Session #9: Trapdoors and Applications

Chris Peikert
Georgia Institute of Technology

Winter School on Lattice-Based Cryptography and Applications
Bar-Ilan University, Israel
19 Feb 2012 – 22 Feb 2012

Agenda

1. Lattices and short 'trapdoor' bases

2. Lattice-based 'preimage sampleable' functions

3. Applications: signatures, ID-based encryption (in RO model)

Digital Signatures

(Images courtesy xkcd.org)

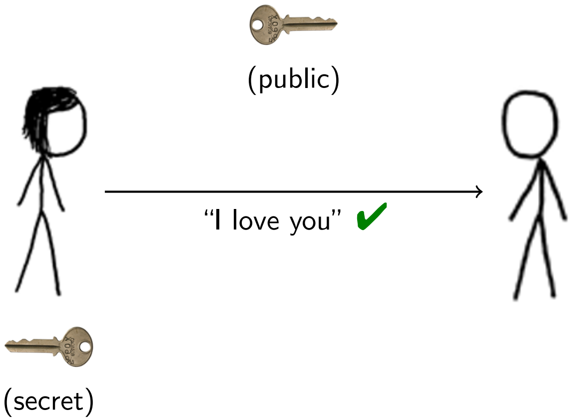
Digital Signatures

(public)

(secret)

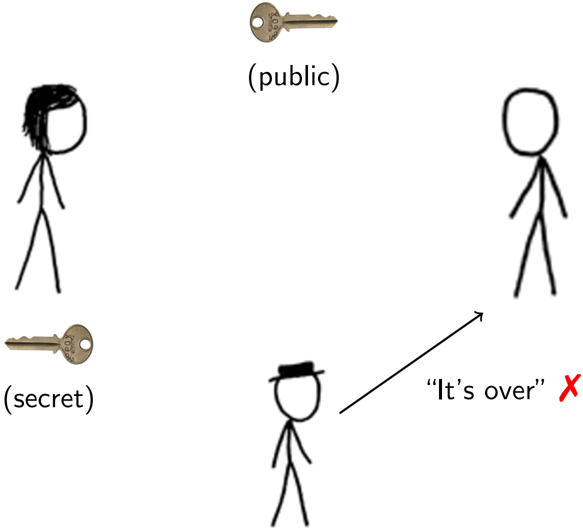
(Images courtesy xkcd.org)

Digital Signatures



(Images courtesy xkcd.org)

Digital Signatures



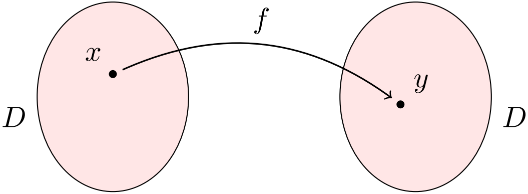
(Images courtesy xkcd.org)

Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}

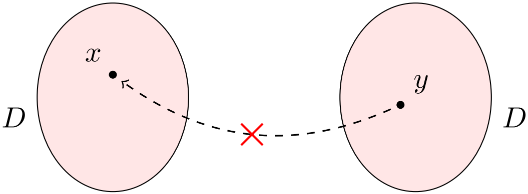
Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- Trapdoor permutation [DH'76,RSA'77,...] (PSF)



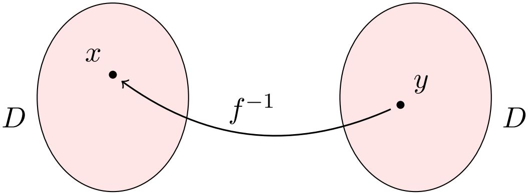
Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- Trapdoor permutation [DH'76,RSA'77,...] (PSF)



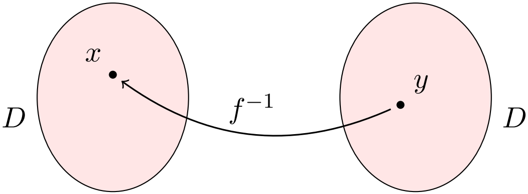
Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- Trapdoor permutation [DH'76,RSA'77,...] (PSF)



Central Tool: Trapdoor Functions

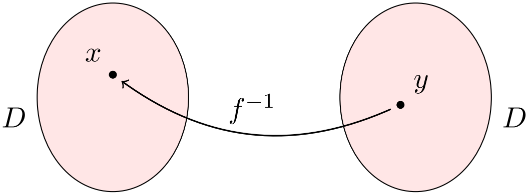
- Public function f generated with secret 'trapdoor' f^{-1}
- Trapdoor permutation [DH'76,RSA'77,...] (PSF)



- 'Hash and sign:' $pk = f$, $sk = f^{-1}$. $\mathsf{Sign}(\mathsf{msg}) = f^{-1}(H(\mathsf{msg}))$.

Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- Trapdoor permutation [DH'76,RSA'77,...] (PSF)



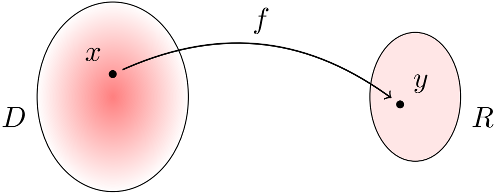
- 'Hash and sign:' $pk = f$, $sk = f^{-1}$. Sign(msg) $= f^{-1}(H(\text{msg}))$.
- Candidate TDPs: [RSA'78,Rabin'79,Paillier'99] ('general assumption')

  All rely on hardness of factoring:

  - ✗ Complex: 2048-bit exponentiation
  - ✗ Broken by quantum algorithms [Shor'97]

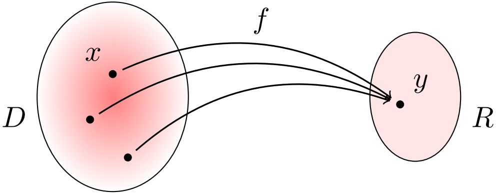
Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- New twist [GPV'08]: preimage sampleable trapdoor function (PSF)



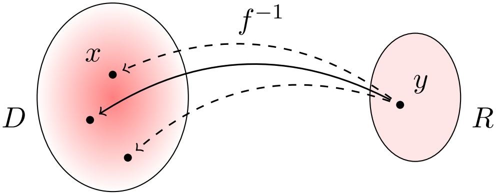
Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- New twist [GPV'08]: preimage sampleable trapdoor function (PSF)



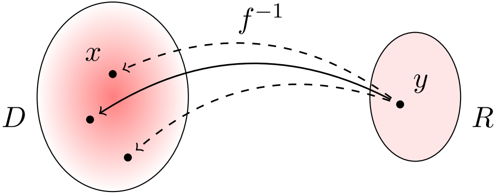
Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- New twist [GPV'08]: preimage sampleable trapdoor function (PSF)



Central Tool: Trapdoor Functions

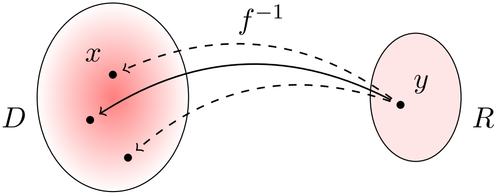
- Public function f generated with secret 'trapdoor' f^{-1}
- New twist [GPV'08]: preimage sampleable trapdoor function (PSF)



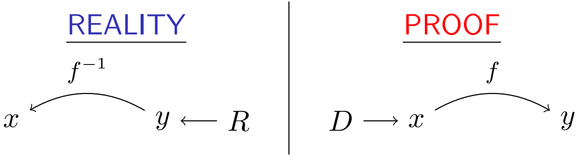
- 'Hash and sign:' $pk = f$, $sk = f^{-1}$. Sign(msg) $= f^{-1}(H(\text{msg}))$.

Central Tool: Trapdoor Functions

- Public function f generated with secret 'trapdoor' f^{-1}
- New twist [GPV'08]: preimage sampleable trapdoor function (PSF)



- 'Hash and sign:' $pk = f$, $sk = f^{-1}$. Sign(msg) $= f^{-1}(H(\text{msg}))$.
- Still secure! Can generate (x, y) in two equivalent ways:

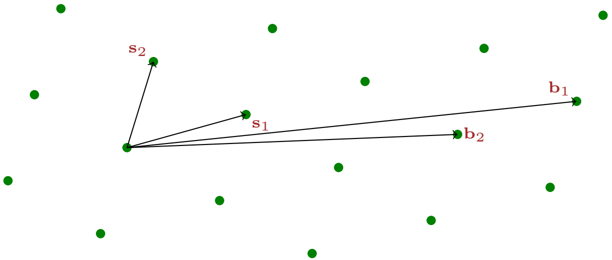


Part 1:

Constructing Preimage Sampleable Trapdoor Functions (PSFs)

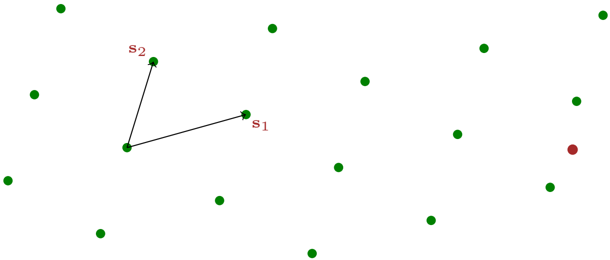
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$



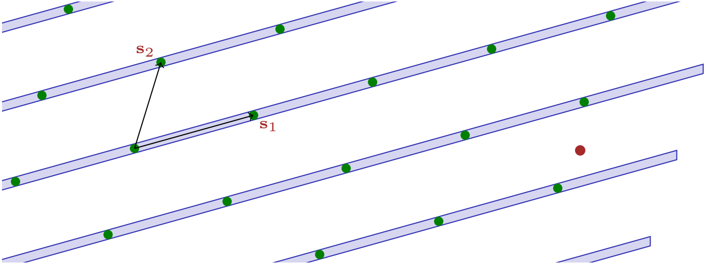
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: pk = 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, sk = 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



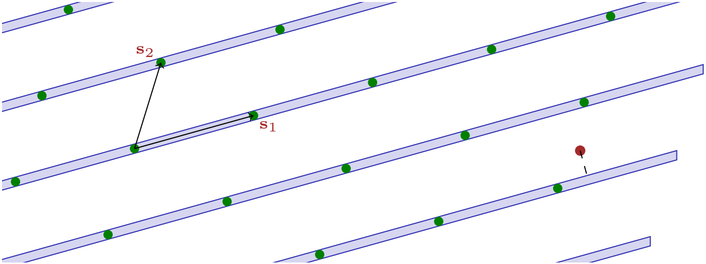
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



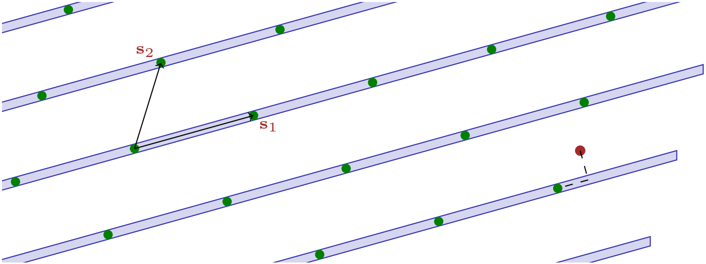
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



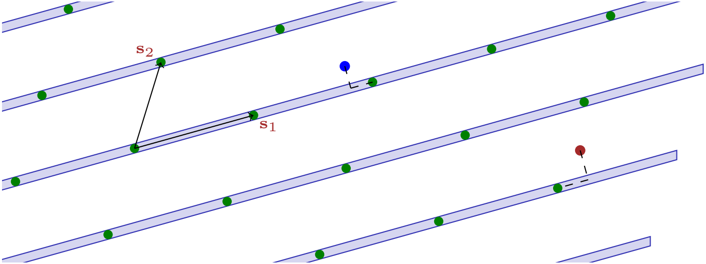
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



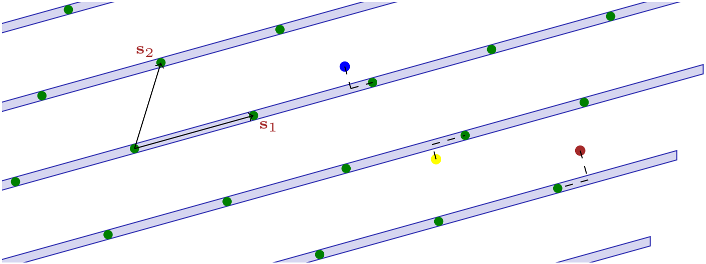
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



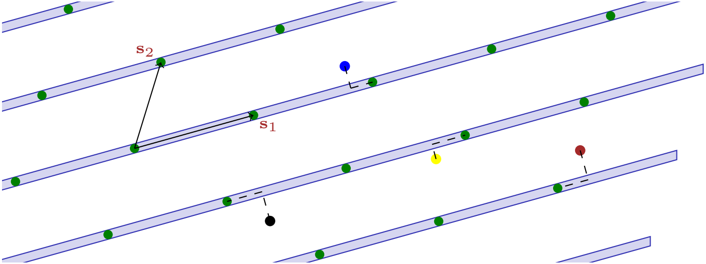
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: pk = 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, sk = 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



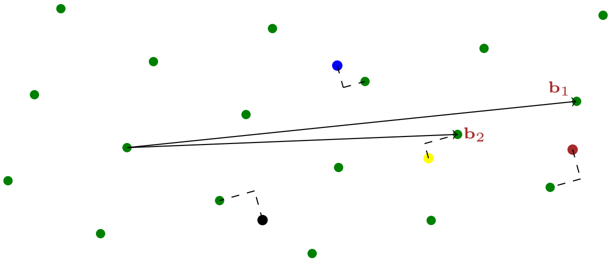
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



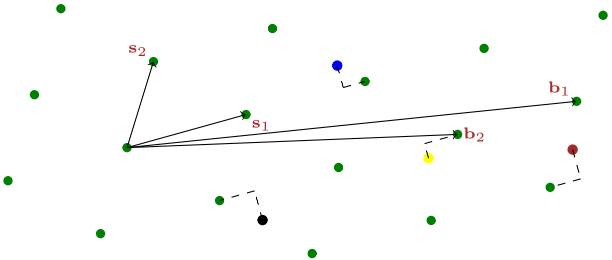
Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]



Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: pk = 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, sk = 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\mathsf{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]

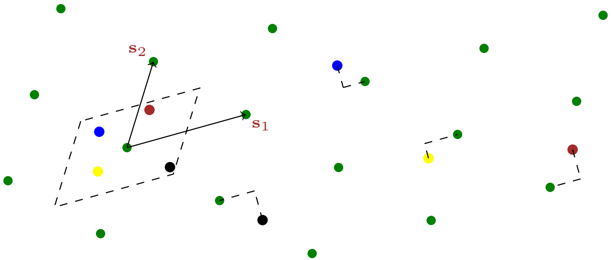


Technical Issues

1. Generating 'hard' lattice together with short basis (later)

Heuristic TDF & Signature Scheme [GGH'96]

- Key idea: $pk =$ 'bad' basis $\mathbf{B}$ for $\mathcal{L}$, $sk =$ 'short' trapdoor basis $\mathbf{S}$
- Sign $H(\text{msg}) \in \mathbb{R}^n$ with "nearest-plane" algorithm [Babai'86]

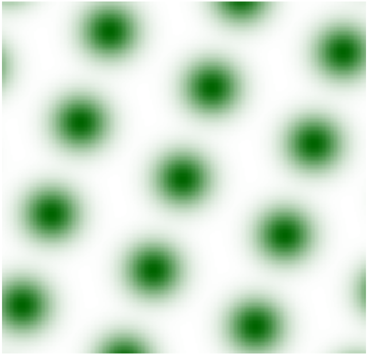


Technical Issues

1. Generating 'hard' lattice together with short basis (later)
2. Signing algorithm leaks secret basis!
   - ⋆ Total break after several signatures [NguyenRegev'06]

Blurring a Lattice

Blurring a Lattice



Blurring a Lattice

Blurring a Lattice

'Uniform' in $\mathbb{R}^n$ when std dev $\geq$ max length of some basis

Blurring a Lattice

Gaussian $\bmod \mathcal{L}$ is uniform when std dev $\geq$ max length of some basis

Blurring a Lattice

Gaussian $\bmod \mathcal{L}$ is uniform when std dev $\geq$ max length of some basis

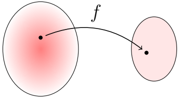
- First used in worst/average-case reductions [Regev'03,MR'04,...]

Blurring a Lattice

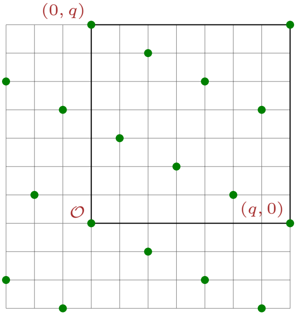
Gaussian $\bmod \mathcal{L}$ is uniform when std dev $\geq$ max length of some basis

- First used in worst/average-case reductions [Regev'03,MR'04,...]
- Now an essential ingredient in many crypto schemes [GPV'08,...]

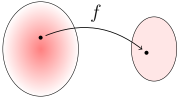
Preimage Sampleable TDF: Evaluation



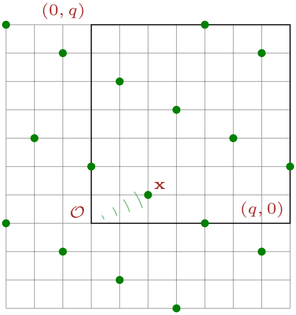
- 'Hard' description of $\mathcal{L}$ specifies f.
  Concretely: SIS matrix $\mathbf{A}$ defines $f_{\mathbf{A}}$.



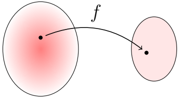
Preimage Sampleable TDF: Evaluation



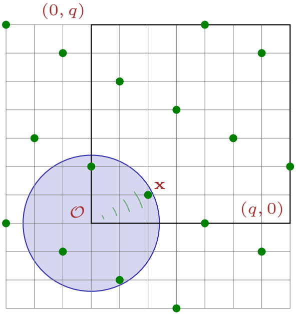
- 'Hard' description of $\mathcal{L}$ specifies f.
  Concretely: SIS matrix $\mathbf{A}$ defines $f_{\mathbf{A}}$.
- $f(\mathbf{x}) = \mathbf{x} \bmod \mathcal{L}$ for Gaussian $\mathbf{x}$.
  Concretely: $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} = \mathbf{u} \in \mathbb{Z}_q^n$.



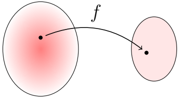
Preimage Sampleable TDF: Evaluation



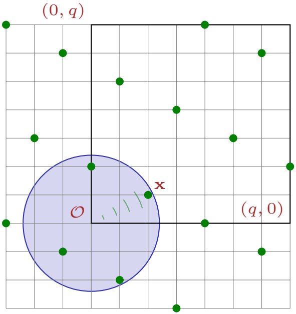
- ‘Hard’ description of $\mathcal{L}$ specifies f.
  Concretely: SIS matrix $\mathbf{A}$ defines $f_{\mathbf{A}}$.
- $f(\mathbf{x}) = \mathbf{x} \bmod \mathcal{L}$ for Gaussian $\mathbf{x}$.
  Concretely: $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} = \mathbf{u} \in \mathbb{Z}_q^n$.
- Inverting $\Leftrightarrow$ decoding syndrome $\mathbf{u}$
  $\Leftrightarrow$ solving SIS.



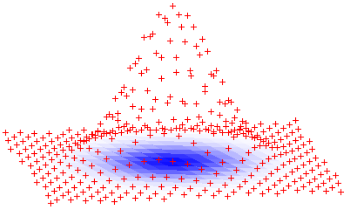
Preimage Sampleable TDF: Evaluation



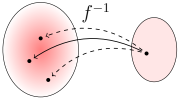
- 'Hard' description of $\mathcal{L}$ specifies f.
  Concretely: SIS matrix $\mathbf{A}$ defines $f_{\mathbf{A}}$.
- $f(\mathbf{x}) = \mathbf{x} \bmod \mathcal{L}$ for Gaussian $\mathbf{x}$.
  Concretely: $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} = \mathbf{u} \in \mathbb{Z}_q^n$.
- Inverting $\Leftrightarrow$ decoding syndrome $\mathbf{u}$
  $\Leftrightarrow$ solving SIS.



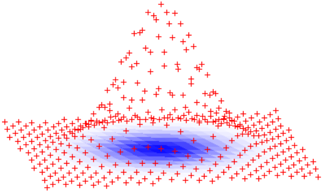
- Given $\mathbf{u}$, conditional distrib. of $\mathbf{x}$ is the discrete Gaussian $D_{\mathcal{L}_{\mathbf{u}}}$.



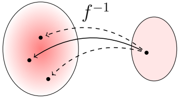
Preimage Sampling: Method #1



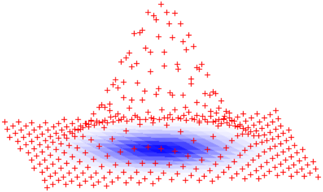
- Sample $D_{\mathcal{L}_\mathbf{u}}$ given any 'short enough' basis $\mathbf{S}$: $\max\|\mathbf{s}_i\| \leq$ std dev
  - ⋆ Unlike [GGH'96], output distribution leaks no information about $\mathbf{S}$!

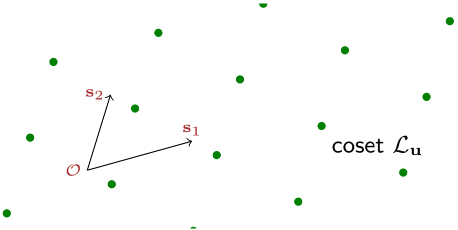


Preimage Sampling: Method #1

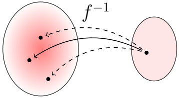


- Sample $D_{\mathcal{L}_\mathbf{u}}$ given any 'short enough' basis $\mathbf{S}$: $\max\|\mathbf{s}_i\| \le$ std dev
  - ⋆ Unlike [GGH'96], output distribution leaks no information about $\mathbf{S}$!
- "Nearest-plane" algorithm with randomized rounding [Klein'00,GPV'08]

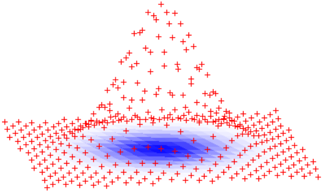


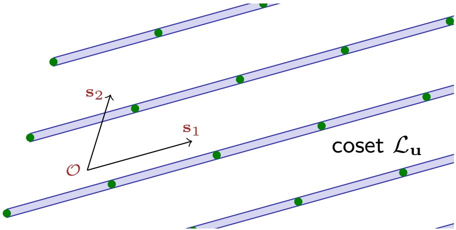


Preimage Sampling: Method #1

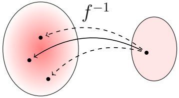


- Sample $D_{\mathcal{L}_\mathbf{u}}$ given any 'short enough' basis $\mathbf{S}$: $\max\|\mathbf{s}_i\| \leq$ std dev
  - ⋆ Unlike [GGH'96], output distribution leaks no information about $\mathbf{S}$!
- "Nearest-plane" algorithm with randomized rounding [Klein'00,GPV'08]



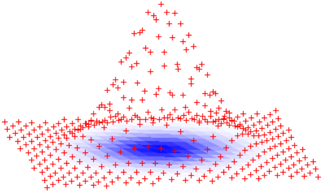


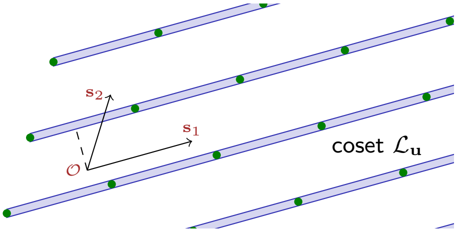
Preimage Sampling: Method #1



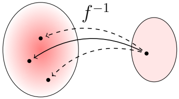
- Sample $D_{\mathcal{L}_\mathbf{u}}$ given any 'short enough' basis $\mathbf{S}$: $\max\|\mathbf{s}_i\| \leq$ std dev
  - ⋆ Unlike [GGH'96], output distribution leaks no information about $\mathbf{S}$!

- "Nearest-plane" algorithm with randomized rounding [Klein'00,GPV'08]



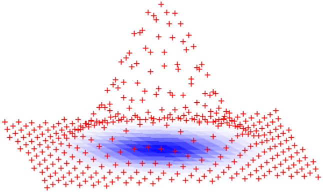


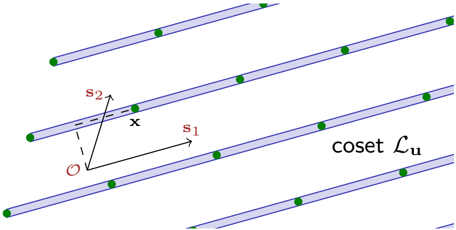
Preimage Sampling: Method #1



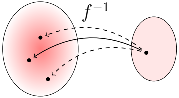
- Sample $D_{\mathcal{L}_{\mathbf{u}}}$ given any 'short enough' basis $\mathbf{S}$: $\max\|\mathbf{s}_i\| \leq$ std dev
  - ⋆ Unlike [GGH'96], output distribution leaks no information about $\mathbf{S}$!

- "Nearest-plane" algorithm with randomized rounding [Klein'00,GPV'08]

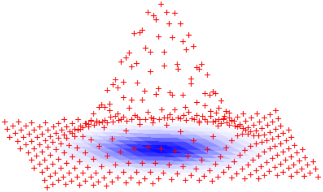


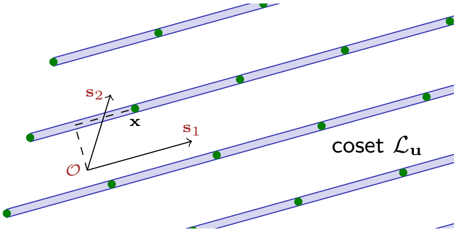


Preimage Sampling: Method #1



- Sample $D_{\mathcal{L}_{\mathbf{u}}}$ given any 'short enough' basis $\mathbf{S}$: $\max\|\mathbf{s}_i\| \leq$ std dev
  - ⋆ Unlike [GGH'96], output distribution leaks no information about $\mathbf{S}$!
- "Nearest-plane" algorithm with randomized rounding [Klein'00,GPV'08]





- Proof idea: $D_{\mathcal{L}_{\mathbf{u}}}$(plane) depends only on $\mathrm{dist}(0, \text{plane})$;
  not affected by shift within plane

Performance of Nearest-Plane Method?

Good News, and Bad News. . .

✔ Tight: std dev $\approx \max\|\tilde{\mathbf{s}}_i\|$ = max dist between adjacent planes

Performance of Nearest-Plane Method?

Good News, and Bad News...

- ✔ Tight: std dev $\approx \max \|\tilde{\mathbf{s}}_i\|$ = max dist between adjacent planes
- ✗ Not efficient: runtime $= \Omega(n^3)$, high-precision arithmetic

Performance of Nearest-Plane Method?

Good News, and Bad News...

- ✔ Tight: std dev $\approx \max \|\tilde{\mathbf{s}}_i\| =$ max dist between adjacent planes
- ✗ Not efficient: runtime $= \Omega(n^3)$, high-precision arithmetic
- ✗ Inherently sequential: n adaptive iterations

Performance of Nearest-Plane Method?

Good News, and Bad News...

- ✔ Tight: std dev $\approx \max\|\tilde{\mathbf{s}}_i\|$ = max dist between adjacent planes
- ✗ Not efficient: runtime $= \Omega(n^3)$, high-precision arithmetic
- ✗ Inherently sequential: n adaptive iterations
- ✗ No efficiency improvement in the ring setting [NTRU'98,M'02,...]

Performance of Nearest-Plane Method?

Good News, and Bad News...

- ✔ Tight: std dev $\approx \max\|\tilde{\mathbf{s}}_i\|$ = max dist between adjacent planes
- ✗ Not efficient: runtime $= \Omega(n^3)$, high-precision arithmetic
- ✗ Inherently sequential: n adaptive iterations
- ✗ No efficiency improvement in the ring setting [NTRU'98,M'02,...]

A Different Sampling Algorithm [P'10]

- Simple & efficient: n^2 online adds and mults (mod q)

Performance of Nearest-Plane Method?

Good News, and Bad News...

- ✔ Tight: std dev $\approx \max\|\tilde{\mathbf{s}}_i\|$ = max dist between adjacent planes
- ✗ Not efficient: runtime $= \Omega(n^3)$, high-precision arithmetic
- ✗ Inherently sequential: n adaptive iterations
- ✗ No efficiency improvement in the ring setting [NTRU'98,M'02,...]

A Different Sampling Algorithm [P'10]

- Simple & efficient: n^2 online adds and mults (mod q)

  Even better: $\tilde{O}(n)$ time in the ring setting

Performance of Nearest-Plane Method?

Good News, and Bad News...

- ✔ Tight: std dev $\approx \max\|\tilde{\mathbf{s}}_i\|$ = max dist between adjacent planes
- ✗ Not efficient: runtime $= \Omega(n^3)$, high-precision arithmetic
- ✗ Inherently sequential: n adaptive iterations
- ✗ No efficiency improvement in the ring setting [NTRU'98,M'02,...]

A Different Sampling Algorithm [P'10]

- Simple & efficient: n^2 online adds and mults (mod q)
  Even better: $\tilde{O}(n)$ time in the ring setting
- Fully parallel: n^2/P operations on any $P \leq n^2$ processors

Performance of Nearest-Plane Method?

Good News, and Bad News...

- ✔ Tight: std dev $\approx \max \|\tilde{\mathbf{s}}_i\|$ = max dist between adjacent planes
- ✗ Not efficient: runtime $= \Omega(n^3)$, high-precision arithmetic
- ✗ Inherently sequential: n adaptive iterations
- ✗ No efficiency improvement in the ring setting [NTRU'98,M'02,...]

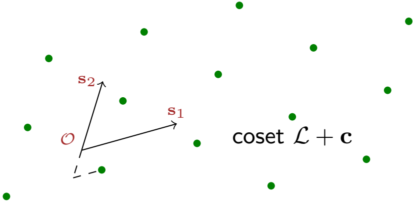
A Different Sampling Algorithm [P'10]

- Simple & efficient: n^2 online adds and mults (mod q)
  Even better: $\tilde{O}(n)$ time in the ring setting
- Fully parallel: n^2/P operations on any $P \leq n^2$ processors
- High quality: same* Gaussian std dev as nearest-plane alg

*in cryptographic applications

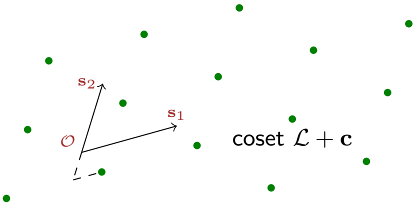
A First Attempt

- [Babai'86] 'simple rounding:' $\mathbf{c} \mapsto \mathbf{S} \cdot \mathsf{frac}(\mathbf{S}^{-1} \cdot \mathbf{c})$. (Fast & parallel!)



A First Attempt

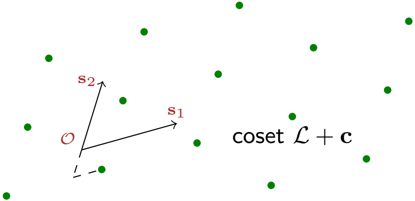
- [Babai'86] 'simple rounding:' $\mathbf{c} \mapsto \mathbf{S} \cdot \mathsf{frac}(\mathbf{S}^{-1} \cdot \mathbf{c})$. (Fast & parallel!)
- Deterministic rounding is <u>insecure</u> [NR'06] . . .



A First Attempt

- [Babai'86] 'simple rounding:' $\mathbf{c} \mapsto \mathbf{S} \cdot \mathsf{frac}(\mathbf{S}^{-1} \cdot \mathbf{c})_{\$}$. (Fast & parallel!)
- Deterministic rounding is <u>insecure</u> [NR'06] . . .

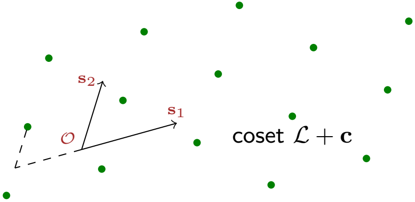
. . . but what about randomized rounding?

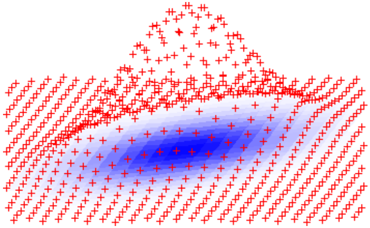


A First Attempt

- [Babai'86] 'simple rounding:' $\mathbf{c} \mapsto \mathbf{S} \cdot \text{frac}(\mathbf{S}^{-1} \cdot \mathbf{c})_{\$}$.   (Fast & parallel!)
- Deterministic rounding is <u>insecure</u> [NR'06] ...

  ... but what about randomized rounding?

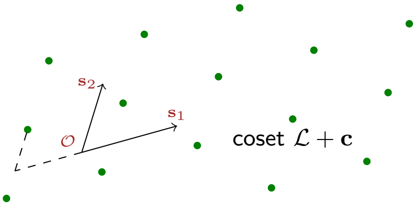


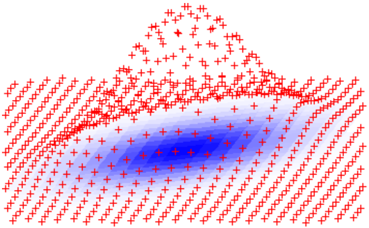


A First Attempt

- [Babai'86] 'simple rounding:' $\mathbf{c} \mapsto \mathbf{S} \cdot \mathrm{frac}(\mathbf{S}^{-1} \cdot \mathbf{c})_{\$}$. (Fast & parallel!)
- Deterministic rounding is insecure [NR'06] ...

  ... but what about randomized rounding?





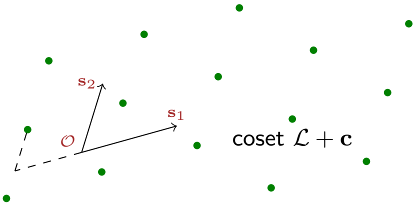
- Non-spherical discrete Gaussian: has covariance

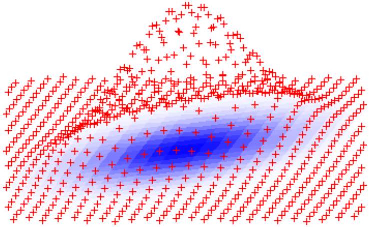
$$\Sigma := \mathbb{E}_{\mathbf{x}} \left[\mathbf{x} \cdot \mathbf{x}^t \right] \approx \mathbf{S} \cdot \mathbf{S}^t.$$

A First Attempt

- [Babai'86] 'simple rounding:' $\mathbf{c} \mapsto \mathbf{S} \cdot \mathsf{frac}(\mathbf{S}^{-1} \cdot \mathbf{c})_{\$}$. (Fast & parallel!)
- Deterministic rounding is <u>insecure</u> [NR'06] . . .

  . . . but what about randomized rounding?





- <u>Non-spherical</u> discrete Gaussian: has covariance

$$\Sigma := \mathbb{E}_{\mathbf{x}} \left[\mathbf{x} \cdot \mathbf{x}^t \right] \approx \mathbf{S} \cdot \mathbf{S}^t.$$

Covariance can be measured — and it leaks $\mathbf{S}$! (up to rotation)

Inspiration: Some Facts About Gaussians

1. Continuous Gaussian $\leftrightarrow$ positive definite covariance matrix Σ.

   (pos def means: $\mathbf{u}^t \, \Sigma \, \mathbf{u} > 0$ for all unit $\mathbf{u}$.)

Inspiration: Some Facts About Gaussians

1. Continuous Gaussian $\leftrightarrow$ positive definite covariance matrix Σ.

   (pos def means: $\mathbf{u}^t \, \Sigma \, \mathbf{u} > 0$ for all unit $\mathbf{u}$.)

   Spherical Gaussian $\leftrightarrow$ covariance $s^2 \, \mathbf{I}$.

Inspiration: Some Facts About Gaussians

1. Continuous Gaussian $\leftrightarrow$ positive definite covariance matrix Σ.

   (pos def means: $\mathbf{u}^t \, \Sigma \, \mathbf{u} > 0$ for all unit $\mathbf{u}$.)

   Spherical Gaussian $\leftrightarrow$ covariance $s^2 \, \mathbf{I}$.

2. Convolution of Gaussians:

$$\Sigma_1 \quad + \quad \Sigma_2 \quad = \quad \Sigma = s^2 \, \mathbf{I}$$

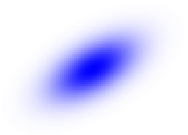
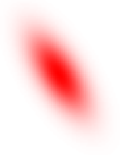
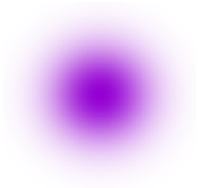
Inspiration: Some Facts About Gaussians

1. Continuous Gaussian $\leftrightarrow$ positive definite covariance matrix Σ.

   (pos def means: $\mathbf{u}^t \, \Sigma \, \mathbf{u} > 0$ for all unit $\mathbf{u}$.)

   Spherical Gaussian $\leftrightarrow$ covariance $s^2 \, \mathbf{I}$.

2. Convolution of Gaussians:

 $+$  $=$ 

$$\Sigma_1 \quad + \quad \Sigma_2 \quad = \quad \Sigma = s^2 \, \mathbf{I}$$

3. Given Σ_1, how small can s be? For $\Sigma_2 := s^2 \, \mathbf{I} - \Sigma_1$,

Inspiration: Some Facts About Gaussians

1. Continuous Gaussian $\leftrightarrow$ positive definite covariance matrix Σ.

   (pos def means: $\mathbf{u}^t \, \Sigma \, \mathbf{u} > 0$ for all unit $\mathbf{u}$.)

   Spherical Gaussian $\leftrightarrow$ covariance $s^2 \, \mathbf{I}$.

2. Convolution of Gaussians:

$$\Sigma_1 \quad + \quad \Sigma_2 \quad = \quad \Sigma = s^2 \, \mathbf{I}$$

3. Given Σ_1, how small can s be? For $\Sigma_2 := s^2 \, \mathbf{I} - \Sigma_1$,

$$\mathbf{u}^t \, \Sigma_2 \, \mathbf{u} \; = \; s^2 - \mathbf{u}^t \, \Sigma_1 \, \mathbf{u} > 0 \quad \Longleftrightarrow \quad \boxed{s^2 > \max \lambda_i(\Sigma_1)}$$

Inspiration: Some Facts About Gaussians

1. Continuous Gaussian $\leftrightarrow$ positive definite covariance matrix Σ.

   (pos def means: $\mathbf{u}^t \, \Sigma \, \mathbf{u} > 0$ for all unit $\mathbf{u}$.)

   Spherical Gaussian $\leftrightarrow$ covariance $s^2 \, \mathbf{I}$.

2. Convolution of Gaussians:

$$\Sigma_1 \quad + \quad \Sigma_2 \quad = \quad \Sigma = s^2 \, \mathbf{I}$$

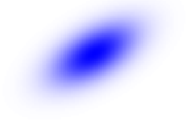
3. Given Σ_1, how small can s be? For $\Sigma_2 := s^2 \, \mathbf{I} - \Sigma_1$,

$$\mathbf{u}^t \, \Sigma_2 \, \mathbf{u} \; = \; s^2 - \mathbf{u}^t \, \Sigma_1 \, \mathbf{u} > 0 \quad \Longleftrightarrow \quad \boxed{s^2 > \max \lambda_i(\Sigma_1)}$$

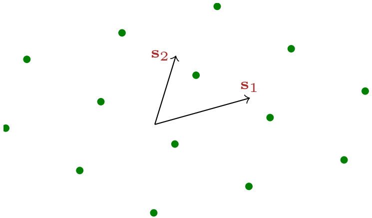
For $\Sigma_1 = \mathbf{S} \, \mathbf{S}^t$, can use any $\boxed{s > s_1(\mathbf{S}) := \text{max singular val of } \mathbf{S}.}$

‘Convolution’ Sampling Algorithm [P’10]

- Given basis $\mathbf{S}$, coset $\mathcal{L} + \mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,

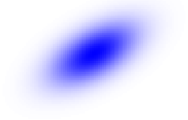


$$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$$

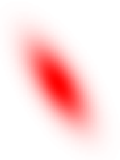


'Convolution' Sampling Algorithm [P'10]

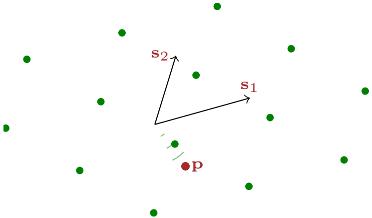
- Given basis $\mathbf{S}$, coset $\mathcal{L} + \mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,
  1. Generate perturbation $\mathbf{p}$ with covariance $\Sigma_2 := s^2\,\mathbf{I} - \Sigma_1 > 0$



$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$

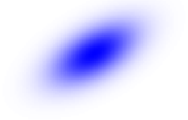


Σ_2

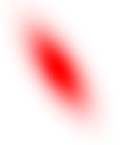


'Convolution' Sampling Algorithm [P'10]

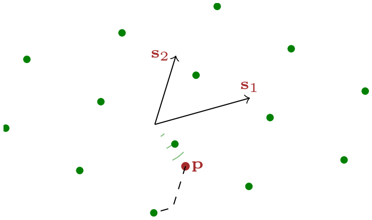
- Given basis $\mathbf{S}$, coset $\mathcal{L} + \mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,
  1. Generate perturbation $\mathbf{p}$ with covariance $\Sigma_2 := s^2\,\mathbf{I} - \Sigma_1 > 0$
  2. Randomly simple-round $\mathbf{p}$ to $\mathcal{L} + \mathbf{c}$



$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$

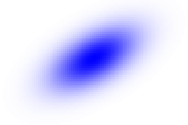


Σ_2

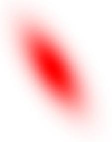


'Convolution' Sampling Algorithm [P'10]

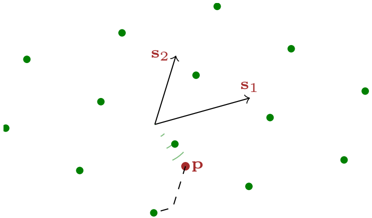
- Given basis $\mathbf{S}$, coset $\mathcal{L} + \mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,
  1. Generate perturbation $\mathbf{p}$ with covariance $\Sigma_2 := s^2 \mathbf{I} - \Sigma_1 > 0$
  2. Randomly simple-round $\mathbf{p}$ to $\mathcal{L} + \mathbf{c}$



$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$



Σ_2

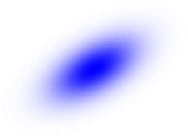


Convolution* Theorem

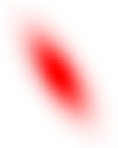
Algorithm generates a spherical discrete Gaussian over $\mathcal{L} + \mathbf{c}$.

‘Convolution’ Sampling Algorithm [P’10]

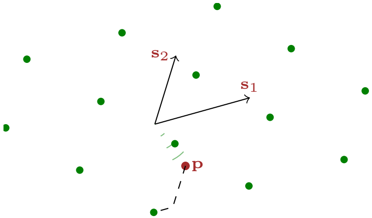
- Given basis $\mathbf{S}$, coset $\mathcal{L} + \mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,
  1. Generate perturbation $\mathbf{p}$ with covariance $\Sigma_2 := s^2\,\mathbf{I} - \Sigma_1 > 0$
  2. Randomly simple-round $\mathbf{p}$ to $\mathcal{L} + \mathbf{c}$



$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$



Σ_2



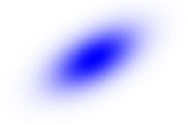
Convolution* Theorem

Algorithm generates a spherical discrete Gaussian over $\mathcal{L} + \mathbf{c}$.

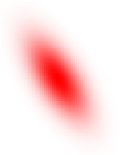
(*technically not a convolution, since step 2 depends on step 1.)

‘Convolution’ Sampling Algorithm [P’10]

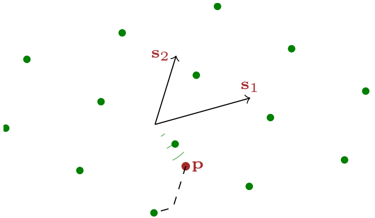
- Given basis $\mathbf{S}$, coset $\mathcal{L} + \mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,
  1. Generate perturbation $\mathbf{p}$ with covariance $\Sigma_2 := s^2\,\mathbf{I} - \Sigma_1 > 0$
  2. Randomly simple-round $\mathbf{p}$ to $\mathcal{L} + \mathbf{c}$



$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$



Σ_2

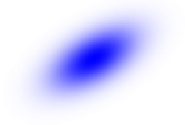


Optimizations

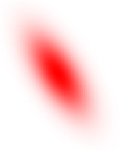
1. Precompute perturbations offline

‘Convolution’ Sampling Algorithm [P’10]

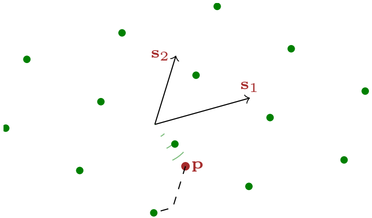
- Given basis $\mathbf{S}$, coset $\mathcal{L} + \mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,
  1. Generate perturbation $\mathbf{p}$ with covariance $\Sigma_2 := s^2\,\mathbf{I} - \Sigma_1 > 0$
  2. Randomly simple-round $\mathbf{p}$ to $\mathcal{L} + \mathbf{c}$



$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$



Σ_2

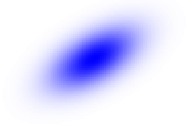


Optimizations

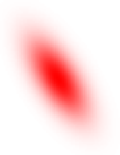
1. Precompute perturbations offline
2. Batch multi-sample using fast matrix multiplication

‘Convolution’ Sampling Algorithm [P’10]

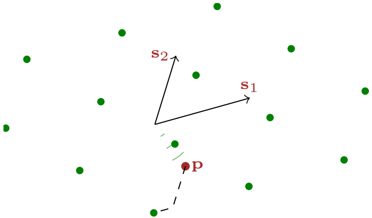
- Given basis $\mathbf{S}$, coset $\mathcal{L}+\mathbf{c}$, and std dev $s > s_1(\mathbf{S})$,
  1. Generate perturbation $\mathbf{p}$ with covariance $\Sigma_2 := s^2\,\mathbf{I} - \Sigma_1 > 0$
  2. Randomly simple-round $\mathbf{p}$ to $\mathcal{L}+\mathbf{c}$



$\Sigma_1 = \mathbf{S}\,\mathbf{S}^t$



Σ_2



Optimizations

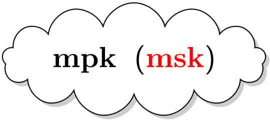
1. Precompute perturbations offline
2. Batch multi-sample using fast matrix multiplication
3. More tricks & simplifications for SIS lattices (next talk)

Part 2:

Identity-Based Encryption

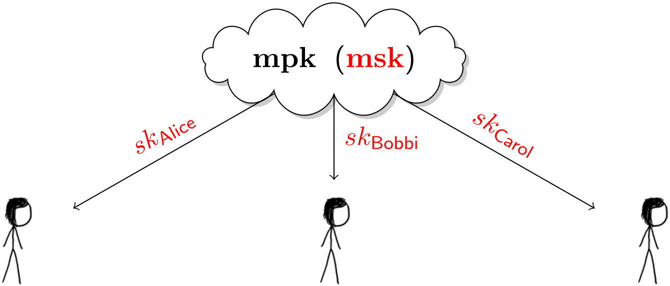
Identity-Based Encryption

- Proposed by [Shamir'84]: could this exist?



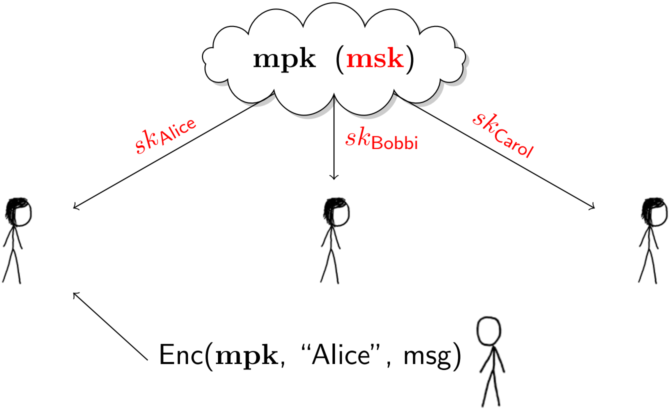
Identity-Based Encryption

- Proposed by [Shamir'84]: could this exist?



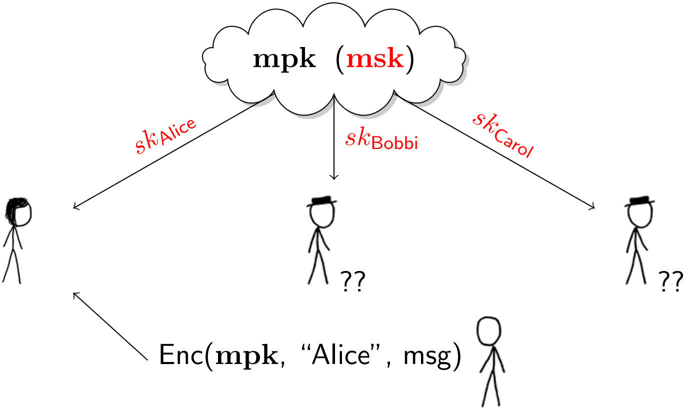
Identity-Based Encryption

- Proposed by [Shamir'84]: could this exist?



Identity-Based Encryption

- Proposed by [Shamir'84]: could this exist?



Fast-Forward 17 Years. . .

1. [BonehFranklin'01,. . .]: first IBE construction, using "new math"
(elliptic curves w/ bilinear pairings)

Fast-Forward 17 Years...

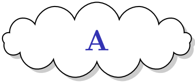
1. [BonehFranklin'01,...]: first IBE construction, using "new math"
   (elliptic curves w/ bilinear pairings)

2. [Cocks'01,BGH'07]: quadratic residuosity mod $N = pq$ [GM'82]

Fast-Forward 17 Years...

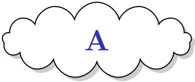
1. [BonehFranklin'01,...]: first IBE construction, using "new math" (elliptic curves w/ bilinear pairings)
2. [Cocks'01,BGH'07]: quadratic residuosity mod $N = pq$ [GM'82]
3. [GPV'08]: lattices!

Recall: 'Dual' LWE Cryptosystem



$\mathbf{x} \leftarrow$ Gauss

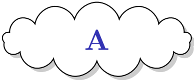
Recall: 'Dual' LWE Cryptosystem



$\mathbf{x} \leftarrow$ Gauss

$$\xrightarrow[\text{(public key)}]{\mathbf{u} = \mathbf{A}\mathbf{x} = f_{\mathbf{A}}(\mathbf{x})}$$

Recall: 'Dual' LWE Cryptosystem



$\mathbf{x} \leftarrow$ Gauss

$\mathbf{s}, \mathbf{e}$

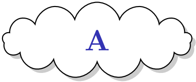
$$\mathbf{u} = \mathbf{A}\mathbf{x} = f_{\mathbf{A}}(\mathbf{x}) \longrightarrow$$

(public key)

$$\mathbf{b}^t = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t \longleftarrow$$

(ciphertext 'preamble')

Recall: ‘Dual’ LWE Cryptosystem



$\mathbf{x} \leftarrow$ Gauss

$\mathbf{s}, \mathbf{e}$

$$\mathbf{u} = \mathbf{A}\mathbf{x} = f_{\mathbf{A}}(\mathbf{x}) \longrightarrow$$

(public key)

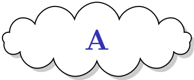
$$\longleftarrow \mathbf{b}^t = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t$$

(ciphertext ‘preamble’)

$$\longleftarrow b' = \mathbf{s}^t\,\mathbf{u} + e' + \mathsf{bit} \cdot \tfrac{q}{2}$$

(‘payload’)

Recall: 'Dual' LWE Cryptosystem



$\mathbf{x} \leftarrow$ Gauss

$\mathbf{s}, \mathbf{e}$

$$\mathbf{u} = \mathbf{A}\mathbf{x} = f_{\mathbf{A}}(\mathbf{x}) \longrightarrow$$

(public key)

$$\longleftarrow \mathbf{b}^t = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t$$

(ciphertext 'preamble')

$$\longleftarrow b' = \mathbf{s}^t\mathbf{u} + e' + \text{bit} \cdot \tfrac{q}{2}$$

('payload')

$$b' - \mathbf{b}^t\mathbf{x} \approx \text{bit} \cdot \tfrac{q}{2}$$

Recall: 'Dual' LWE Cryptosystem

$\mathbf{A}$

$\mathbf{x} \leftarrow$ Gauss

$\mathbf{s}, \mathbf{e}$

$$\mathbf{u} = \mathbf{A}\mathbf{x} = f_{\mathbf{A}}(\mathbf{x}) \longrightarrow$$

(public key)

$$\longleftarrow \mathbf{b}^t = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t$$

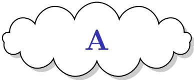
(ciphertext 'preamble')

$$b' - \mathbf{b}^t\mathbf{x} \approx \text{bit} \cdot \frac{q}{2} \qquad \longleftarrow b' = \mathbf{s}^t\mathbf{u} + e' + \text{bit} \cdot \frac{q}{2}$$

('payload')

? $(\mathbf{A}, \mathbf{u}, \mathbf{b}, b')$

Recall: 'Dual' LWE Cryptosystem



$\mathbf{x} \leftarrow$ Gauss

$\mathbf{s}, \mathbf{e}$

$$\mathbf{u} = \mathbf{A}\mathbf{x} = f_{\mathbf{A}}(\mathbf{x}) \longrightarrow$$

(public key)

$$\mathbf{b}^t = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t \longleftarrow$$

(ciphertext 'preamble')

$$b' = \mathbf{s}^t\mathbf{u} + e' + \text{bit} \cdot \tfrac{q}{2} \longleftarrow$$

('payload')

$$b' - \mathbf{b}^t\mathbf{x} \approx \text{bit} \cdot \tfrac{q}{2}$$

? $(\mathbf{A}, \mathbf{u}, \mathbf{b}, b')$

ID-Based Encryption

$\mathbf{x} \leftarrow f_{\mathbf{A}}^{-1}(\mathbf{u})$

$mpk = \mathbf{A}$

$\mathbf{s}, \mathbf{e}$

$\mathbf{u} = H(\text{“Alice”})$

(‘identity’ public key)

$\mathbf{b} = \mathbf{s}^t \mathbf{A} + \mathbf{e}^t$

(ciphertext preamble)

$b' - \mathbf{b}^t \mathbf{x} \approx \text{bit} \cdot \frac{q}{2}$

$b' = \mathbf{s}^t \mathbf{u} + e' + \text{bit} \cdot \frac{q}{2}$

(‘payload’)

When We Come Back. . .

- Generating trapdoors ($\mathbf{A}$ with short basis)

When We Come Back...

- Generating trapdoors ($\mathbf{A}$ with short basis)
- Removing the random oracle from signatures & IBE

When We Come Back...

- Generating trapdoors ($\mathbf{A}$ with short basis)
- Removing the random oracle from signatures & IBE
- More surprising applications

When We Come Back...

- Generating trapdoors ($\mathbf{A}$ with short basis)
- Removing the random oracle from signatures & IBE
- More surprising applications

Selected bibliography for this talk:

MR'04 D. Micciancio and O. Regev, "Worst-Case to Average-Case Reductions Based on Gaussian Measures," FOCS'04 / SICOMP'07.

GPV'08 C. Gentry, C. Peikert, V. Vaikuntanathan, "Trapdoors for Hard Lattices and New Cryptographic Constructions," STOC'08.

P'10 C. Peikert, "An Efficient and Parallel Gaussian Sampler for Lattices," Crypto'10.