Digital signatures from lattice trapdoors: We can use lattice trapdoors to obtain a digital signature scheme in the random oracle model (this is essentially an analog of RSA signatures):

- KeyGen: $(A, td_A) \leftarrow \text{TrapGen}(n, m, q, \beta)$ [lattice parameters n, m, q, β are based on security parameter λ]
  Output $vk = A$ and $sk = td_A$
- Sign (sk, m): Output $\sigma \leftarrow f_A^{-1}(td_A, H(m))$. Here, $H: \{0,1\}^* \to \mathbb{Z}_q^n$ is modeled as a random oracle.
- Verify (vk, m, σ): Check that $\|\sigma\| \leq \beta$ and that $f_A(\sigma) = H(m)$.

Consider instantiation with gadget trapdoors:

- verification key: $A \in \mathbb{Z}_q^{n \times m}$
  signing key: $R \in \{0,1\}^{m \times m}$ such that $AR = G$
- signature on m: $y \leftarrow H(m) \in \mathbb{Z}_q^n$
  output $\sigma = v = R \cdot G^{-1}(y)$
- verification: check that
  $$A \cdot v = ARG^{-1}(y) = G \cdot G^{-1}(y) = y$$
  and v is short

Rationale for security:

- To forge a signature on m, adversary has to find v such that $Av = H(m)$
- Matrix A is statistically close to uniform and v is uniform, so this corresponds to solving the ISIS problem

Problem: Signing queries leak information about R. Adversary can compute $H(m) = y$ and $G^{-1}(y)$, so signing becomes a linear function!

Early approach of Goldreich-Goldwasser-Halevi was insecure — explicit key-recovery attack by Nguyen, Regev

In the context of the security proof, simulator needs a way to answer signing queries (without a trapdoor for A).

Requirement: Randomize the signing algorithm to hide trapdoor R

Approach: Instead of outputting a fixed (deterministic) preimage, use the trapdoor to sample a preimage where $Av = H(m)$.
↳ Can show that sample does not leak the trapdoor used [see Gentry-Peikert-Vaikuntanathan]

This is the basis of the Falcon post-quantum signature scheme standardized by NIST.

For key aggrement / public-key encryption, we turn to a dual problem: learning with errors

Learning with Errors (LWE): The LWE problem is defined with respect to lattice parameters n, m, q, χ, where χ is an error distribution over $\mathbb{Z}_q$ (oftentimes, this is a discrete Gaussian distribution over $\mathbb{Z}_q$). The $\mathsf{LWE}_{n,m,q,\chi}$ assumption states that for a random choice $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $s \xleftarrow{R} \mathbb{Z}_q^n$, $e \leftarrow \chi^m$, the following two distributions are computationally indistinguishable:

$$(A, s^T A + e^T) \overset{c}{\approx} (A, r)$$

where $r \xleftarrow{R} \mathbb{Z}_q^m$.

Observe that LWE implies SIS. Namely, an algorithm for SIS can be used to break LWE:

1. On input an LWE challenge (A, b^T), use the SIS solver to obtain a low-norm $x \in \mathbb{Z}_q^m$ where $Ax = 0$.
2. Output 1 if $|b^T x|$ is small and 0 otherwise.

If $b^T = s^T A + e^T$, then $b^T x = s^T A x + e^T x = e^T x$, which is small.
If $b \xleftarrow{R} \mathbb{Z}_q^m$, then $b^T x$ is uniform over $\mathbb{Z}_q$ (since $x \neq 0$), so $|b^T x|$ will not be small.

Symmetric encryption from LWE (for binary-valued messages) [Regev]

Setup(1^λ): Sample $s \xleftarrow{R} \mathbb{Z}_q^n$.

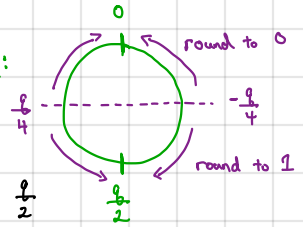
Encrypt$(s, \mu \in \{0,1\})$: Sample $a \xleftarrow{R} \mathbb{Z}_q^n$ and $e \leftarrow \chi$. Output $\left(a, s^T a + e + \mu \cdot \lfloor \frac{q}{2} \rfloor\right)$.

Decrypt(s, ct): Output $\underbrace{\lfloor ct_2 - s^T ct_1 \rceil_2}_{\text{"rounding operation"}}$

$$\lfloor x \rceil_2 = \begin{cases} 0 & \text{if } -\frac{q}{4} \leq x < \frac{q}{4} \\ 1 & \text{otherwise} \end{cases}$$

↑ take $x \in \mathbb{Z}_q$ to be representative between $-\frac{q}{2}$ and $\frac{q}{2}$

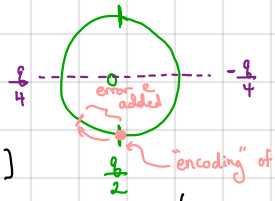
Visually:



Correctness: $ct_2 - s^T ct_1 = s^T a + e + \mu \cdot \lfloor \frac{q}{2} \rfloor - s^T a$

$= \mu \cdot \lfloor \frac{q}{2} \rfloor + e$

if $|e| < \frac{q}{4}$, then decryption recovers the correct bit



(message encrypted in "most significant bits" of the ciphertext)
↳ will see variant in HW5

Security: By the $\mathsf{LWE}_{n,m,q,\chi}$ assumption, $(a, s^T a + e) \overset{c}{\approx} (a, r)$ [m = 1]

where $r \xleftarrow{R} \mathbb{Z}_q$. Thus,

$$\left(a, s^T a + e + \mu \cdot \lfloor \tfrac{q}{2} \rfloor\right) \overset{c}{\approx} \left(a, r + \mu \cdot \lfloor \tfrac{q}{2} \rfloor\right)$$

↑ $r \xleftarrow{R} \mathbb{Z}_q$: one-time pad encryption of the message μ

Extends to K messages using $\mathsf{LWE}_{n,K,q,\chi}$ where i^{th} message encrypted using k^{th} column of A

Observe: this encryption scheme is additively homomorphic (over $\mathbb{Z}_2$):

$$\left(a_1, s^T a_1 + e_1 + \mu_1 \cdot \lfloor \tfrac{q}{2} \rfloor\right)$$
$$\left(a_2, s^T a_2 + e_2 + \mu_2 \cdot \lfloor \tfrac{q}{2} \rfloor\right)$$
$$\Rightarrow \left(a_1 + a_2,\ s^T (a_1 + a_2) + (e_1 + e_2) + (\mu_1 + \mu_2) \cdot \lfloor \tfrac{q}{2} \rfloor\right)$$

decryption then computes

$$(\mu_1 + \mu_2) \cdot \lfloor \tfrac{q}{2} \rfloor + e_1 + e_2$$

which when rounded yields $\mu_1 + \mu_2 \pmod 2$ provided that $|e_1 + e_2 + 1| < \frac{q}{4}$

This will give a simple approach for constructing a public-key encryption scheme from LWE

Idea: We will include encryptions of 0 in the public key and refresh ciphertexts by taking a subset sum of encryptions of 0:

Regev's public-key encryption scheme

- **Setup**: $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $s \xleftarrow{R} \mathbb{Z}_q^n$, $e \leftarrow \chi^m$; $b^T \leftarrow s^T A + e^T$; output $pk = (A, b^T)$, $sk = s$
  - ↑ can be viewed as m encryptions of 0 under the symmetric scheme with secret key s
- **Encrypt**(pk, μ): sample $r \xleftarrow{R} \{0,1\}^m$; output $(Ar, b^T r + \mu \cdot \lfloor \frac{q}{2} \rfloor)$
- **Decrypt**(sk, ct): output $\lfloor ct_2 - s^T ct_1 \rceil_2$

More compact form: $pk = A = \begin{bmatrix} \bar{A} \\ s^T \bar{A} + e^T \end{bmatrix}$, $ct = C = Ar + \begin{bmatrix} 0^n \\ \mu \cdot \lfloor q/2 \rfloor \end{bmatrix}$

Decryption: $[-s^T \mid 1] \cdot C = e^T r + \mu \cdot \lfloor \frac{q}{2} \rfloor$ → then round

Correctness: $ct_2 - s^T ct_1 = b^T r + \mu \cdot \lfloor \frac{q}{2} \rfloor - s^T A r = s^T A r + e^T r + \mu \cdot \lfloor \frac{q}{2} \rfloor - s^T A r = \mu \cdot \lfloor \frac{q}{2} \rfloor + e^T r$

if $|e^T r| < \frac{q}{4}$, then decryption succeeds (since e is small and r is binary, $e^T r$ is not large: $|e^T r| < m \|e\| \|r\| = m \|e\|$)

Security: Under LWE assumption public key

$(A, s^T A + e^T) \approx (A, u)$ where $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $u \xleftarrow{R} \mathbb{Z}_q^m$

By the "leftover hash lemma" if we sample $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $u \xleftarrow{R} \mathbb{Z}_q^n$, $r \xleftarrow{R} \{0,1\}^m$ where $m > 2n \log q$

$(Ar, u^T r) \approx (v, w)$ where $v \xleftarrow{R} \mathbb{Z}_q^n$ and $w \xleftarrow{R} \mathbb{Z}_q$

$\Rightarrow b^T r$ in ciphertext functions as a one-time pad

typically χ is B-bounded distribution ($\Pr[|e| \le B : e \leftarrow \chi] = 1$)

Formally: We will show encryption of 0 is computationally indistinguishable from encryption of 1

- **LWE** — Hyb_0: Challenger samples $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $s \xleftarrow{R} \mathbb{Z}_q^n$, $e \leftarrow \chi^m$ and computes $b^T = s^T A + e^T$. It sets $pk = (A, b)$. Then it samples $r \xleftarrow{R} \{0,1\}^m$ and computes $ct = (ct_0, ct_1) = (Ar, b^T r)$. It gives pk and ct to A.
- **LHL** — Hyb_1: Challenger samples $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ and $b \xleftarrow{R} \mathbb{Z}_q^m$.
- **LHL** — Hyb_2: Challenger samples $c_0 \xleftarrow{R} \mathbb{Z}_q^n$, $c_1 \xleftarrow{R} \mathbb{Z}_q$.
- **LWE** — Hyb_3: Challenger sets $c_0 = Ar$ and $c_1 = b^T r + \lfloor \frac{q}{2} \rfloor$.
- Hyb_4: Challenger sets $b^T = s^T A + e^T$.

Many generalizations:

- Encrypting elements mod p: $(Ar, b^T r + \lfloor \frac{q}{p} \rfloor \cdot \mu)$
  Correctness requires that $|e^T r| < \frac{q}{2p}$
- Encrypting vectors $\vec{\mu} \in \{0,1\}^k$.
  Sample secret key $S \xleftarrow{R} \mathbb{Z}_q^{n \times k}$ and $E \leftarrow \chi^{m \times k}$. Let $B^T = S^T A + E^T \in \mathbb{Z}_q^{k \times m}$
  Public key is $pk = (A, B)$
  Ciphertext is $ct = (Ar, B^T r + \lfloor \frac{q}{2} \rfloor \cdot \vec{\mu})$
  ($Ar \in \mathbb{Z}_q^n$, $B^T r + \lfloor \frac{q}{2} \rfloor \cdot \vec{\mu} \in \mathbb{Z}_q^k$)
  Observe: $|ct| = (n+k) \log q$ bits to encrypt k bits
  - ↳ Naively encrypting bit-by-bit would require $kn \log q$ bits.
  - ↳ Approach essentially is reusing the encryption randomness (e.g., same r used to encrypt each bit).