only attack weak keys ($\approx 7\%$)
- Claim: the linear leakage is not useful
  - noisy linear system of dimension $\geq$ original lattice problem
  - so this should not help
- This talk: actually, it is useful!
  - much faster attack than CCS
  - works against all keys
  - drawback: requires more traces

Introduction

- Espitau, Fouque, Gérard and T. (CCS 2017): the rejection sampling step in BLISS leaks secret key info through timing side-channels
- More precisely, leakage of two functions of the secret key
  - exact leakage of a quadratic function of the key
  - noisy leakage of a linear function of the key
- In the CCS paper: exploit the quadratic leakage
  - requires relatively few side-channel traces
  - heavy-weight, expensive algebraic number theory
  - can only attack weak keys ($\approx 7\%$)
- Claim: the linear leakage is not useful
  - noisy linear system of dimension $\geq$ original lattice problem
  - so this should not help
- This talk: actually, it is useful!
  - much faster attack than CCS
  - works against all keys
  - drawback: requires more traces

Introduction

- Espitau, Fouque, Gérard and T. (CCS 2017): the rejection sampling step in BLISS leaks secret key info through timing side-channels
- More precisely, leakage of two functions of the secret key
  - exact leakage of a quadratic function of the key
  - noisy leakage of a linear function of the key
- In the CCS paper: exploit the quadratic leakage
  - requires relatively few side-channel traces
  - heavy-weight, expensive algebraic number theory
  - can only attack weak keys ($\approx 7\%$)
- Claim: the linear leakage is not useful
  - noisy linear system of dimension $\geq$ original lattice problem
  - so this should not help
- This talk: actually, it is useful!
  - much faster attack than CCS
  - works against all keys
  - drawback: requires more traces

Introduction

- Espitau, Fouque, Gérard and T. (CCS 2017): the rejection sampling step in BLISS leaks secret key info through timing side-channels
- More precisely, leakage of two functions of the secret key
  - exact leakage of a quadratic function of the key
  - noisy leakage of a linear function of the key
- In the CCS paper: exploit the quadratic leakage
  - requires relatively few side-channel traces
  - heavy-weight, expensive algebraic number theory
  - can only attack weak keys ($\approx 7\%$)
- Claim: the linear leakage is not useful
  - noisy linear system of dimension $\geq$ original lattice problem
  - so this should not help
- This talk: actually, it is useful!
  - much faster attack than CCS
  - works against all keys
  - drawback: requires more traces

Introduction

- Espitau, Fouque, Gérard and T. (CCS 2017): the rejection sampling step in BLISS leaks secret key info through timing side-channels
- More precisely, leakage of two functions of the secret key
  - exact leakage of a quadratic function of the key
  - noisy leakage of a linear function of the key
- In the CCS paper: exploit the quadratic leakage
  - requires relatively few side-channel traces
  - heavy-weight, expensive algebraic number theory
  - can only attack weak keys ($\approx 7\%$)
- Claim: the linear leakage is not useful
  - noisy linear system of dimension $\geq$ original lattice problem
  - so this should not help
- This talk: actually, it is useful!
  - much faster attack than CCS
  - works against all keys
  - drawback: requires more traces

BLISS: the basics

- Introduced by Ducas, Durmus, Lepoint and Lyubashevsky at CRYPTO'13
- Not a NIST candidate, but probably still the fastest lattice-based signature
- Implementations on various platforms: desktop computers, microcontrollers/smartcards, FPGAs
- Deployed in the VPN library strongSwan

BLISS: the basics

- Introduced by Ducas, Durmus, Lepoint and Lyubashevsky at CRYPTO'13
- Not a NIST candidate, but probably still the fastest lattice-based signature
- Implementations on various platforms: desktop computers, microcontrollers/smartcards, FPGAs
- Deployed in the VPN library strongSwan

BLISS: the basics

- Introduced by Ducas, Durmus, Lepoint and Lyubashevsky at CRYPTO'13
- Not a NIST candidate, but probably still the fastest lattice-based signature
- Implementations on various platforms: desktop computers, microcontrollers/smartcards, FPGAs
- Deployed in the VPN library strongSwan

BLISS: the basics

- Introduced by Ducas, Durmus, Lepoint and Lyubashevsky at CRYPTO'13
- Not a NIST candidate, but probably still the fastest lattice-based signature
- Implementations on various platforms: desktop computers, microcontrollers/smartcards, FPGAs
- Deployed in the VPN library strongSwan

BLISS: signing and verification keys

- Works in the cyclotomic ring $R = \mathbb{Z}[\mathbf{x}]/(x^n + 1)$, $n = 512$
- Computations modulo the prime $q = 12289$
- Secret key: random sparse $\mathbf{s}_1, \mathbf{s}_2 \in R$ with coefficients in $\{-1, 0, 1\}$
- Verification key: $\mathbf{a} = -\mathbf{s}_2/\mathbf{s}_1 \bmod q$
  - restart if $\mathbf{s}_1$ not invertible

BLISS: signing and verification keys

- Works in the cyclotomic ring $R = \mathbb{Z}[\mathbf{x}]/(x^n + 1)$, $n = 512$
- Computations modulo the prime $q = 12289$
- Secret key: random sparse $\mathbf{s}_1, \mathbf{s}_2 \in R$ with coefficients in $\{-1, 0, 1\}$
- Verification key: $\mathbf{a} = -\mathbf{s}_2/\mathbf{s}_1 \bmod q$
  - restart if $\mathbf{s}_1$ not invertible

BLISS: signing and verification keys

- Works in the cyclotomic ring $R = \mathbb{Z}[\mathbf{x}]/(x^n + 1)$, $n = 512$
- Computations modulo the prime $q = 12289$
- Secret key: random sparse $\mathbf{s}_1, \mathbf{s}_2 \in R$ with coefficients in $\{-1, 0, 1\}$
- Verification key: $\mathbf{a} = -\mathbf{s}_2/\mathbf{s}_1 \bmod q$
  - restart if $\mathbf{s}_1$ not invertible

BLISS: signing and verification keys

- Works in the cyclotomic ring $R = \mathbb{Z}[\mathbf{x}]/(x^n + 1)$, $n = 512$
- Computations modulo the prime $q = 12289$
- Secret key: random sparse $\mathbf{s}_1, \mathbf{s}_2 \in R$ with coefficients in $\{-1, 0, 1\}$
- Verification key: $\mathbf{a} = -\mathbf{s}_2/\mathbf{s}_1 \bmod q$
  - restart if $\mathbf{s}_1$ not invertible

BLISS: signature (simplified)

1: **function** $\textsc{Sign}(\mu, pk = \mathbf{a},\ sk = \mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2))$
2: $\quad \mathbf{y}_1, \mathbf{y}_2 \leftarrow D^n_{\mathbb{Z},\sigma}$ ▷ Gaussian sampling
3: $\quad \mathbf{c} \leftarrow H(\mathbf{a} \cdot \mathbf{y}_1 + \mathbf{y}_2, \mu)$ ▷ special hashing
4: $\quad$ choose a random bit b
5: $\quad \mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$
6: $\quad \mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
7: $\quad$ **continue** with probability
$1/\left(M \exp(-\|\mathbf{Sc}\|^2/(2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{Sc} \rangle/\sigma^2\right)$ otherwise **restart**
8: $\quad \mathbf{z}_2^\dagger \leftarrow \textsc{Compress}(\mathbf{z}_2)$
9: $\quad$ **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
10: **end function**

BLISS: signature (simplified)

1: **function** $\textsc{Sign}(\mu, pk = \mathbf{a},\ sk = \mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2))$
2: $\quad \mathbf{y}_1, \mathbf{y}_2 \leftarrow D^n_{\mathbb{Z},\sigma}$ ▷ Gaussian sampling
3: $\quad \mathbf{c} \leftarrow H(\mathbf{a} \cdot \mathbf{y}_1 + \mathbf{y}_2, \mu)$ ▷ special hashing
4: $\quad$ choose a random bit b
5: $\quad \mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$
6: $\quad \mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
7: $\quad$ **continue** with probability
$1/\left(M \exp(-\|\mathbf{Sc}\|^2/(2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{Sc} \rangle/\sigma^2\right)$ otherwise **restart**
8: $\quad \mathbf{z}_2^\dagger \leftarrow \textsc{Compress}(\mathbf{z}_2)$
9: $\quad$ **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
10: **end function**

BLISS: signature (simplified)

1: **function** $\textsc{Sign}(\mu, pk = \mathbf{a},\ sk = \mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2))$
2: $\quad \mathbf{y}_1, \mathbf{y}_2 \leftarrow D^n_{\mathbb{Z},\sigma}$ ▷ Gaussian sampling
3: $\quad \mathbf{c} \leftarrow H(\mathbf{a} \cdot \mathbf{y}_1 + \mathbf{y}_2, \mu)$ ▷ special hashing
4: $\quad$ choose a random bit b
5: $\quad \mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$
6: $\quad \mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
7: $\quad$ **continue** with probability $1/\left(M \exp(-\|\mathbf{Sc}\|^2/(2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{Sc} \rangle/\sigma^2\right)$ otherwise **restart**
8: $\quad \mathbf{z}_2^\dagger \leftarrow \textsc{Compress}(\mathbf{z}_2)$
9: $\quad$ **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
10: **end function**

BLISS: signature (simplified)

1: **function** $\textsc{Sign}(\mu, pk = \mathbf{a},\ sk = \mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2))$
2: $\quad \mathbf{y}_1, \mathbf{y}_2 \leftarrow D^n_{\mathbb{Z},\sigma}$ ▷ Gaussian sampling
3: $\quad \mathbf{c} \leftarrow H(\mathbf{a} \cdot \mathbf{y}_1 + \mathbf{y}_2, \mu)$ ▷ special hashing
4: $\quad$ choose a random bit b
5: $\quad \mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$
6: $\quad \mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
7: $\quad$ **continue** with probability
$1/\left(M \exp(-\|\mathbf{Sc}\|^2/(2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{Sc} \rangle/\sigma^2\right)$ otherwise **restart**
8: $\quad \mathbf{z}_2^\dagger \leftarrow \textsc{Compress}(\mathbf{z}_2)$
9: $\quad$ **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
10: **end function**

BLISS: signature (simplified)

1: **function** $\textsc{Sign}(\mu, pk = \mathbf{a},\ sk = \mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2))$
2: $\quad \mathbf{y}_1, \mathbf{y}_2 \leftarrow D^n_{\mathbb{Z},\sigma}$ ▷ Gaussian sampling
3: $\quad \mathbf{c} \leftarrow H(\mathbf{a} \cdot \mathbf{y}_1 + \mathbf{y}_2, \mu)$ ▷ special hashing
4: $\quad$ choose a random bit b
5: $\quad \mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$
6: $\quad \mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
7: $\quad$ **continue** with probability $1/\left(M \exp(-\|\mathbf{Sc}\|^2/(2\sigma^2)) \cosh(\langle \mathbf{z}, \mathbf{Sc} \rangle/\sigma^2\right)$ otherwise **restart**
8: $\quad \mathbf{z}_2^\dagger \leftarrow \textsc{Compress}(\mathbf{z}_2)$
9: $\quad$ **return** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
10: **end function**

Outline

Overview of the CCS 2017 attack

- Attack on the rejection sampling
  - cornerstone of BLISS security/efficiency
- Straightforward implementation of rejection sampling would be inefficient: use optimized rejection algorithm
- Idea of the optimization: iterated Bernoulli trials on the bits of $\|\mathbf{Sc}\|^2$
- Side-channel leakage: can read off $\|\mathbf{Sc}\|^2$ on SPA/SEMA trace!
- From a few of these: recover $\mathbf{s_1} \cdot \bar{\mathbf{s}_1}$ ("relative norm" of the secret key)
- Then, algebraic number theory to retrieve $\mathbf{s}_1$

Overview of the CCS 2017 attack

- Attack on the rejection sampling
  - cornerstone of BLISS security/efficiency
- Straightforward implementation of rejection sampling would be inefficient: use optimized rejection algorithm
- Idea of the optimization: iterated Bernoulli trials on the bits of $\|\mathbf{Sc}\|^2$
- Side-channel leakage: can read off $\|\mathbf{Sc}\|^2$ on SPA/SEMA trace!
- From a few of these: recover $\mathbf{s_1} \cdot \bar{\mathbf{s}_1}$ ("relative norm" of the secret key)
- Then, algebraic number theory to retrieve $\mathbf{s}_1$

Overview of the CCS 2017 attack

- Attack on the rejection sampling
  - cornerstone of BLISS security/efficiency
- Straightforward implementation of rejection sampling would be inefficient: use optimized rejection algorithm
- Idea of the optimization: iterated Bernoulli trials on the bits of $\|\mathbf{Sc}\|^2$
- Side-channel leakage: can read off $\|\mathbf{Sc}\|^2$ on SPA/SEMA trace!
- From a few of these: recover $\mathbf{s_1} \cdot \bar{\mathbf{s}_1}$ ("relative norm" of the secret key)
- Then, algebraic number theory to retrieve $\mathbf{s}_1$

Overview of the CCS 2017 attack

- Attack on the rejection sampling
  - cornerstone of BLISS security/efficiency
- Straightforward implementation of rejection sampling would be inefficient: use optimized rejection algorithm
- Idea of the optimization: iterated Bernoulli trials on the bits of $\|\mathbf{Sc}\|^2$
- Side-channel leakage: can read off $\|\mathbf{Sc}\|^2$ on SPA/SEMA trace!
- From a few of these: recover $\mathbf{s_1} \cdot \bar{\mathbf{s}_1}$ ("relative norm" of the secret key)
- Then, algebraic number theory to retrieve $\mathbf{s}_1$

Overview of the CCS 2017 attack

- Attack on the rejection sampling
  - cornerstone of BLISS security/efficiency
- Straightforward implementation of rejection sampling would be inefficient: use optimized rejection algorithm
- Idea of the optimization: iterated Bernoulli trials on the bits of $\|\mathbf{Sc}\|^2$
- Side-channel leakage: can read off $\|\mathbf{Sc}\|^2$ on SPA/SEMA trace!
- From a few of these: recover $\mathbf{s_1} \cdot \bar{\mathbf{s}_1}$ ("relative norm" of the secret key)
- Then, algebraic number theory to retrieve $\mathbf{s}_1$

Overview of the CCS 2017 attack

- Attack on the rejection sampling
  - cornerstone of BLISS security/efficiency
- Straightforward implementation of rejection sampling would be inefficient: use optimized rejection algorithm
- Idea of the optimization: iterated Bernoulli trials on the bits of $\|\mathbf{Sc}\|^2$
- Side-channel leakage: can read off $\|\mathbf{Sc}\|^2$ on SPA/SEMA trace!
- From a few of these: recover $\mathbf{s_1} \cdot \bar{\mathbf{s}_1}$ ("relative norm" of the secret key)
- Then, algebraic number theory to retrieve $\mathbf{s}_1$

BLISS rejection sampling

function SAMPLEBERNEXP(x)
  for $i = 0$ to $\ell - 1$ **do**
    if $x_i = 1$ **then**
      Sample $a \leftarrow \mathscr{B}_{c_i}$
      if $a = 0$ **then return** 0
    end if
  end for
  return 1
end function ▷ $x = K - \|\mathbf{Sc}\|^2$

function SAMPLEBERNCOSH(x)
  Sample $a \leftarrow \mathscr{B}_{\exp(-x/f)}$
  if $a = 1$ **then return** 1
  Sample $b \leftarrow \mathscr{B}_{1/2}$
  if $b = 1$ **then restart**
  Sample $c \leftarrow \mathscr{B}_{\exp(-x/f)}$
  if $c = 1$ **then restart**
  return 0
end function ▷ $x = 2 \cdot \langle \mathbf{z}, \mathbf{Sc} \rangle$

Sampling algorithms for the distributions $\mathscr{B}_{\exp(-x/f)}$ and $\mathscr{B}_{1/\cosh(x/f)}$ ($c_i = 2^i/f$ precomputed)

BLISS rejection sampling

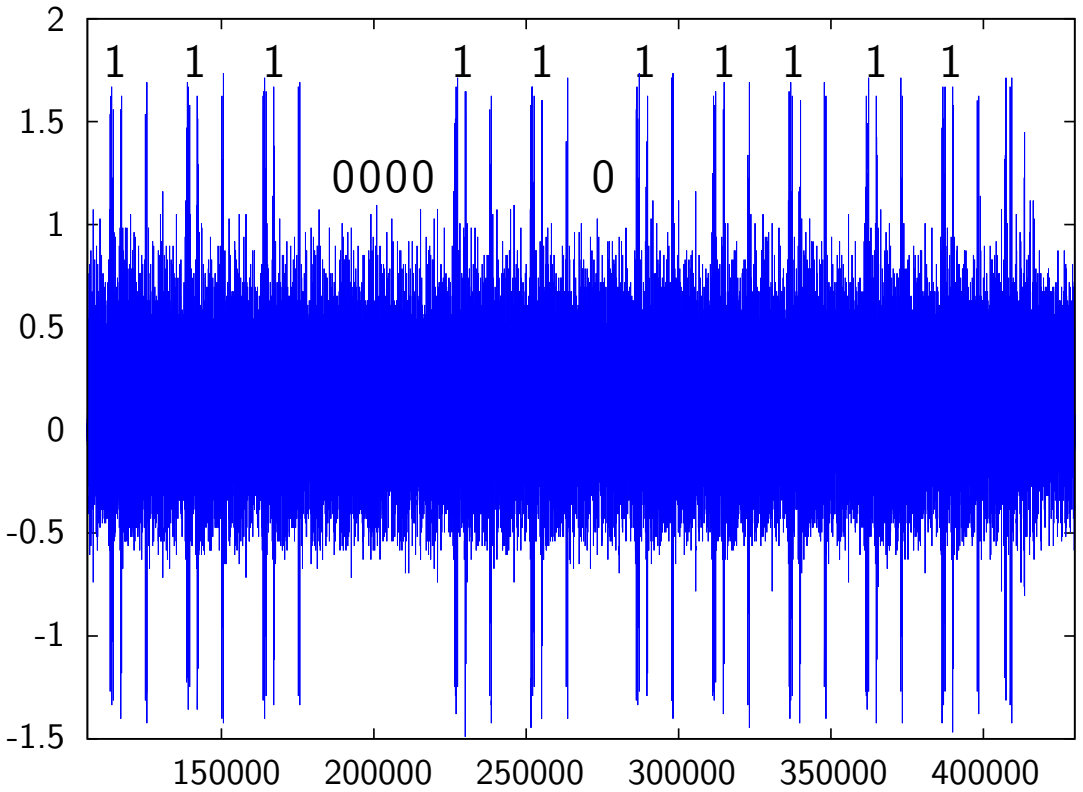
function SAMPLEBERNEXP(x)
  for $i = 0$ to $\ell - 1$ **do**
    if $x_i = 1$ **then**
      Sample $a \leftarrow \mathscr{B}_{c_i}$
      if $a = 0$ **then return** 0
    end if
  end for
  return 1
end function ▷ $x = K - \|\mathbf{Sc}\|^2$

function SAMPLEBERNCOSH(x)
  Sample $a \leftarrow \mathscr{B}_{\exp(-x/f)}$
  if $a = 1$ **then return** 1
  Sample $b \leftarrow \mathscr{B}_{1/2}$
  if $b = 1$ **then restart**
  Sample $c \leftarrow \mathscr{B}_{\exp(-x/f)}$
  if $c = 1$ **then restart**
  return 0
end function ▷ $x = 2 \cdot \langle \mathbf{z}, \mathbf{Sc} \rangle$

Sampling algorithms for the distributions $\mathscr{B}_{\exp(-x/f)}$ and $\mathscr{B}_{1/\cosh(x/f)}$ ($c_i = 2^i/f$ precomputed)

Experimental leakage

EMA trace of BLISS rejection sampling on 8-bit AVR for norm $\|\mathbf{Sc}\|^2 = 14404$. One reads the value:
$K - \|\mathbf{Sc}\|^2 = 46539 - 14404 = 32135 = \overline{111110110000111}_2$



Exploiting the leakage

- From each trace, get:

$$\|\mathbf{Sc}\|^2 = \|\mathbf{s}_1 \cdot \mathbf{c}\|^2 + \|\mathbf{s}_2 \cdot \mathbf{c}\|^2$$

  for known $\mathbf{c}$, different each time
- Linear equation on $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ and $\mathbf{s}_2 \cdot \bar{\mathbf{s}}_2$
- Collect $\approx 2 \times 256 = 512$ to recover the relative norms $\mathbf{s}_i \cdot \bar{\mathbf{s}}_i$
- Going from $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ to $\mathbf{s}_1$: algebraic number theory
  - computationally costly (Howgrave-Graham–Szydlo)
  - only possible if we can factor $N_{K/\mathbb{Q}}(\mathbf{s}_1)$ ($\approx 7\%$ of weak keys)

Exploiting the leakage

- From each trace, get:

$$\|\mathbf{Sc}\|^2 = \|\mathbf{s}_1 \cdot \mathbf{c}\|^2 + \|\mathbf{s}_2 \cdot \mathbf{c}\|^2$$

  for known $\mathbf{c}$, different each time
- Linear equation on $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ and $\mathbf{s}_2 \cdot \bar{\mathbf{s}}_2$
- Collect $\approx 2 \times 256 = 512$ to recover the relative norms $\mathbf{s}_i \cdot \bar{\mathbf{s}}_i$
- Going from $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ to $\mathbf{s}_1$: algebraic number theory
  - computationally costly (Howgrave-Graham–Szydlo)
  - only possible if we can factor $N_{K/\mathbb{Q}}(\mathbf{s}_1)$ ($\approx 7\%$ of weak keys)

Exploiting the leakage

- From each trace, get:

$$\|\mathbf{Sc}\|^2 = \|\mathbf{s}_1 \cdot \mathbf{c}\|^2 + \|\mathbf{s}_2 \cdot \mathbf{c}\|^2$$

  for known $\mathbf{c}$, different each time
- Linear equation on $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ and $\mathbf{s}_2 \cdot \bar{\mathbf{s}}_2$
- Collect $\approx 2 \times 256 = 512$ to recover the relative norms $\mathbf{s}_i \cdot \bar{\mathbf{s}}_i$
- Going from $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ to $\mathbf{s}_1$: algebraic number theory
  - computationally costly (Howgrave-Graham–Szydlo)
  - only possible if we can factor $N_{K/\mathbb{Q}}(\mathbf{s}_1)$ ($\approx 7\%$ of weak keys)

Exploiting the leakage

- From each trace, get:

$$\|\mathbf{Sc}\|^2 = \|\mathbf{s}_1 \cdot \mathbf{c}\|^2 + \|\mathbf{s}_2 \cdot \mathbf{c}\|^2$$

  for known $\mathbf{c}$, different each time
- Linear equation on $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ and $\mathbf{s}_2 \cdot \bar{\mathbf{s}}_2$
- Collect $\approx 2 \times 256 = 512$ to recover the relative norms $\mathbf{s}_i \cdot \bar{\mathbf{s}}_i$
- Going from $\mathbf{s}_1 \cdot \bar{\mathbf{s}}_1$ to $\mathbf{s}_1$: algebraic number theory
  - computationally costly (Howgrave-Graham–Szydlo)
  - only possible if we can factor $N_{K/\mathbb{Q}}(\mathbf{s}_1)$ ($\approx 7\%$ of weak keys)

Outline

Outline

What about the inner product leakage? (I)

- Recall the rejection sampling probability of BLISS signing:

$$1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2} \right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2} \right) \right),$$

- The exp part of the rejection sampling leaks $\|\mathbf{Sc}\|^2$ and ultimately the relative norm of $\mathbf{s}_1$ and $\mathbf{s}_2$: what we have used so far
- Can't we use the cosh part instead? It directly leaks:

$$\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c} \rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c} \rangle$$

- If we know $(\mathbf{c}, \mathbf{z}_1, \mathbf{z}_2)$, this gives a *linear* relation on the secret: recover everything from around 1024 signatures without breaking a sweat!

What about the inner product leakage? (I)

- Recall the rejection sampling probability of BLISS signing:

$$1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2} \right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2} \right) \right),$$

- The exp part of the rejection sampling leaks $\|\mathbf{Sc}\|^2$ and ultimately the relative norm of $\mathbf{s}_1$ and $\mathbf{s}_2$: what we have used so far
- Can't we use the cosh part instead? It directly leaks:

$$\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c} \rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c} \rangle$$

- If we know $(\mathbf{c}, \mathbf{z}_1, \mathbf{z}_2)$, this gives a *linear* relation on the secret: recover everything from around 1024 signatures without breaking a sweat!

What about the inner product leakage? (I)

- Recall the rejection sampling probability of BLISS signing:

$$1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2} \right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2} \right) \right),$$

- The exp part of the rejection sampling leaks $\|\mathbf{Sc}\|^2$ and ultimately the relative norm of $\mathbf{s}_1$ and $\mathbf{s}_2$: what we have used so far
- Can't we use the cosh part instead? It directly leaks:

$$\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c} \rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c} \rangle$$

- If we know $(\mathbf{c}, \mathbf{z}_1, \mathbf{z}_2)$, this gives a *linear* relation on the secret: recover everything from around 1024 signatures without breaking a sweat!

What about the inner product leakage? (I)

- Recall the rejection sampling probability of BLISS signing:

$$1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2} \right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2} \right) \right),$$

- The exp part of the rejection sampling leaks $\|\mathbf{Sc}\|^2$ and ultimately the relative norm of $\mathbf{s}_1$ and $\mathbf{s}_2$: what we have used so far
- Can't we use the cosh part instead? It directly leaks:

$$\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c} \rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c} \rangle$$

- If we know $(\mathbf{c}, \mathbf{z}_1, \mathbf{z}_2)$, this gives a *linear* relation on the secret: recover everything from around 1024 signatures without breaking a sweat!

What about the inner product leakage? (II)

- Problem: signatures do not contain $\mathbf{z}_2$, but only a compressed variant $\mathbf{z}_2^\dagger$, and compression is lossy
  - only obtain noisy linear system in the secret key
- First reaction: like LWE in twice the original dimension, so probably hopeless
- Second opinion: not hopeless at all, because there is no modular reduction

What about the inner product leakage? (II)

- Problem: signatures do not contain $\mathbf{z}_2$, but only a compressed variant $\mathbf{z}_2^\dagger$, and compression is lossy
  - only obtain noisy linear system in the secret key
- First reaction: like LWE in twice the original dimension, so probably hopeless
- Second opinion: not hopeless at all, because there is no modular reduction

What about the inner product leakage? (II)

- Problem: signatures do not contain $\mathbf{z}_2$, but only a compressed variant $\mathbf{z}_2^\dagger$, and compression is lossy
  - only obtain noisy linear system in the secret key
- First reaction: like LWE in twice the original dimension, so probably hopeless
- Second opinion: not hopeless at all, because there is no modular reduction

More precise description of the leakage

$$\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle = \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle 2^d\mathbf{z}_2^\dagger + (\mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger), \mathbf{s}_2\mathbf{c}\rangle$$
$$= \langle \mathbf{z}_1\mathbf{c}^*, \mathbf{s}_1\rangle + \langle 2^d\mathbf{z}_2^\dagger\mathbf{c}^*, \mathbf{s}_2\rangle + \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle$$
$$b = \langle \mathbf{a}, \mathbf{s}\rangle + e$$

where

$\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2)$ (secret key)

$\mathbf{a} = \left(\mathbf{z}_1\mathbf{c}^*, 2^d\mathbf{z}_2^\dagger\mathbf{c}^*\right)$ (known from sig.)

$b = \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle$ (leakage)

$e = \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle$ (small unknown value)

More precise description of the leakage

$$\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle = \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle 2^d\mathbf{z}_2^\dagger + (\mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger), \mathbf{s}_2\mathbf{c}\rangle$$

$$= \langle \mathbf{z}_1\mathbf{c}^*, \mathbf{s}_1\rangle + \langle 2^d\mathbf{z}_2^\dagger\mathbf{c}^*, \mathbf{s}_2\rangle + \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle$$

$$b = \langle \mathbf{a}, \mathbf{s}\rangle + e$$

where

$$\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2) \quad \text{(secret key)}$$

$$\mathbf{a} = \left(\mathbf{z}_1\mathbf{c}^*, 2^d\mathbf{z}_2^\dagger\mathbf{c}^*\right) \quad \text{(known from sig.)}$$

$$b = \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle \quad \text{(leakage)}$$

$$e = \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle \quad \text{(small unknown value)}$$

More precise description of the leakage

$$\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle = \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle 2^d\mathbf{z}_2^\dagger + (\mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger), \mathbf{s}_2\mathbf{c}\rangle$$
$$= \langle \mathbf{z}_1\mathbf{c}^*, \mathbf{s}_1\rangle + \langle 2^d\mathbf{z}_2^\dagger\mathbf{c}^*, \mathbf{s}_2\rangle + \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle$$
$$b = \langle \mathbf{a}, \mathbf{s}\rangle + e$$

where

$\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2)$ (secret key)

$\mathbf{a} = \left(\mathbf{z}_1\mathbf{c}^*, 2^d\mathbf{z}_2^\dagger\mathbf{c}^*\right)$ (known from sig.)

$b = \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle$ (leakage)

$e = \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle$ (small unknown value)

More precise description of the leakage

$$\begin{aligned}\langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle &= \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle 2^d\mathbf{z}_2^\dagger + (\mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger), \mathbf{s}_2\mathbf{c}\rangle \\ &= \langle \mathbf{z}_1\mathbf{c}^*, \mathbf{s}_1\rangle + \langle 2^d\mathbf{z}_2^\dagger\mathbf{c}^*, \mathbf{s}_2\rangle + \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle \\ b &= \langle \mathbf{a}, \mathbf{s}\rangle + e\end{aligned}$$

where

$$\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2) \qquad \text{(secret key)}$$

$$\mathbf{a} = \left(\mathbf{z}_1\mathbf{c}^*, 2^d\mathbf{z}_2^\dagger\mathbf{c}^*\right) \qquad \text{(known from sig.)}$$

$$b = \langle \mathbf{z}_1, \mathbf{s}_1\mathbf{c}\rangle + \langle \mathbf{z}_2, \mathbf{s}_2\mathbf{c}\rangle \qquad \text{(leakage)}$$

$$e = \langle \mathbf{z}_2 - 2^d\mathbf{z}_2^\dagger, \mathbf{s}_2\mathbf{c}\rangle \qquad \text{(small unknown value)}$$

Outline

Abstracting the situation: Integer-LWE

- $\mathbf{s}$ secret vector in $\mathbb{Z}^n$
- χ_a, χ_e probability distributions over $\mathbb{Z}$

Integer-LWE Problem

Given m samples $(\mathbf{a}_i, b_i)$ of the form:

$$\mathbf{a}_i \leftarrow \chi_a^n \qquad b_i = \langle \mathbf{a}, \mathbf{s} \rangle + e \quad (e \leftarrow \chi_e)$$

find $\mathbf{s}$.

Like LWE, without the modular reduction.
Can we solve this efficiently?

Abstracting the situation: Integer-LWE

- $\mathbf{s}$ secret vector in $\mathbb{Z}^n$
- χ_a, χ_e probability distributions over $\mathbb{Z}$

Integer-LWE Problem

Given m samples $(\mathbf{a}_i, b_i)$ of the form:

$$\mathbf{a}_i \leftarrow \chi_a^n \qquad b_i = \langle \mathbf{a}, \mathbf{s} \rangle + e \quad (e \leftarrow \chi_e)$$

find $\mathbf{s}$.

Like LWE, without the modular reduction.

Can we solve this efficiently?

Abstracting the situation: Integer-LWE

- $\mathbf{s}$ secret vector in $\mathbb{Z}^n$
- χ_a, χ_e probability distributions over $\mathbb{Z}$

Integer-LWE Problem

Given m samples $(\mathbf{a}_i, b_i)$ of the form:

$$\mathbf{a}_i \leftarrow \chi_a^n \qquad b_i = \langle \mathbf{a}, \mathbf{s} \rangle + e \quad (e \leftarrow \chi_e)$$

find $\mathbf{s}$.

Like LWE, without the modular reduction.
Can we solve this efficiently?

Our main result

Integer-LWE is easy

Suppose χ_a, χ_e are centered distributions of std. dev. σ_a, σ_e. Under some technical conditions, we can recover $\mathbf{s}$ with m samples for some

$$m = \Omega\left(\left(\frac{\sigma_e}{\sigma_a}\right)^2 \cdot \log n\right),$$

in time polynomial in m, n.

- In particular, unless σ_e is superpolynomially large, we can always recover $\mathbf{s}$ with poly-many samples
- Technical condition: essentially that χ_e, χ_a are subgaussian distributions
- We also need $m \geq Cn$ for some explicit constant C
- Proposed algorithm: least squares regression

Our main result

Integer-LWE is easy

Suppose χ_a, χ_e are centered distributions of std. dev. σ_a, σ_e. Under some technical conditions, we can recover $\mathbf{s}$ with m samples for some

$$m = \Omega\left(\left(\frac{\sigma_e}{\sigma_a}\right)^2 \cdot \log n\right),$$

in time polynomial in m, n.

- In particular, unless σ_e is superpolynomially large, we can always recover $\mathbf{s}$ with poly-many samples
- Technical condition: essentially that χ_e, χ_a are subgaussian distributions
- We also need $m \geq Cn$ for some explicit constant C
- Proposed algorithm: least squares regression

Our main result

Integer-LWE is easy

Suppose χ_a, χ_e are centered distributions of std. dev. σ_a, σ_e. Under some technical conditions, we can recover $\mathbf{s}$ with m samples for some

$$m = \Omega\left(\left(\frac{\sigma_e}{\sigma_a}\right)^2 \cdot \log n\right),$$

in time polynomial in m, n.

- In particular, unless σ_e is superpolynomially large, we can always recover $\mathbf{s}$ with poly-many samples
- Technical condition: essentially that χ_e, χ_a are subgaussian distributions
- We also need $m \geq Cn$ for some explicit constant C
- Proposed algorithm: least squares regression

Our main result

Integer-LWE is easy

Suppose χ_a, χ_e are centered distributions of std. dev. σ_a, σ_e. Under some technical conditions, we can recover $\mathbf{s}$ with m samples for some

$$m = \Omega\left(\left(\frac{\sigma_e}{\sigma_a}\right)^2 \cdot \log n\right),$$

in time polynomial in m, n.

- In particular, unless σ_e is superpolynomially large, we can always recover $\mathbf{s}$ with poly-many samples
- Technical condition: essentially that χ_e, χ_a are subgaussian distributions
- We also need $m \geq Cn$ for some explicit constant C
- Proposed algorithm: least squares regression

The least squares approach (I)

- Given $m > n$ integer-LWE samples, we can put them in matrix form:

$$A \in \mathbb{Z}^{m \times n} \qquad \mathbf{b} = A\mathbf{s} + \mathbf{e} \quad (\mathbf{b}, \mathbf{e} \in \mathbb{Z}^m)$$

- Overdetermined linear system with errors. Least squares: find $\tilde{\mathbf{s}} \in \mathbb{R}^n$ minimizing $\|A\tilde{\mathbf{s}} - \mathbf{b}\|_2^2$
- Solution:

$$\tilde{\mathbf{s}} = (A^T A)^{-1} A^T \mathbf{b}$$

- Claim: under the hypotheses of the theorem, rounding this $\tilde{\mathbf{s}}$ gives the secret $\mathbf{s}$ w.h.p.
- (Only makes sense if $A^T A$ invertible, but this is the case for $m \geq Cn$; in fact, we then have $A^T A \approx m\sigma_a^2 \cdot I_n$)

The least squares approach (I)

- Given $m > n$ integer-LWE samples, we can put them in matrix form:

$$A \in \mathbb{Z}^{m \times n} \qquad \mathbf{b} = A\mathbf{s} + \mathbf{e} \quad (\mathbf{b}, \mathbf{e} \in \mathbb{Z}^m)$$

- Overdetermined linear system with errors. Least squares: find $\tilde{\mathbf{s}} \in \mathbb{R}^n$ minimizing $\|A\tilde{\mathbf{s}} - \mathbf{b}\|_2^2$
- Solution:

$$\tilde{\mathbf{s}} = (A^T A)^{-1} A^T \mathbf{b}$$

- Claim: under the hypotheses of the theorem, rounding this $\tilde{\mathbf{s}}$ gives the secret $\mathbf{s}$ w.h.p.
- (Only makes sense if $A^T A$ invertible, but this is the case for $m \geq Cn$; in fact, we then have $A^T A \approx m\sigma_a^2 \cdot I_n$)

The least squares approach (I)

- Given $m > n$ integer-LWE samples, we can put them in matrix form:

$$A \in \mathbb{Z}^{m \times n} \qquad \mathbf{b} = A\mathbf{s} + \mathbf{e} \quad (\mathbf{b}, \mathbf{e} \in \mathbb{Z}^m)$$

- Overdetermined linear system with errors. Least squares: find $\tilde{\mathbf{s}} \in \mathbb{R}^n$ minimizing $\|A\tilde{\mathbf{s}} - \mathbf{b}\|_2^2$
- Solution:

$$\tilde{\mathbf{s}} = (A^T A)^{-1} A^T \mathbf{b}$$

- Claim: under the hypotheses of the theorem, rounding this $\tilde{\mathbf{s}}$ gives the secret $\mathbf{s}$ w.h.p.
- (Only makes sense if $A^T A$ invertible, but this is the case for $m \geq Cn$; in fact, we then have $A^T A \approx m\sigma_a^2 \cdot I_n$)

The least squares approach (I)

- Given $m > n$ integer-LWE samples, we can put them in matrix form:

$$A \in \mathbb{Z}^{m \times n} \qquad \mathbf{b} = A\mathbf{s} + \mathbf{e} \quad (\mathbf{b}, \mathbf{e} \in \mathbb{Z}^m)$$

- Overdetermined linear system with errors. Least squares: find $\tilde{\mathbf{s}} \in \mathbb{R}^n$ minimizing $\|A\tilde{\mathbf{s}} - \mathbf{b}\|_2^2$
- Solution:

$$\tilde{\mathbf{s}} = (A^T A)^{-1} A^T \mathbf{b}$$

- Claim: under the hypotheses of the theorem, rounding this $\tilde{\mathbf{s}}$ gives the secret $\mathbf{s}$ w.h.p.
- (Only makes sense if $A^T A$ invertible, but this is the case for $m \geq Cn$; in fact, we then have $A^T A \approx m\sigma_a^2 \cdot I_n$)

The least squares approach (I)

- Given $m > n$ integer-LWE samples, we can put them in matrix form:

$$A \in \mathbb{Z}^{m \times n} \qquad \mathbf{b} = A\mathbf{s} + \mathbf{e} \quad (\mathbf{b}, \mathbf{e} \in \mathbb{Z}^m)$$

- Overdetermined linear system with errors. Least squares: find $\tilde{\mathbf{s}} \in \mathbb{R}^n$ minimizing $\|A\tilde{\mathbf{s}} - \mathbf{b}\|_2^2$
- Solution:

$$\tilde{\mathbf{s}} = (A^T A)^{-1} A^T \mathbf{b}$$

- Claim: under the hypotheses of the theorem, rounding this $\tilde{\mathbf{s}}$ gives the secret $\mathbf{s}$ w.h.p.
- (Only makes sense if $A^T A$ invertible, but this is the case for $m \geq Cn$; in fact, we then have $A^T A \approx m\sigma_a^2 \cdot I_n$)

The least squares approach (II)

- Proof idea: it suffices to prove that, w.h.p.:

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty < 1/2.$$

- But we have:

$$\begin{aligned}\tilde{\mathbf{s}} - \mathbf{s} &= (A^T A)^{-1} A^T \mathbf{b} - \mathbf{s} \\ &= (A^T A)^{-1} A^T \Big(A\mathbf{s} + \mathbf{e} \Big) - \mathbf{s} = (A^T A)^{-1} A^T \mathbf{e}\end{aligned}$$

- The argument then goes like this:

$$\ldots$$

- Putting all together, we get that

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty = O\Big(\frac{\sigma_e}{\sigma_a} \cdot \frac{\sqrt{\log n}}{m} \Big)$$

hence the result

The least squares approach (II)

- Proof idea: it suffices to prove that, w.h.p.:

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty < 1/2.$$

- But we have:

$$\begin{aligned}\tilde{\mathbf{s}} - \mathbf{s} &= (A^T A)^{-1} A^T \mathbf{b} - \mathbf{s} \\ &= (A^T A)^{-1} A^T \Big(A\mathbf{s} + \mathbf{e} \Big) - \mathbf{s} = (A^T A)^{-1} A^T \mathbf{e}\end{aligned}$$

- The argument then goes like this:

$$\ldots$$

- Putting all together, we get that

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty = O\Big(\frac{\sigma_e}{\sigma_a} \cdot \frac{\sqrt{\log n}}{m} \Big)$$

hence the result

The least squares approach (II)

- Proof idea: it suffices to prove that, w.h.p.:

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty < 1/2.$$

- But we have:

$$\begin{aligned}\tilde{\mathbf{s}} - \mathbf{s} &= (A^T A)^{-1} A^T \mathbf{b} - \mathbf{s} \\ &= (A^T A)^{-1} A^T \Big(A\mathbf{s} + \mathbf{e} \Big) - \mathbf{s} = (A^T A)^{-1} A^T \mathbf{e}\end{aligned}$$

- The argument then goes like this:
  1. $\mathbf{e}$ is a subgaussian vector of parameter (essentially) σ_e
  2. hence, $M\mathbf{e}$ is also a subgaussian for any matrix M, with parameter $\leq \sigma_e \cdot s_{\max}(M^T)$ (largest singular value)
  3. $s_{\max}(M^T) = \lambda_{\min}(A^T A)^{-1/2} \approx \frac{1}{\sigma_a \sqrt{m}}$
  4. w.h.p., a subgaussian vector $\mathbf{x} \in \mathbb{R}^n$ of parameter τ satisfies $\|\mathbf{x}\|_\infty = O(\tau \sqrt{\log n})$
- Putting all together, we get that

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty = O\Big(\frac{\sigma_e}{\sigma_a} \cdot \frac{\sqrt{\log n}}{\sqrt{m}} \Big)$$

The least squares approach (II)

- Proof idea: it suffices to prove that, w.h.p.:

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty < 1/2.$$

- But we have:

$$\begin{aligned}\tilde{\mathbf{s}} - \mathbf{s} &= (A^T A)^{-1} A^T \mathbf{b} - \mathbf{s} \\ &= (A^T A)^{-1} A^T \Big(A\mathbf{s} + \mathbf{e} \Big) - \mathbf{s} = (A^T A)^{-1} A^T \mathbf{e}\end{aligned}$$

- The argument then goes like this:

$$\ldots$$

- Putting all together, we get that

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_\infty = O\Big(\frac{\sigma_e}{\sigma_a} \cdot \frac{\sqrt{\log n}}{m} \Big)$$

  hence the result

Outline

Optimality

Minimal number of samples to solve ILWE

Suppose χ_e is either the uniform distribution on an integer interval, or a discrete Gaussian supported on $\mathbb{Z}$. Then, solving the ILWE problem requires at least

$$m = \Omega\left(\left(\frac{\sigma_e}{\sigma_a}\right)^2\right)$$

samples, even information-theoretically.

- Proved by estimating the statistical distance between the ILWE distribution for distinct secrets $\mathbf{s}$ and $\mathbf{s}'$
- Same bound as above, up to a $O(\log n)$ factor
- Hence, the least-squares approach is sample-optimal up to a log factor

Optimality

Minimal number of samples to solve ILWE

Suppose χ_e is either the uniform distribution on an integer interval, or a discrete Gaussian supported on $\mathbb{Z}$. Then, solving the ILWE problem requires at least

$$m = \Omega\left(\left(\frac{\sigma_e}{\sigma_a}\right)^2\right)$$

samples, even information-theoretically.

- Proved by estimating the statistical distance between the ILWE distribution for distinct secrets $\mathbf{s}$ and $\mathbf{s}'$
- Same bound as above, up to a $O(\log n)$ factor
- Hence, the least-squares approach is sample-optimal up to a log factor

Optimality

Minimal number of samples to solve ILWE

Suppose χ_e is either the uniform distribution on an integer interval, or a discrete Gaussian supported on $\mathbb{Z}$. Then, solving the ILWE problem requires at least

$$m = \Omega\left(\left(\frac{\sigma_e}{\sigma_a}\right)^2\right)$$

samples, even information-theoretically.

- Proved by estimating the statistical distance between the ILWE distribution for distinct secrets $\mathbf{s}$ and $\mathbf{s}'$
- Same bound as above, up to a $O(\log n)$ factor
- Hence, the least-squares approach is sample-optimal up to a log factor

Application to BLISS

- The learning problem arising from the leakage in BLISS is not exactly ILWE:
  - the coefficients of $\mathbf{a}$ are not i.i.d.
  - $\mathbf{a}$ and $\mathbf{e}$ may not be independent
- However, close enough that the same appraoch works as expected
- Required number of samples also in line with estimates:
  - BLISS–0: key recovery with ≈ 1400 samples
  - BLISS–I/II: key recovery with ≈ 20000 samples
- Compared to the attack of CCS 2017:
  - much faster attack (just linear algebra! recovery in seconds)
  - works against 100% of secret keys
  - drawback: requires more traces

Application to BLISS

- The learning problem arising from the leakage in BLISS is not exactly ILWE:
  - the coefficients of $\mathbf{a}$ are not i.i.d.
  - $\mathbf{a}$ and $\mathbf{e}$ may not be independent
- However, close enough that the same appraoch works as expected
- Required number of samples also in line with estimates:
  - BLISS–0: key recovery with ≈ 1400 samples
  - BLISS–I/II: key recovery with ≈ 20000 samples
- Compared to the attack of CCS 2017:
  - much faster attack (just linear algebra! recovery in seconds)
  - works against 100% of secret keys
  - drawback: requires more traces

Application to BLISS

- The learning problem arising from the leakage in BLISS is not exactly ILWE:
  - the coefficients of $\mathbf{a}$ are not i.i.d.
  - $\mathbf{a}$ and $\mathbf{e}$ may not be independent
- However, close enough that the same appraoch works as expected
- Required number of samples also in line with estimates:
  - BLISS–0: key recovery with ≈ 1400 samples
  - BLISS–I/II: key recovery with ≈ 20000 samples
- Compared to the attack of CCS 2017:
  - much faster attack (just linear algebra! recovery in seconds)
  - works against 100% of secret keys
  - drawback: requires more traces

Application to BLISS

- The learning problem arising from the leakage in BLISS is not exactly ILWE:
  - the coefficients of $\mathbf{a}$ are not i.i.d.
  - $\mathbf{a}$ and $\mathbf{e}$ may not be independent
- However, close enough that the same appraoch works as expected
- Required number of samples also in line with estimates:
  - BLISS–0: key recovery with ≈ 1400 samples
  - BLISS–I/II: key recovery with ≈ 20000 samples
- Compared to the attack of CCS 2017:
  - much faster attack (just linear algebra! recovery in seconds)
  - works against 100% of secret keys
  - drawback: requires more traces

Other results on ILWE

- ILWE as a CVP_∞ problem:
  - least-squares regression amounts to solving CVP_∞ by Babai's rounding in the lattice $L = A^T A \cdot \mathbb{Z}^n$
  - however, access to an exact CVP_p oracle (for any $p \in [2, \infty]$) does not improve the sample complexity: $\log n$ factor remains
- ILWE and linear programming:
  - LP techniques do not appear to bring an improvement either for general secrets
  - however, for very sparse secrets, recovery can be possible even with $m \ll n$ samples (impossible with least squares) using LP/compressed sensing

Other results on ILWE

- ILWE as a CVP_∞ problem:
  - least-squares regression amounts to solving CVP_∞ by Babai's rounding in the lattice $L = A^T A \cdot \mathbb{Z}^n$
  - however, access to an exact CVP_p oracle (for any $p \in [2, \infty]$) does not improve the sample complexity: $\log n$ factor remains
- ILWE and linear programming:
  - LP techniques do not appear to bring an improvement either for general secrets
  - however, for very sparse secrets, recovery can be possible even with $m \ll n$ samples (impossible with least squares) using LP/compressed sensing

Thank you!