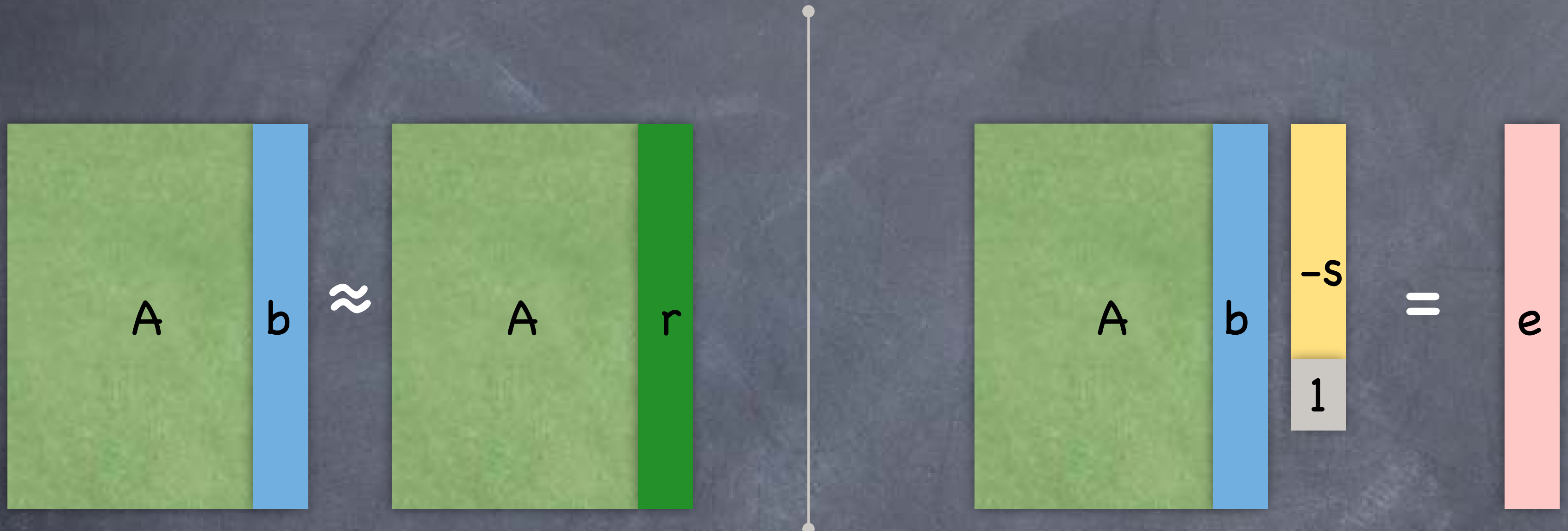
Fully Homomorphic Encryption

Lecture 21

Recall

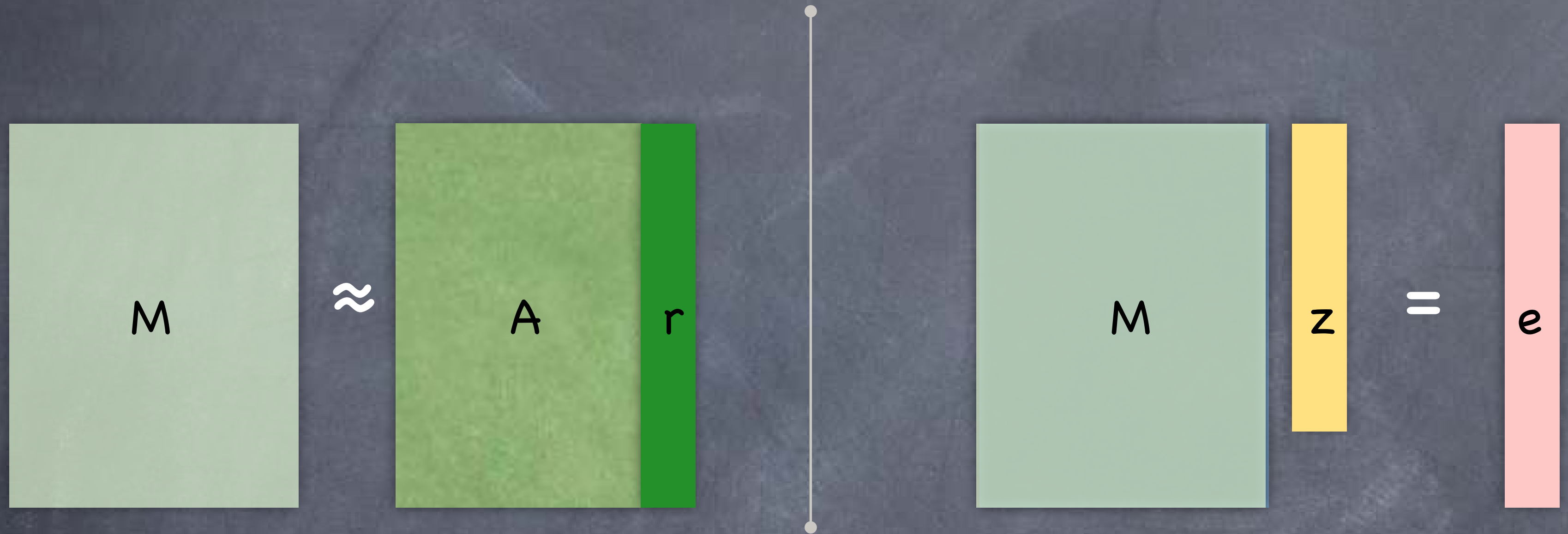
Learning With Errors



- LWE (decision version): $(A, A\underline{s}+\underline{e}) \approx (A, \underline{r})$, where A random matrix in $A \in \mathbb{Z}_q^{m\times n}$, $\underline{s}$ uniform, $\underline{e}$ has "small" entries from a Gaussian distribution, and $\underline{r}$ uniform.
- Average-case solution for LWE $\Rightarrow$ Worst-case solution for GapSVP (for appropriate choice of parameters)

Recall

Learning With Errors



- A pseudorandom matrix $M \in \mathbb{Z}_q^{m \times n'}$ and $\underline{\mathbf{z}} \in \mathbb{Z}_q^{n'}$ s.t. entries of $M\underline{\mathbf{z}}$ are all small

Gentry-Sahai-Waters

- Want to allow homomorphic operations on the ciphertext
- Rough plan: Ciphertext is a matrix. Addition and multiplication of messages by addition and multiplication of ciphertexts
- Recall from LWE: pseudorandom $M \in \mathbb{Z}_q^{m\times n}$ and random $\underline{\mathbf{z}} \in \mathbb{Z}_q^{n}$ s.t. $\underline{\mathbf{z}}^T M^T$ has small entries

$$z^T \quad M^T \quad = \quad e^T$$

- Public key $M \in \mathbb{Z}_q^{m\times n}$ and private key $\underline{\mathbf{z}}$
- $\mathrm{Enc}(\mu) = M^T R + \mu G$ where $R \leftarrow \{0,1\}^{m\times km}$ and $G \in \mathbb{Z}_q^{n\times km}$ the matrix to reverse bit-decomposition operation $B : \mathbb{Z}_q^{n\times d} \rightarrow \mathbb{Z}_q^{km\times d}$
- $\mathrm{Dec}_z(C) : \underline{\mathbf{z}}^T C = \underline{\delta}^T + \mu \underline{\mathbf{z}}^T G$ where $\underline{\delta}^T = e^T R$

Gentry-Sahai-Waters

- Supports messages $\mu \in \{0,1\}$ and NAND operations up to an a priori bounded depth of NANDs
- Public key $M \in \mathbb{Z}_q^{m\times n}$ and private key $\underline{\mathbf{z}}$ s.t. $\underline{\mathbf{z}}^T M$ has small entries
- $\mathrm{Enc}(\mu) = M^T R + \mu G$ where $R \leftarrow \{0,1\}^{m\times km}$ (and $G \in \mathbb{Z}_q^{n\times km}$ the matrix to reverse bit-decomposition)
- $\mathrm{Dec}_z(C)$: $\underline{\mathbf{z}}^T C = \underline{\delta}^T + \mu \underline{\mathbf{z}}^T G$ where $\underline{\delta}^T = e^T R$
- $\mathrm{NAND}(C_1,C_2)$: $G - C_1 \cdot B(C_2)$ (G is a (non-random) encryption of 1)
  - $\underline{\mathbf{z}}^T C_1 \cdot B(C_2) = \underline{\mathbf{z}}^T C_1 \cdot B(C_2) = (\underline{\delta}_1^T + \mu_1 \underline{\mathbf{z}}^T G)\, B(C_2)$

    $= \underline{\delta}_1^T B(C_2) + \mu_1 \underline{\mathbf{z}}^T C_2 = \underline{\delta}^T + \mu_1 \mu_2 \underline{\mathbf{z}}^T G$

    where $\underline{\delta}^T = \underline{\delta}_1^T B(C_2) + \mu_1 \underline{\delta}_2^T$ has small entries
  - In general, error gets multiplied by km. Allows depth $\approx \log_{km} q$

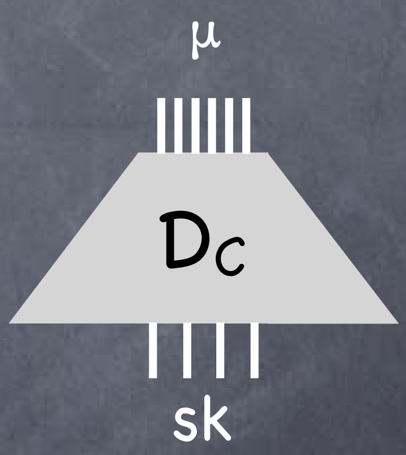
Only "left depth" counts, since $\underline{\delta} \leq k \cdot m \cdot \underline{\delta}_1 + \underline{\delta}_2$

Bootstrapping

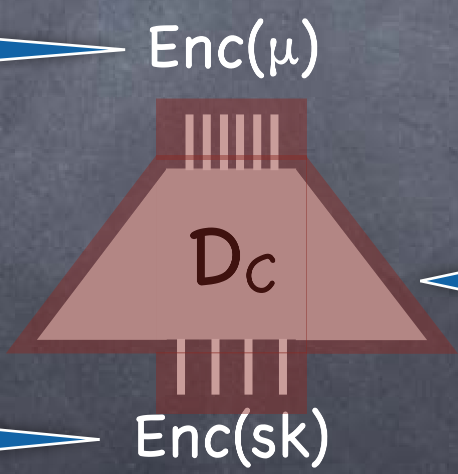
- Removing the need for an a priori bound
- Main idea: Can "refresh" the ciphertext to reduce noise
  - Refresh: homomorphically decrypt the given ciphertext under a fresh layer of encryption
    - cf. Degree reduction via share-switching: Homomorphically reconstruct under a fresh layer of sharing
    - But here, we have a secret-key (and there is only one party who knows the ciphertext fully)
  - Ciphertext is known, but secret-key should be kept encrypted
  - Consider decryption of a given ciphertext as a function applied to the secret-key: $D_C(sk) := Dec(C,sk)$

Bootstrapping

- Given a ciphertext C and hence the decryption function D_C s.t. $D_C(sk) := Dec(C,sk)$
- Also given: an encryption of sk (beware: circularity!)
- Goal: a fresh ciphertext with message $D_C(sk)$



Refreshed: Doesn't depend on how unfresh C was, but only on the depth of D_C

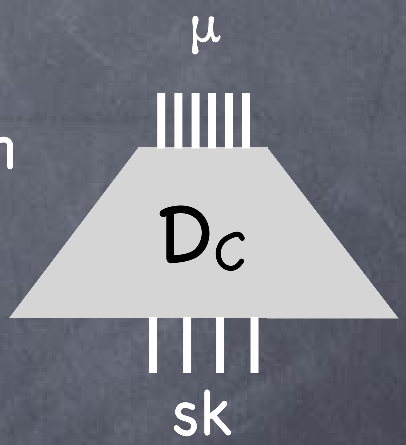


Homomorphic evaluation in the ciphertext space

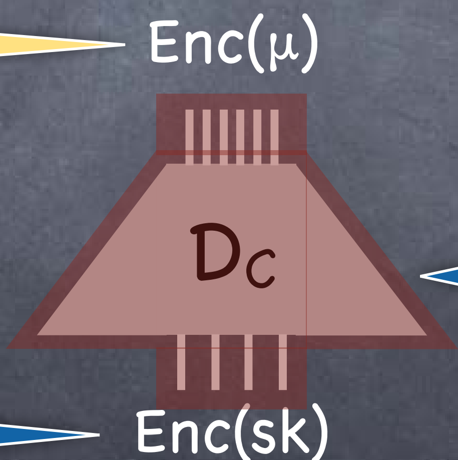
Fresh encryption of sk, provided along with the public key

Bootstrapping

- If depth of D_C s.t. $D_C(sk) := Dec(C,sk)$ is strictly less than the depth allowed by the homomorphic encryption scheme, a ciphertext C can be strictly refreshed
  - Then can carry out at least one more operation on such ciphertexts (before refreshing again)



Refreshed: Doesn't depend on how unfresh C was, but only on the depth of D_C

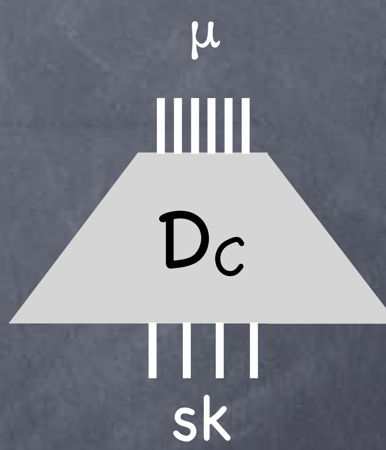


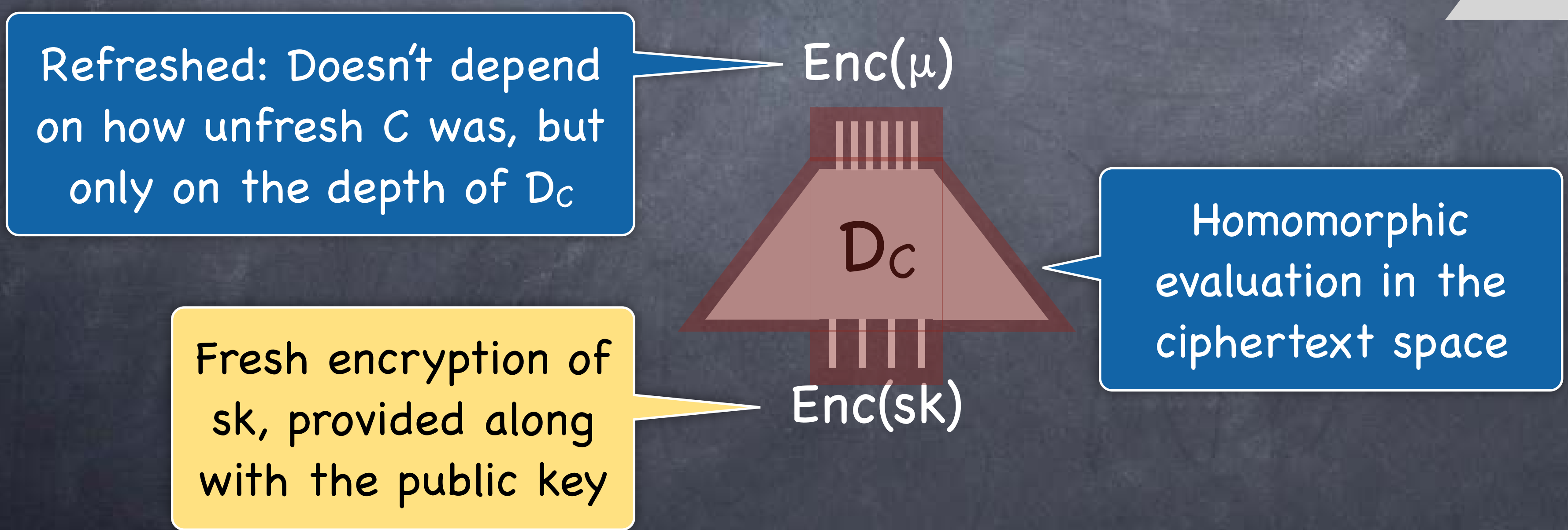
Homomorphic evaluation in the ciphertext space

Fresh encryption of sk, provided along with the public key

Bootstrapping

- Circularity: Encrypting the secret-key of a scheme under the scheme itself
  - Can break security in general!
- LWE does not by itself imply security
- Stronger assumption: "Circular Security of LWE"





Bootstrapping GSW

- Supports log(k) depth computation with poly(k) complexity
- Need low depth decryption (as a function of secret-key)
- $\mathrm{Dec}_z(C)$: $\underline{\mathbf{z}}^T C = \underline{\delta}^T + \mu \underline{\mathbf{z}}^T G$ where $\underline{\delta}^T = e^T R$
  - And then check if the result is close to $\underline{\mathbf{0}}^T$ or $\underline{\mathbf{z}}^T G$
  - How?
  - Multiply by $B(\underline{\mathbf{w}})$ where last coordinate of $\underline{\mathbf{w}}$ is $\lfloor q/2 \rfloor$ and other coordinates 0
  - $\underline{\mathbf{z}}^T C\, B(\underline{\mathbf{w}}) = \underline{\delta}^T\, B(\underline{\mathbf{w}}) + \mu \underline{\mathbf{z}}^T \underline{\mathbf{w}} \;=\; \varepsilon + \mu \lfloor q/2 \rfloor$
    - Has most significant bit = μ (since error $|\varepsilon| << q/4$)
- $\mathrm{Dec}_z(C)$: MSB($\underline{\mathbf{z}}^T C\, B(\underline{\mathbf{w}})$). All operations mod q.
  - If q were small (poly(k)) this would be small depth (log(k))
  - Problem: q is super-polynomial in security parameter k
  - Idea: Can change modulus for decryption!

Modulus Switching for GSW

- $\mathrm{Dec}_z(C)$: MSB($\underline{\mathbf{z}}^T Y \bmod q$), where $Y = C\,B(\underline{\mathbf{w}})$
  - $\underline{\mathbf{z}}^T Y = \varepsilon_0 + \mu\,(q/2) + aq$ (for some $a \in \mathbb{Z}$)
- To switch to a smaller modulus $p < q$:
  - Consider $Y' = \lceil (p/q)\,Y \rfloor$. Let $\Delta = Y' - (p/q)Y$.
  - $\underline{\mathbf{z}}^T Y' = (p/q)\,\underline{\mathbf{z}}^T Y + \underline{\mathbf{z}}^T \Delta$

    $= \varepsilon_1 + \mu\,(p/2) + ap$ where $\varepsilon_1 = (p/q)\varepsilon_0 + \underline{\mathbf{z}}^T \Delta$
  - Need $\underline{\mathbf{z}}^T \Delta$ to be small. But $\underline{\mathbf{z}}^T = [\ -\underline{\mathbf{s}}^T\ 1\ ]$ for $\underline{\mathbf{s}}$ uniform in $\mathbb{Z}_q^n$.
    - Fix: LWE with small $\underline{\mathbf{s}}$ is as good as with uniform $\underline{\mathbf{s}}$ [Exercise]
- Final bootstrapping:
  - Given C, let $Y' = \lceil (p/q)\,C\,B(\underline{\mathbf{w}}) \rfloor$ where p small (poly(k)). Define function $D_{Y'}$ which does decryption mod p. Homomorphically evaluate $D_{Y'}$ on encryption of $\underline{\mathbf{z}}$ mod p (encryption is mod q).

FHE in Practice

- Several implementations in recent years
  - Prominent ones based on schemes of Fan-Vercauteren (FV) and Brakerski-Gentry-Vaikuntanathan (BGV) with various subsequent optimisations
    - BGV implementations: HELib (IBM), $\Lambda \circ \lambda$
    - FV implementations: SEAL (Microsoft), FV-NFLlib (CryptoExperts), HomomorphicEncryption R Package …
  - Both based on "Ring LWE"
- Moderately fast
  - E.g., HELib can apply AES (encipher/decipher) to about 200 plaintext blocks using an encrypted key in about 20 minutes (a bit faster without bootstrapping, if no need to further compute on the ciphertext)