GPV Signatures (Hash-then-Sign Signatures)

Advanced Topics in ~~Cybersecurity~~ Cryptography (7CCSMATC)

Martin R. Albrecht

Falcon (signature scheme)

Falcon is a post-quantum signature scheme selected by the NIST at the fourth round of the post-quantum standardisation process. It was designed by Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang.[1][2][3] It relies on the hash-and-sign technique over the Gentry, Peikert, and Vaikuntanathan framework[4] over NTRU lattices. The name *Falcon* is an acronym for *Fast Fourier lattice-based compact signatures over NTRU*.
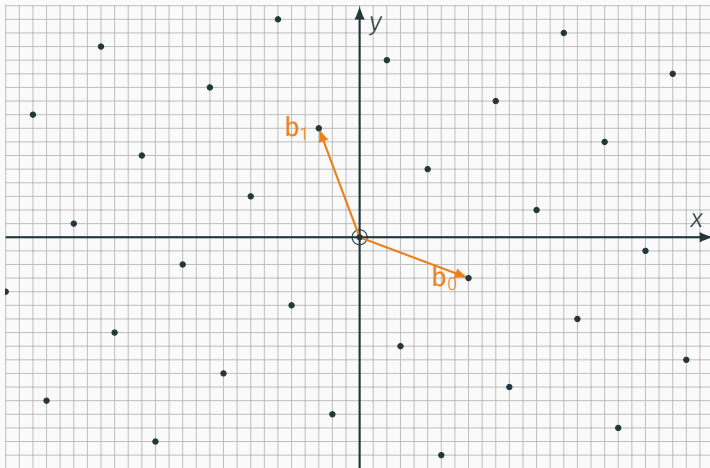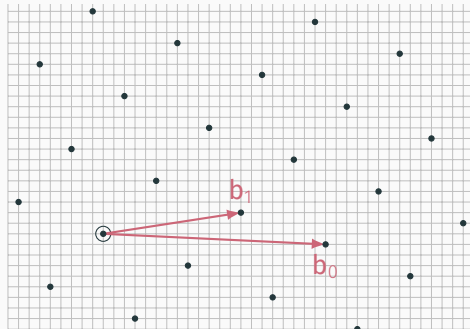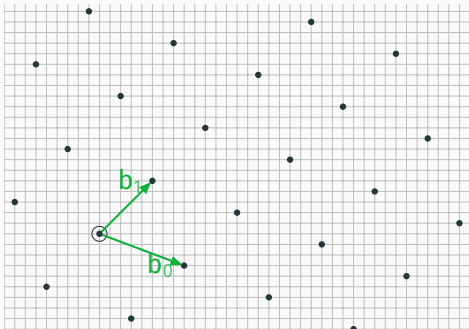
LATTICES

A lattice Λ is a discrete subgroup of $\mathbb{R}^d$.

Picture credit: Léo Ducas

Lattice Bases

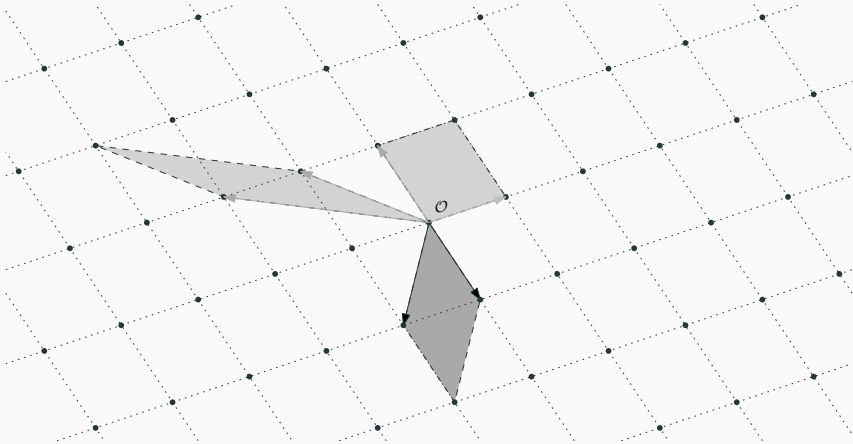|				
G	$\rightarrow$	B	: easy	(compute Hermite Normal form);
B	$\rightarrow$	G	: hard	(BKZ, lattice sieve ...).

Picture credit: Léo Ducas

The volume of a lattice is the volume of its fundamental domain.

For any two bases $\textcolor{green}{G}$, $\textcolor{red}{B}$ of the same lattice Λ:

$$\det(\textcolor{green}{G} \cdot \textcolor{green}{G}^T) = \det(\textcolor{red}{B} \cdot \textcolor{red}{B}^T).$$

We can therefore define:

$$\mathrm{Vol}(\Lambda) = \sqrt{\det(\textcolor{red}{B} \cdot \textcolor{red}{B}^T)}.$$

Picture credit: https://tex.stackexchange.com/a/42559

It will be useful to consider the lengths of the Gram–Schmidt vectors.

The vector $\mathbf{b}_i^*$ is the orthogonal projection of $\mathbf{b}_i$ to the space spanned by the vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.

Informally, this means taking out the contributions in the directions of previous vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.
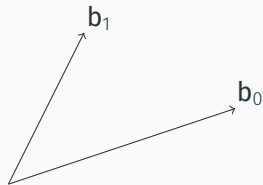
It will be useful to consider the lengths of the Gram–Schmidt vectors.

The vector $\mathbf{b}_i^*$ is the orthogonal projection of $\mathbf{b}_i$ to the space spanned by the vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.

Informally, this means taking out the contributions in the directions of previous vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.
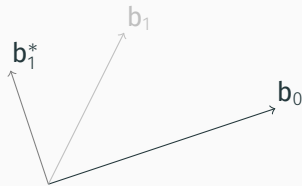
Let $B^\star$ be the Gram–Schmidt Orthogonalisation of B. The matrix $B^\star$ is **not** a basis for Λ, but

$$\mathsf{Vol}(\Lambda) = \sqrt{\det(B^\star \cdot B^{\star^T})} = \prod \|b_i^\star\| \,.$$

"Good" Bases

Recall that, independently of the basis B it hold that:

$$\mathrm{vol}(\Lambda) = \prod \|b_i^\star\|.$$

Therefore, it is somehow equivalent that

- $\max_i \|b_i^\star\|$ is small
- $\min_i \|b_i^\star\|$ is large
- $\kappa(B) = \min_i \|b_i^\star\| / \max_i \|b_i^\star\|$ is small

Good Basis

$$\kappa(G) = \mathrm{poly}(d), \qquad \forall i : \|g_i^\star\| = \mathrm{poly}(d) \cdot \mathsf{Vol}(\Lambda)^{1/d}.$$

LLL-reduced Basis

$$\kappa(G) \approx (1.04)^d, \qquad \max_i \|g_i^\star\| \approx (1.02)^d \cdot \mathsf{Vol}(\Lambda)^{1/d}.$$

```
A = IntegerMatrix.random(120, "qary", k=60, bits=20)[::-1]
M = GSO.Mat(A, update=True)
line([(i,log(r_, 2)/2) for i, r_ in enumerate(M.r())], **plot_kwds)
```
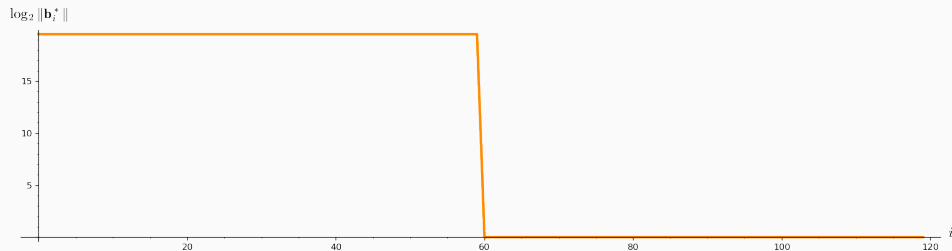
```
A = LLL.reduction(A)
M = GSO.Mat(A, update=True)
line([(i,log(r_, 2)/2) for i, r_ in enumerate(M.r())], **plot_kwds)
```
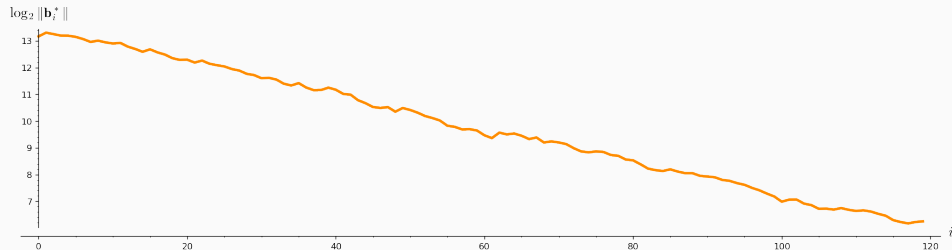
Bases and Fundamental Domains

Each basis defines a **tiling**.
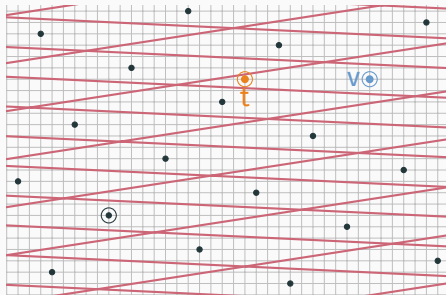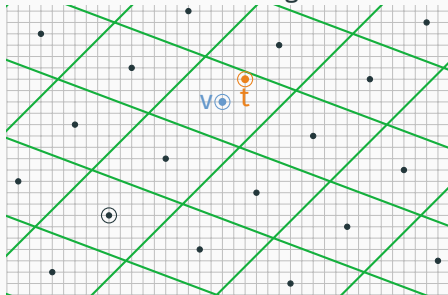
Round'off Algorithm [Lenstra, Babai]:

Given a target t, find $v \in \Lambda$ at the center the tile.

Round Off Algorithm [Lenstra,Babai]:
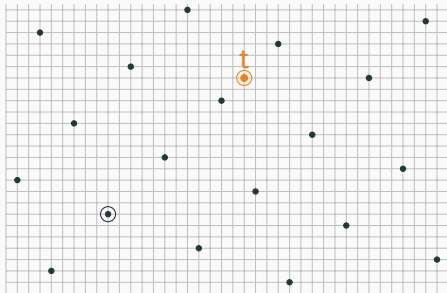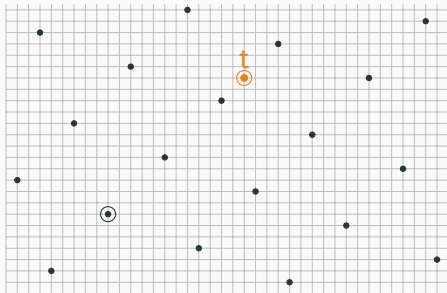
Credit: Léo Ducas

$\cdot B^{-1}$

$\longrightarrow$
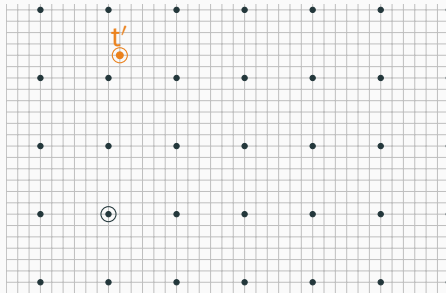
Round Off Algorithm [Lenstra,Babai]:

Credit: Léo Ducas

$\cdot$ Use B to switch to the lattice $\mathbb{Z}^d$ $t' = B^{-1} \cdot t$

Round Off Algorithm [Lenstra,Babai]:
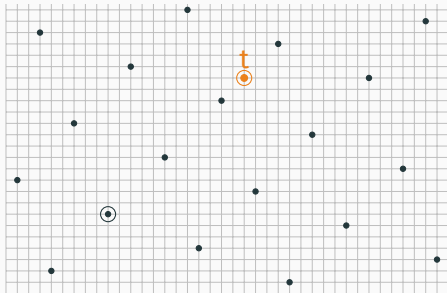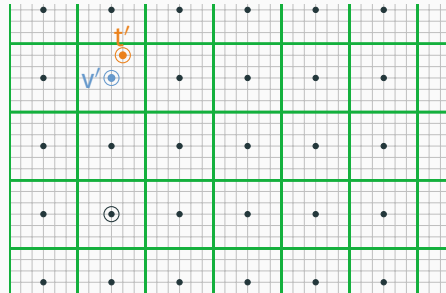
Credit: Léo Ducas

- Use B to switch to the lattice $\mathbb{Z}^d$ $\quad t' = B^{-1} \cdot t$
- Round each coordinate $\quad v' = \lfloor t' \rceil$

Round Off Algorithm [Lenstra,Babai]:
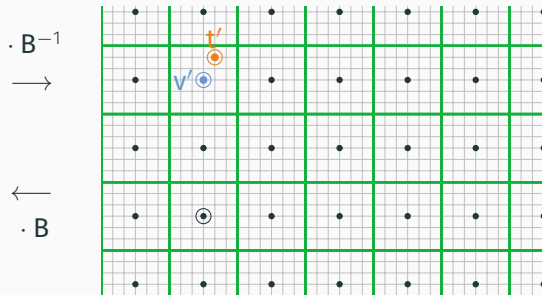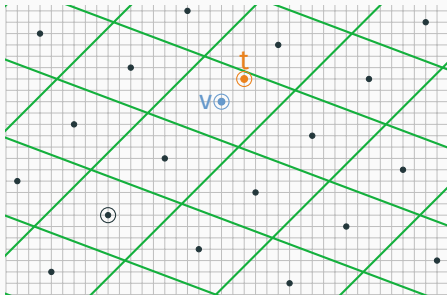
Credit: Léo Ducas

- Use B to switch to the lattice $\mathbb{Z}^d$ $\quad t' = B^{-1} \cdot t$
- Round each coordinate $\quad v' = \lfloor t' \rceil$
- Switch back to Λ $\quad v = B \cdot v'$

NEAREST-PLANE ALGORITHM

There is a better algorithm (Nearest Plane) based on Gram–Schmidt Orthogonalisation:
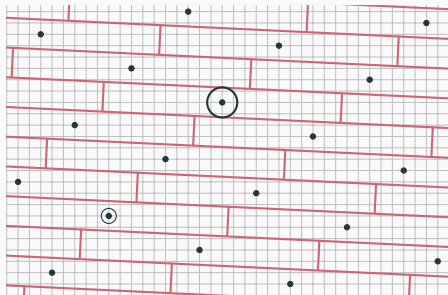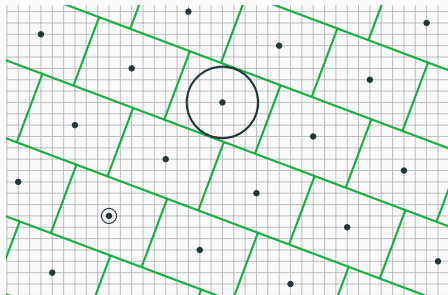
- Worst-case distance: $\frac{1}{2}\sqrt{\sum \left\| \mathbf{b}_i^\star \right\|^2}$ (Approx-CVP)
- Correct decoding of $\mathbf{t} = \mathbf{v} + \mathbf{e}$ for $\mathbf{v} \in \Lambda$ if $\|\mathbf{e}\| \leq \min \left\| \mathbf{b}_i^\star \right\|$ (BDD)

Credit: Léo Ducas

Good basis **G**: can solve Approx-CVP / BDD.   Bad basis **B**: solving CVP is **hard**.
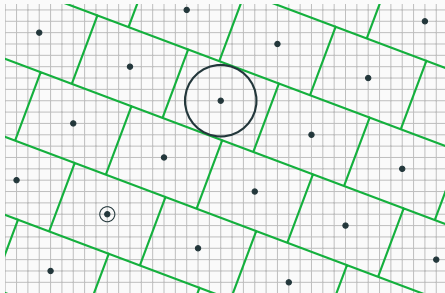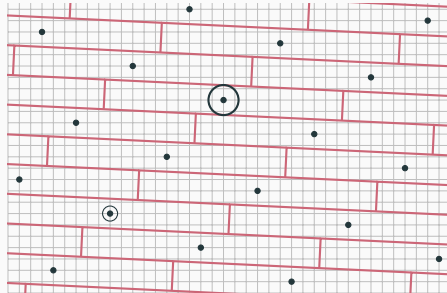
Can we use this as a trapdoor?

SIGNATURES

Sign:

1. Hash the message to a random vector **m**.

2. Apply Nearest Plane with a good basis **G**: find $s \in \Lambda$ close to **m**.
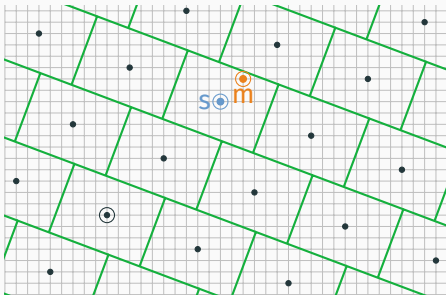
Credit: Léo Ducas

Verify:

1. Check that $s \in \Lambda$ using the bad basis **B**

2. Check that **m** is close to **s**.
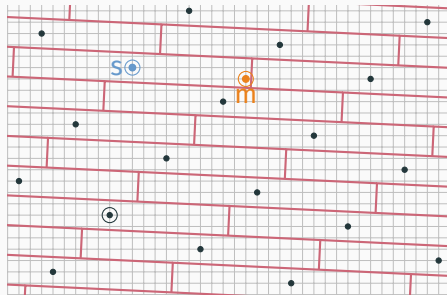
The difference $s - m$ is always inside the parallelepiped spanned by the good basis G or its Gram–Schmidt Orthogonalisation $G^\star$:
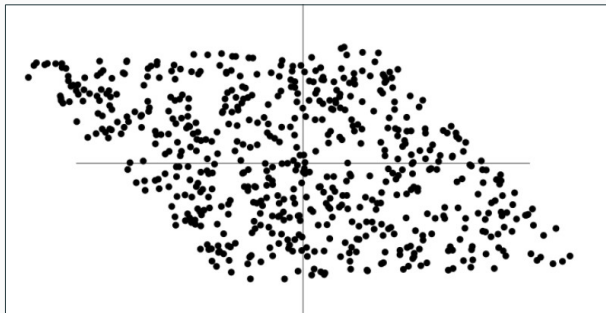
- Each signatures (s, m) leaks information about G.
- Learning a parallelepiped from few signatures [NR06]
- Total break of original GGH and NTRUSign schemes

The distribution $s - m$ can be made independent of G by randomising the above algorithms:

Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In: *40th ACM STOC*. ed. by Richard E. Ladner and Cynthia Dwork. ACM Press, May 2008, pp. 197–206. DOI: 10.1145/1374376.1374407

q-ary Lattices

- Let q be a prime integer[1] and $n < m$ two positive integers.
- The matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ spans the q-ary lattice:

$$\Lambda_q(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m \mid \exists \mathbf{y} \in \mathbb{Z}_q^n, \mathbf{x} \equiv \mathbf{A} \cdot \mathbf{y} \bmod q\} = \mathbf{A} \cdot \mathbb{Z}_q^n + q\mathbb{Z}^m$$

Lattice Parameters

Assuming $\mathbf{A}$ is full-rank:

- $\dim(\Lambda_q(\mathbf{A})) = m$
- $\mathsf{Vol}(\Lambda_q(\mathbf{A})) = q^{m-n}$

---

[1]for simplicity

Example

```
q = 7; n = 3; m = 7
A = random_matrix(GF(q), n, m)
B = A.lift().stack(q * identity_matrix(m)); B
```

```
[3 3 2 1 2 6 2]
[2 6 1 2 5   5]
[1 3 6   4 4 3]
[7           ]
[  7         ]
[    7       ]
[      7     ]
[        7   ]
[          7 ]
[            7]
```

```
B.echelon_form()[:B.rank()]
```

```
[1     3 4 5  ]
[  1     2 5 4]
[    1 3 6 2 2]
[      7      ]
[        7    ]
[          7  ]
[            7]
```

Row-Based

SageMath's convention is "row based", i.e. it considers combinations of rows not columns. The literature on lattices favours column-based notation, i.e. combinations of columns. You'll need to transpose the above picture to make it consistent with the slide before.

- Let q be a prime integer[2] and $n < m$ two positive integers.
- The matrix $A \in \mathbb{Z}_q^{n \times m}$ is the parity-check of the lattice:

$$\Lambda_q^\perp(A) := \{x \in \mathbb{Z}^m \mid A \cdot x \equiv 0 \bmod q\} = \ker(x \mapsto A \cdot x \bmod q)$$

Lattice parameters

Assuming A is full-rank:

- $\dim(\Lambda_q^\perp(A)) = m$
- $\mathrm{Vol}(\Lambda_q^\perp(A)) = q^n$

---

[2]for simplicity

Example

```
q = 7; n = 3; m = 7
A = random_matrix(GF(q), n, m)
K = A.right_kernel().basis_matrix()
B = K.lift().stack(q * identity_matrix(m)); B
```

```
[1     2 6 6]
[ 1     6   3]
[   1   4 1   ]
[     1 4 1 1]
[7           ]
[ 7         ]
[   7       ]
[     7     ]
[       7   ]
[         7 ]
[           7]
```

```
B.echelon_form()[:B.rank()]
```

```
[1     2 6 6]
[ 1     6   3]
[   1   4 1   ]
[     1 4 1 1]
[       7   ]
[         7 ]
[           7]
```

The Short Integer Solution Problem (SIS)

Definition (SIS Assumption)

Given a random matrix $A \in \mathbb{Z}_q^{n \times m}$, finding a small non-zero $x \in \mathbb{Z}^m$ such that $A \cdot x \equiv 0 \bmod q$ is hard.[3]

Lattice Formulation

Solving Approx-SVP in $\Lambda_q^\perp(A)$ is hard.

In [Ajt98], Ajtai established that if solving SIS is easy then solving Approx-SVP for any lattice is also easy.

---

[3]Also on a quantum computer!

Consider the function:

$$f_{\mathbf{A}} : \{0,1\}^m \to \mathbb{Z}_q^n, \qquad \mathbf{x} \mapsto \mathbf{A} \cdot \mathbf{x} \bmod q$$

SIS $\Rightarrow$ Collision Resistant Hashing and One-Way Function

- Finding collisions is as hard as SIS[4]   (take the difference)

Moreover, if $m \gg n \log q$:

- $f_{\mathbf{A}}$ is highly surjective   (many pre-images for any image)
- Finding pre-images is hard   (show it!)

---

[4]Collisions must exist when $m > n \log q$.

GPV Signatures

KeyGen generate a random (looking) matrix $\mathbf{A}$ together with a short basis td for the lattice $\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m \,|\, \mathbf{A} \cdot \mathbf{x} \equiv \mathbf{0} \bmod q\}$.

Sign Compute $H(m)$.
Find an arbitrary, not-necessarily short, preimage $\mathbf{u}$: $\mathbf{A} \cdot \mathbf{u} \equiv H(m) \bmod q$.
Use the trapdoor td to solve Approx-CVP for $\mathbf{u}$ on $\Lambda_q^\perp(\mathbf{A})$, call it $\mathbf{z}$. By construction $\mathbf{A} \cdot \mathbf{z} \equiv \mathbf{0} \bmod q$.
Output $\mathbf{y} := \mathbf{u} - \mathbf{z}$. By construction $\|\mathbf{y}\|$ is small.

Verify Check that $\mathbf{A} \cdot \mathbf{y} \equiv H(m) \bmod q$ and that $\|\mathbf{y}\|$ is small.

KeyGen(1^λ)

$\mathbf{A}, \mathrm{td} \leftarrow\!\!\$ \mathrm{TrapGen}(1^n, 1^m, q, \beta)$
return $\mathrm{vk} := \mathbf{A}, \mathrm{sk} := \mathrm{td}$

Sign(m, sk)

$\mathbf{r} \leftarrow\!\!\$ \{0,1\}^\lambda$
$\mathbf{y} \leftarrow\!\!\$ \mathrm{SampPre}(\mathrm{td}, H(m, \mathbf{r}), \beta')$
return $\sigma := (\mathbf{y}, \mathbf{r})$

Verify(vk, σ, m)

---

return $\|\mathbf{y}\| \overset{?}{\leq} \beta' \wedge H(m, \mathbf{r}) \overset{?}{\equiv} \mathbf{A} \cdot \mathbf{y}$

- $(\mathbf{A}, \mathrm{td}) \leftarrow \mathrm{TrapGen}(1^n, 1^m, q, \beta)$ takes dimensions $n, m \in \mathbb{N}$, a modulus $q \in \mathbb{N}$ and a norm bound $\beta \in \mathbb{R}$ and generates a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a trapdoor td.
  When $\ell > 2m \log q$, the distribution of $\mathbf{A}$ is within $\mathrm{negl}(\lambda)$ statistical distance to the uniform distribution on $\mathbb{Z}_q^{n \times m}$.

- $\mathbf{u} \leftarrow \mathrm{SampD}(1^n, 1^m, \beta')$ outputs an element in $\mathbf{u} \in \mathbb{Z}^m$ with norm bound $\beta' \geq \beta$.
  $\mathbf{v} := \mathbf{A} \cdot \mathbf{u} \bmod q$ *is within $\mathrm{negl}(\lambda)$ statistical distance to the uniform distribution on $\mathbb{Z}_q^n$.*

- $\mathbf{u} \leftarrow \mathrm{SampPre}(\mathrm{td}, \mathbf{v}, \beta')$ takes a trapdoor td, a vector $\mathbf{v} \in \mathbb{Z}_q^n$ and a norm bound $\beta' \geq \beta$ and samples $\mathbf{u} \in \mathbb{Z}^\ell$ satisfying $\mathbf{A} \cdot \mathbf{u} \equiv \mathbf{v} \bmod q$ and $\|\mathbf{u}\| \leq \beta'$.
  $\mathbf{u}$ *is within $\mathrm{negl}(\lambda)$ statistical distance to*
  $\mathbf{u} \leftarrow \mathrm{SampD}(1^n, 1^m, \mathcal{R}, \beta')$ *conditioned on $\mathbf{v} \equiv \mathbf{A} \cdot \mathbf{u} \bmod q$.*

Security Proof

EUF-CMA	$S(m)$	$H(m)$
$\mathcal{Q}, \mathcal{H} \leftarrow \emptyset,;$	$\sigma \leftarrow\!\!{\scriptstyle\$}\ \Sigma.\text{Sign}(\text{sk}, m)$	if $m \notin \mathcal{H}$ then
$\text{vk}, \text{sk} \leftarrow\!\!{\scriptstyle\$}\ \Sigma.\text{KeyGen}(1^\lambda);$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$	$\quad \mathcal{H}[m] \leftarrow\!\!{\scriptstyle\$}\ \{0, 1\}^\lambda$
$(m^\star, \sigma^\star) \leftarrow\!\!{\scriptstyle\$}\ \mathcal{A}^{\text{S,H}}(\text{vk});$	return σ	return $\mathcal{H}[m]$
$b_0 := (m^\star, \cdot) \notin \mathcal{Q}$		
$b_1 := \Sigma.\text{Verify}(\text{vk}, \sigma^\star, m^\star) = 1$		
return $b_0 \wedge b_1$		

$$\text{Adv}_{\mathcal{A}, \Sigma}^{\text{euf-cma}}(\lambda) := \Pr[\text{EUF-CMA}_{\Sigma}^{\mathcal{A}}(\lambda) \Rightarrow 1]$$

Game_0	$S(m)$	$H(m, r)$
$\mathcal{Q}, \mathcal{H} \leftarrow \emptyset, \emptyset;$	$r \leftarrow_\$ \{0, 1\}^\lambda$	if $(m, r) \notin \mathcal{H}$ then
$A, \text{td} \leftarrow_\$ \text{TrapGen}(\cdots)$	$y \leftarrow_\$ \text{SampPre}(\text{td}, H(m, r), \beta')$	$\quad \mathcal{H}[m, r] \leftarrow_\$ \mathbb{Z}_q^n$
$(m^\star, (y^\star, r^\star)) \leftarrow \mathcal{A}^{S,H}(\text{vk});$	$\sigma := (y, r)$	return $\mathcal{H}[m, r]$
$b_0 := (m^\star, \cdot) \notin \mathcal{Q}$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$	
$b_1 := \|y^\star\| \leq \beta'$	return σ	
$b_2 := H(m^\star, r^\star) \equiv A \cdot y^\star$		
return $b_0 \wedge b_1 \wedge b_2$		

Game$_1$	S(m)	H(m, r)
$\mathcal{Q}, \mathcal{H}, \mathcal{P} \leftarrow \emptyset, \emptyset, \emptyset;$	$r \leftarrow\!\!\$\ \{0, 1\}^\lambda$	if $(m, r) \notin \mathcal{H}$ then
$\mathbf{A}, \mathrm{td} \leftarrow\!\!\$\ \mathrm{TrapGen}(\cdots)$	$y \leftarrow\!\!\$\ \mathrm{SampPre}(\mathrm{td}, H(m, r), \beta')$	$y \leftarrow\!\!\$\ \mathrm{SampD}(\cdots, \beta')$
$(m^\star, (y^\star, r^\star)) \leftarrow \mathcal{A}^{\mathrm{S}, \mathrm{H}}(\mathrm{vk});$	$\sigma := (y, r)$	$t := \mathbf{A} \cdot y \bmod q$
$b_0 := (m^\star, \cdot) \notin \mathcal{Q}$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$	$\mathcal{H}[m, r] \leftarrow t$
$b_1 := \|y^\star\| \leq \beta'$	return σ	$\mathcal{P}[m, r] \leftarrow y$
$b_2 := H(m^\star, r^\star) \equiv \mathbf{A} \cdot y^\star$		return $\mathcal{H}[m, r]$
$b_0 \wedge b_1 \wedge b_2$		

By the Leftover Hash Lemma the distributions of Game$_0$ and Game$_1$ are statistically close.

Bookkeeping

Game$_2$	S(m)	H(m, r)
$\mathcal{Q}, \mathcal{H}, \mathcal{P} \leftarrow \emptyset, \emptyset, \emptyset$;	$\mathsf{r} \leftarrow\!\!\$\ \{0,1\}^\lambda$	if $(m, \mathsf{r}) \notin \mathcal{H}$ then
bad $\leftarrow$ false	if $(m, \mathsf{r}) \in \mathcal{H}$ then	$\mathsf{y} \leftarrow\!\!\$\ \mathsf{SampD}(\cdots, \beta')$
$\mathsf{A}, \mathsf{td} \leftarrow\!\!\$\ \mathsf{TrapGen}(\cdots)$	bad $\leftarrow$ true	$\mathsf{t} := \mathsf{A} \cdot \mathsf{y} \bmod q$
$(m^\star, (\mathsf{y}^\star, \mathsf{r}^\star)) \leftarrow \mathcal{A}^{\mathsf{S,H}}(\mathsf{vk})$;	$\mathsf{y} \leftarrow\!\!\$\ \mathsf{SampPre}(\mathsf{td}, H(m, \mathsf{r}), \beta')$	$\mathcal{H}[m, \mathsf{r}] \leftarrow \mathsf{t}$
$b_0 := (m^\star, \cdot) \notin \mathcal{Q}$	$\sigma := (\mathsf{y}, \mathsf{r})$	$\mathcal{P}[m, \mathsf{r}] \leftarrow \mathsf{y}$
$b_1 := \|\mathsf{y}^\star\| \leq \beta'$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$	return $\mathcal{H}[m, \mathsf{r}]$
$b_2 := H(m^\star, \mathsf{r}^\star) \equiv \mathsf{A} \cdot \mathsf{y}^\star$	return σ	
$b_0 \wedge b_1 \wedge b_2$		

No change in behaviour.

Avoid Collisions

Game_3	$S(m)$	$H(m, r)$
$\mathcal{Q}, \mathcal{H}, \mathcal{P} \leftarrow \emptyset, \emptyset, \emptyset;$	$r \leftarrow\!\!\$\ \{0,1\}^\lambda$	if $(m, r) \notin \mathcal{H}$ then
bad $\leftarrow$ false	if $(m, r) \in \mathcal{H}$ then	$\quad y \leftarrow\!\!\$\ \mathsf{SampD}(\cdots, \beta')$
$A, \text{td} \leftarrow\!\!\$\ \mathsf{TrapGen}(\cdots)$	$\quad$ bad $\leftarrow$ true	$\quad t := A \cdot y \bmod q$
$(m^\star, (y^\star, r^\star)) \leftarrow \mathcal{A}^{S,H}(\text{vk});$	$\quad$ abort	$\quad \mathcal{H}[m, r] := t$
$b_0 := (m^\star, \cdot) \notin \mathcal{Q}$	$y \leftarrow\!\!\$\ \mathsf{SampPre}(\text{td}, H(m, r), \beta')$	$\quad \mathcal{P}[m, r] := y$
$b_1 := \|y^\star\| \leq \beta'$	$\sigma := (y, r)$	return $\mathcal{H}[m, r]$
$b_2 := H(m^\star, r^\star) \equiv A \cdot y^\star$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$	
$b_0 \wedge b_1 \wedge b_2$	return σ	

Apply Fundamental Lemma of Game Playing with $\Pr[\text{bad}] \approx |\mathcal{Q}|/2^{\lambda/2}$.

Game$_4$	S(m)	H(m, r)
$\mathcal{Q}, \mathcal{H}, \mathcal{P} \leftarrow \emptyset, \emptyset, \emptyset;$	$r \leftarrow\!\!\$ \{0, 1\}^\lambda$	if $(m, r) \notin \mathcal{H}$ then
$\text{bad} \leftarrow \text{false}$	if $(m, r) \in \mathcal{H}$ then	$\quad y \leftarrow\!\!\$ \text{SampD}(\cdots, \beta')$
$A, \text{td} \leftarrow\!\!\$ \text{TrapGen}(\cdots)$	$\quad \text{bad} \leftarrow \text{true}$	$\quad t := A \cdot y \bmod q$
$(m^\star, (y^\star, r^\star)) \leftarrow \mathcal{A}^{S,H}(vk);$	$\quad \text{abort}$	$\quad \mathcal{H}[m, r] \leftarrow t$
$b_0 := (m^\star, \cdot) \notin \mathcal{Q}$	$H(m, r)$	$\quad \mathcal{P}[m, r] \leftarrow y$
$b_1 := \|y^\star\| \leq \beta'$	$y \leftarrow \mathcal{P}[m, r]$	return $\mathcal{H}[m, r]$
$b_2 := H(m^\star, r^\star) \equiv A \cdot y^\star$	$\sigma := (y, r)$	
$b_0 \wedge b_1 \wedge b_2$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$	
	return σ	

Use SampPre $\approx$ SampD.

Throw Away Trapdoor

Game$_5$	S(m)	H(m, r)
$\mathcal{Q}, \mathcal{H}, \mathcal{P} \leftarrow \emptyset, \emptyset, \emptyset;$	r $\leftarrow\$ \{0, 1\}^\lambda$	if $(m, r) \notin \mathcal{H}$ then
bad $\leftarrow$ false	if $(m, r) \in \mathcal{H}$ then	$\quad$ y $\leftarrow\$ \mathsf{SampD}(\cdots, \beta')$
$\mathsf{A} \leftarrow\$ \mathbb{Z}_q^{n \times 2\,n\lceil \log q \rceil}$	$\quad$ bad $\leftarrow$ true	$\quad$ t $:= \mathsf{A} \cdot$ y mod q
$(m^\star, (y^\star, r^\star)) \leftarrow \mathcal{A}^{\mathsf{S,H}}(\mathsf{vk});$	$\quad$ abort	$\quad \mathcal{H}[m, r] \leftarrow$ t
$b_0 := (m^\star, \cdot) \notin \mathcal{Q}$	$H(m, r)$	$\quad \mathcal{P}[m, r] \leftarrow$ y
$b_1 := \|y^\star\| \leq \beta'$	y $\leftarrow \mathcal{P}[m, r]$	return $\mathcal{H}[m, r]$
$b_2 := H(m^\star, r^\star) \equiv \mathsf{A} \cdot y^\star$	$\sigma := (y, r)$	
$b_0 \wedge b_1 \wedge b_2$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$	
	return σ	

Use that TrapGen produces pseudorandom A.

We turn an adversary $\mathcal{A}$ that wins Game_5 into an adversary that breaks SIS for $\mathbf{A}$.

- The adversary outputs $(m^\star, (\mathbf{y}^\star, \mathbf{r}^\star))$ s.t. $H(m^\star, \mathbf{r}^\star) \equiv \mathbf{A} \cdot \mathbf{y}^\star$ and $\|\mathbf{y}^\star\| \leq \beta'$.
- If $\mathcal{A}$ has not called $H(m^\star, \mathbf{r}^\star)$ before producing its forgery, $\mathbf{t} := H(m^\star, \mathbf{r}^\star)$ is unknown and (close to) uniformly random: it succeeds with probability $1/q^n < 2^\lambda$.
- If $\mathcal{A}$ has called $H(m^\star, \mathbf{r}^\star)$ then $\mathbf{y} \leftarrow \mathcal{P}[m^\star, \mathbf{r}^\star]$ with $\|\mathbf{y}\| \leq \beta'$ and with high probability $\mathbf{y} \neq \mathbf{y}'$.

$$H(m^\star, \mathbf{r}^\star) \equiv \mathbf{A} \cdot \mathbf{y}^\star \equiv \mathbf{A} \cdot \mathbf{y} \quad \Rightarrow \quad \mathbf{0} \equiv \mathbf{A} \cdot (\mathbf{y}^\star - \mathbf{y}) \text{ and } \|\mathbf{y}^\star - \mathbf{y}\| \leq 2\beta'$$

The Role of r

If the signing algorithm did not sample a fresh r for each signature, then

- $Game_3$ would output new, fresh y on each call, but
- $Game_4$ would output the same y on each call.

which is easy to distinguish by just calling the signing algorithm twice.

Corresponding Attack

Without r, when $\mathcal{A}$ calls the signing oracle twice on the same m, it would get two different preimages y_0, y_1 for $H(m)$:

$$H(m) \equiv A \cdot y_0 \equiv A \cdot y_1 \quad \Rightarrow \quad 0 \equiv A \cdot (y_0 - y_1) \text{ and } \|y_0 - y_1\| \le 2\beta'.$$

- This solves SIS, which is supposed to be hard for our security proof to work.

The Role of r

If the signing algorithm did not sample a fresh r for each signature, then

- Game_3 would output new, fresh y on each call, but
- Game_4 would output the same y on each call.

which is easy to distinguish by just calling the signing algorithm twice.

Corresponding Attack

Without r, when $\mathcal{A}$ calls the signing oracle twice on the same m, it would get two different preimages y_0, y_1 for $H(m)$:

$$H(m) \equiv A \cdot y_0 \equiv A \cdot y_1 \quad \Rightarrow \quad 0 \equiv A \cdot (y_0 - y_1) \text{ and } \|y_0 - y_1\| \leq 2\,\beta'.$$

- This solves SIS, which is supposed to be hard for our security proof to work.
- This gives a short vector in $\Lambda_q^\perp(A)$, such short vectors make up our trapdoor!

In the Random Oracle Model we can choose what the hash function outputs.

[Ajt98]   Miklós Ajtai. The Shortest Vector Problem in L2 is NP-hard for Randomized
            Reductions (Extended Abstract). In: *30th ACM STOC*. ACM Press, May 1998,
            pp. 10–19. DOI: 10.1145/276698.276705.

[GPV08]   Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard
            lattices and new cryptographic constructions. In: *40th ACM STOC*. Ed. by
            Richard E. Ladner and Cynthia Dwork. ACM Press, May 2008, pp. 197–206. DOI:
            10.1145/1374376.1374407.

[NR06]   Phong Q. Nguyen and Oded Regev. Learning a Parallelepiped: Cryptanalysis of
            GGH and NTRU Signatures. In: *EUROCRYPT 2006*. Ed. by Serge Vaudenay. Vol. 4004.
            LNCS. Springer, Berlin, Heidelberg, 2006, pp. 271–288. DOI: 10.1007/11761679_17.