Lattice Based Cryptography

Kunwar Singh

Department of Computer Science and Engineering
National Institute of Technology Tiruchirappalli

December 3, 2019

Outline

Motivation

- Alternating option
- Strong hardness guarantees
- Efficient operations, parallelizable
- No quantum algorithm (yet)
- Fully Homomorphic Encryption (Secure Computation on encrypted data)

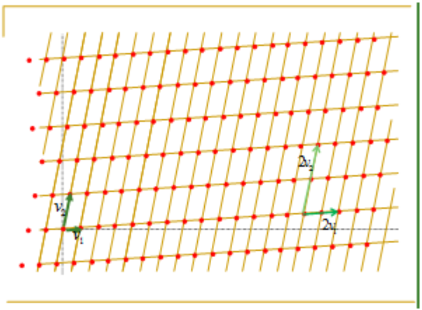
Negative

- Ciphertext blowup
- Very large (public) keys

Lattices

Definition

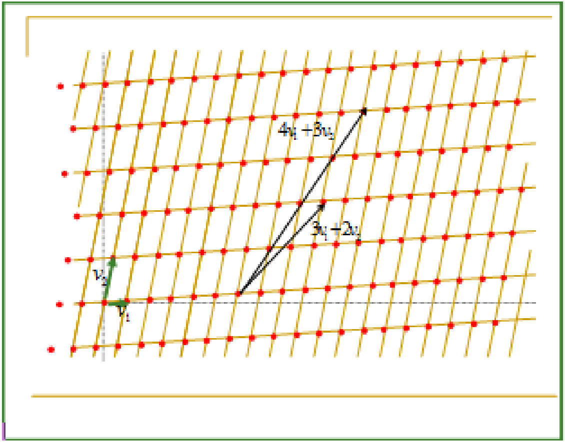
A Lattice is set of integer linear combination of n linearly independent vectors.
$L = \{b_1 x_1 + ... + b_n x_n |\ x_i :\ \text{integers}\}$ where the vectors $b_1, ..., b_n$ are a basis for L (column vectors).
Equivalently, a lattice L is a set of points in n-dimension with periodic structure.



Bases are not unique

Bases are not unique, but they can be obtained from each other by integer transforms of determinant ± 1

$$\begin{pmatrix} 3 & 4 \\ 2 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 3 & 4 \\ 2 & 3 \end{pmatrix}$$



Hard problems on lattices

Closest Vector Problem (CVP)

Given an arbitrary basis for L, and a point x(not lattice point) find the vector in L closest to x.

Shortest Vector Problem (SVP)

Given an arbitrary basis for L, find the shortest vector in L.

The Small Integer Solution (SIS) problem:

Given an integer q, a matrix $A \in Z_q^{n \times m}$ and a real β, find a nonzero integer vector $e \in Z^m$ such that $Ae = 0 \ mod \ q$ and $|e| \leq \beta$.

The Inhomogeneous Small Integer Solution (ISIS) problem:

Given an integer q, syndrome u, a matrix $A \in Z_q^{n \times m}$ and a real β, find a nonzero integer vector $e \in Z^m$ such that $Ae = u \ mod \ q$ and $|e| \leq \beta$.

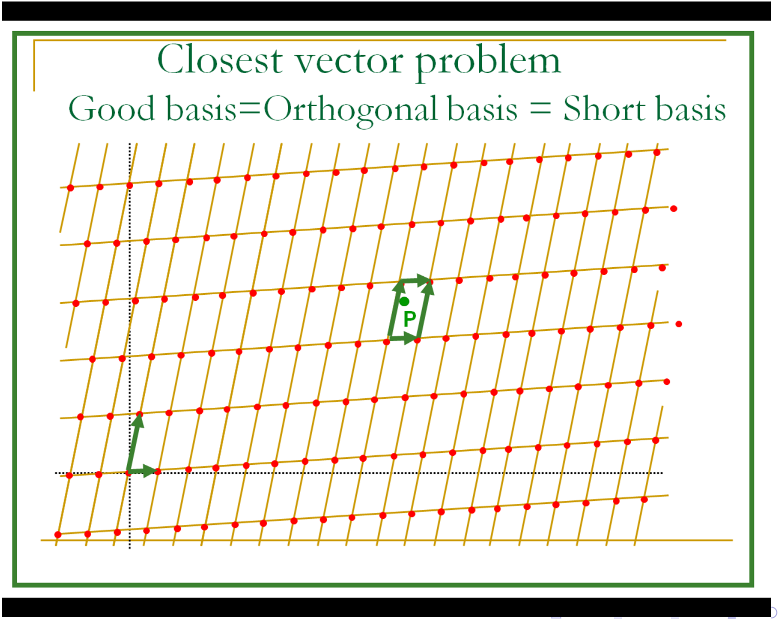
Babai's closest vertex algorithm:

Let $L \subset R^n$ has a basis $v_1, ..., v_n$ and let $w \in R^n$ be an arbitrary vector.

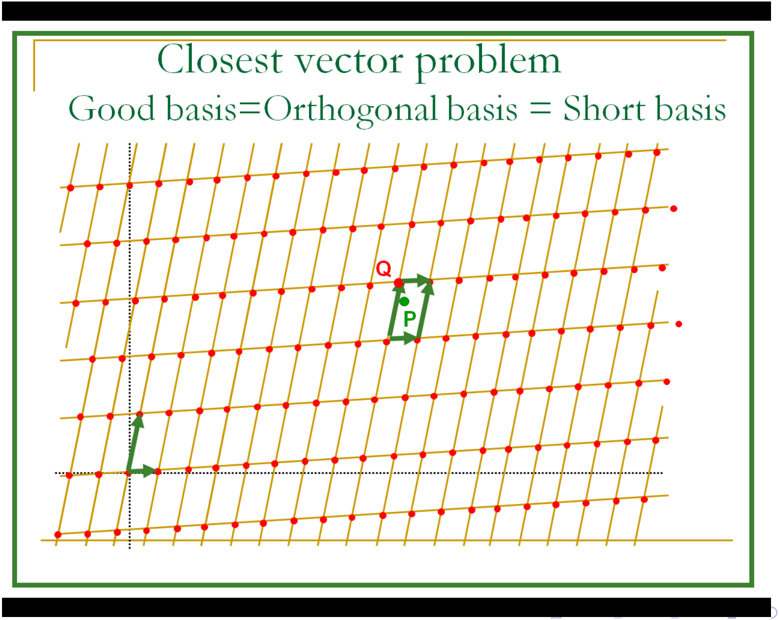
1. Write $w = t_1 v_1 + ... + t_n v_n$ with $t_1, ..., t_n \in R$.
2. Set $a_i = [t_i]$ for $i = 1, ..., n$.
3. Return the vector $a_1 v_1 + ... + a_n v_n$.

- If the vectors in the basis are reasonably orthogonal to one another, then the algorithm solves some version of apprCVP.
- If basis are highly nonorthogonal, then the vector returned by algorithm is generally far from the closest lattice vector.

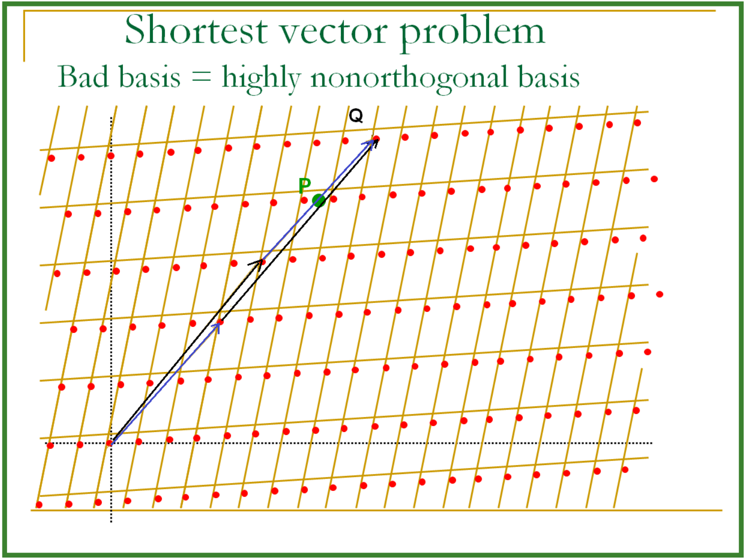
Babai's closest vertex algorithm:



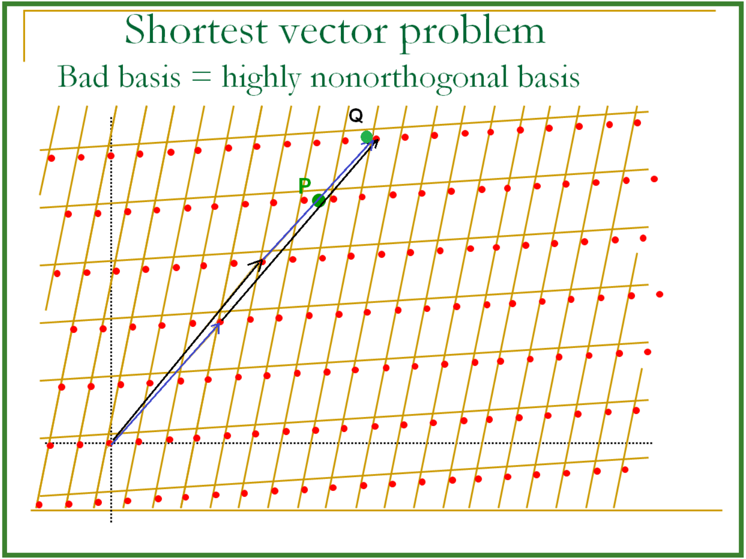
Babai's closest vertex algorithm:



Babai's closest vertex algorithm:



Babai's closest vertex algorithm:



So cryptosystem based on lattice:

- Make bad basis (long basis or highly unorthogonal basis) as public key
- Make good basis (short basis or nearly orthogonal basis) as private key
- Encrypt using bad basis, decrypt using good basis
- Recovering good basis from bad basis is hard !

Proposition (Hadamard's Inequality)

Let L be a lattice, take any basis $v_1, \ldots, v_n$ for L, and let F be a fundamental domain for L. Then

$det(L) = Vol(F) \leq |v_1||v_2| \ldots |v_n|$

Hadamard's ratio $H(v_1, \ldots, v_n) = (det(L)/|v_1||v_2| \ldots |v_n|)^{1/2}$

The GGH cryptosystem:

Key Generation

Choose a good basis $v_1, \ldots, v_n$. Choose an integer matrix U satisfying $det(U) = 1$. Compute a bad basis $w_1, \ldots, w_n$ as the rows of $W = UV$. If Hadamard's ratio is much less than 1 then publish the public key $w_1, \ldots, w_n$ else choose another U and repeat the procedure.

Encryption :

Choose small plaintext vector $m = x_1, \ldots, x_n$. Choose random small vector r. Use Receiver's public key to compute
$e = x_1 w_1 + \ldots + x_n w_n + r$.
Send the ciphtertext e to Receiver.

Decryption:

Use Babai's algorithm to compute the vector $v \in L$ closest to e. Compute vW^1 to recover m.

Provable Security approach:

Security Means:

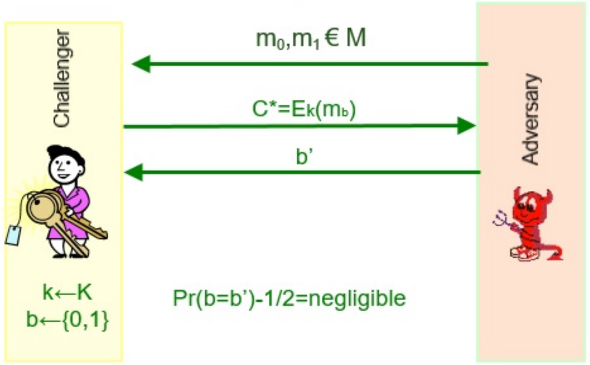
- Recovery of secret key is hard: But attacker can obtain plaintext from ciphertext.
- Obtaining plaintext from ciphertext is hard: Attacker may be able to obtain some part of the plaintext.
  For ex.: Transfer 1000 to Bob's account.
  Ciphertext:0111000. . . 100100001100. 101

Security means-:

Shannon's idea of perfect security: ciphertext should reveal no information about plaintext: Probability of obtaining any bit of information is zero.
$Pr(M = m) = Pr(M = m)/C$

Obtaining any bit of information from ciphertext is hard. Probability of obtaining any bit of information is negligible (2^{-80}).

It is equivalent to following Game!



Also called Indistinguishability under Chosen Plaintext Attack (IND-CPA)

Security means-:

$$Adv_A^{ind-cpa} = Pr[b = b'] - 1/2$$

Scheme is IND-CPA secure if it is computationally infeasible to obtain a non negligible advantage.

For perfect security (Shannon'idea)
$Adv_A^{ind-cpa} = 0$ Or $Pr[b = b'] = 1/2$.

Indistinguishability under Chosen Ciphertext Attack (IND-CCA)

In some cases attacker may know his choice of ciphertext and corresponding plaintext.
To accommodate this in game model, adversary is provided the decryption oracle.

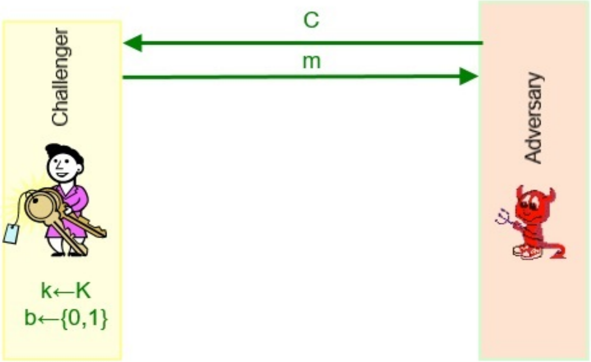
Indistinguishability under Adaptive Chosen Ciphertext Attack (IND-CCA2)

Scheme is IND-CPA secure if it is computationally infeasible to obtain a non negligible advantage.

$(i+1)^{th}$ query may depend on answer of i^{th}query. Decryption oracle is available even after receiving the challenged ciphertext.

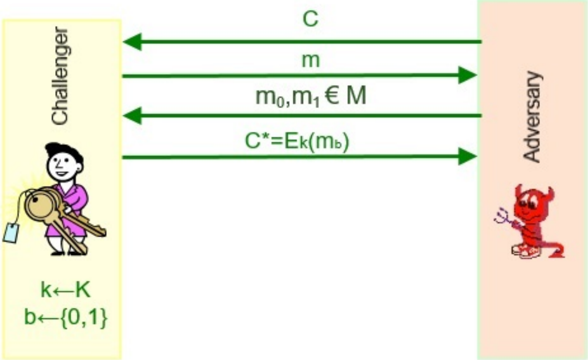
Provable security (IND-CCA) is not only a theoretical. Standard bodies, product developer asks for proof.

It is equivalent to following Game!



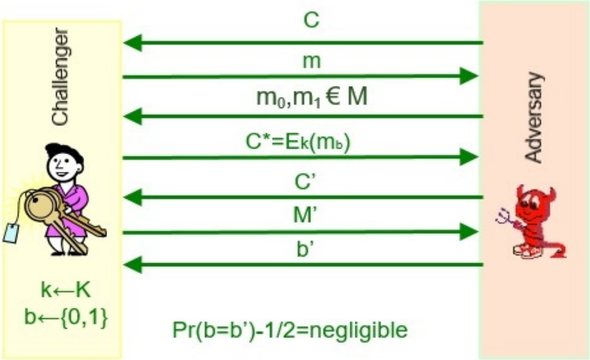
Also called Indistinguishability under Chosen Plaintext Attack (IND-CPA)

It is equivalent to following Game!



Also called Indistinguishability under Chosen Plaintext Attack (IND-CPA)

It is equivalent to following Game!



$$\mathrm{Adv}_A^{\text{ind-cpa}} = \Pr[b=b']-1/2$$

Scheme is IND-CCA2 secure if it is computationally infeasible to obtain a non negligible advantage.

Proof by Reduction

Reduction of a problem **P** to an attack *Atk*:

- Let *A* be an adversary that breaks the scheme then *A* can be used to solve **P**

Instance **I** of **P** →

→ Solution of **I**

P intractable ⇒ scheme unbreakable

Regev's Lattice Hard problem : Learning With Error (LWE)

Search

Given an arbitrary number of approximate random linear equation on $s \in Z_{17}^4$.

$$14s_1 + 15s_2 + 5s_3 + 2s_4 \approx 8 \ (mod\ 17)$$
$$13s_1 + 14s_2 + 14s_3 + 6s_4 \approx 16 \ (mod\ 17)$$
$$6s_1 + 10s_2 + 13s_3 + 1s_4 \approx 3 \ (mod\ 17)$$
$$\vdots \qquad \vdots$$
$$\vdots \qquad \vdots$$

Find $s \in Z_q^n$ is hard, when $n \approx 2^9$ and q is polynomial in n.

More precisely:

- Fix a size parameter $n \geq 1$, a modulus $q \geq 2$ and an probability distribution (Gaussian) χ on Z.
- An oracle (who knows s) generates a uniform vector $a \in Z_q^n$ and noise $e \in Z$ according to χ.
- The Oracle outputs $(a, < a, s > +e)$.
- This procedure is repeated arbitrary number of times with same s and fresh a and e.
- Find s is hard.

Decision version

Distinguish between following two oracles:

Oracle 1:

- Outputs samples of the form $(a, < a, s > +e)$ where s is fixed, a is uniform in Z_q^n and $e \in Z$ is fresh sample from χ.

Oracle 2:

- Outputs truly uniform samples from $Z_q^n \times Z_q$.

LWE Based Public Key Cryptosystem (Regev):

System Parameter:

Integers n (the security parameter), m (number of equations), q modulus, and a real $a > 0$ (noise parameter).

Private Key:

is a vector $s \in_R Z_q^n$.

Public Key:

consists of m samples $(a_i, b_i)_{i=1}^m$ from the LWE distribution with secret s.
$a_i \in_R Z_q^n$ and $b_i \in_R Z_q$. OR
$A \in_R Z_q^{n \times m}$ and $b = s^T A + e \in Z_q^m$.

Encryption:

To encrypt each bit of message, do the following.
Choose a string $x \in_R \{0,1\}^m$
Compute $u = \sum x_i a_i = Ax,\ u' = bit \left\lfloor \frac{q}{2} \right\rfloor + b^t x$.

Decryption:

Compute $u' - s^T u = bit \left\lfloor \frac{q}{2} \right\rfloor + ex$. Output is 0 if $u' - s^T u$ is closer to 0 than $\left\lfloor \frac{q}{2} \right\rfloor$ and 1 otherwise.

Correctness:

- Error $= \sum x_i a_i = ex$. Error is atmost m normal error terms each with standard deviation αq and mean zero.
  Sum of normal distribution is also normal distribution with mean zero and variance $\sigma^2 = m\alpha^2 q^2 \leq \frac{q^2}{(log\ \ n)^3}$, since $\alpha = \frac{1}{\sqrt{n}(log\ \ n)^2}$.
  $\frac{q}{4} = \frac{(log\ \ n)^{3/2}}{4} \times \frac{q}{(log\ \ n)^{3/2}} \geq 10\sigma$.
- Hence probability that error term is greater than $q/4$ is negligible.

Dual Public Key Cryptosystem under LWE Assumption (GPV):

System Parameter:

Same as previous.

Private Key:

is a vector $x \in_R \{0,1\}^m$.

Public Key:

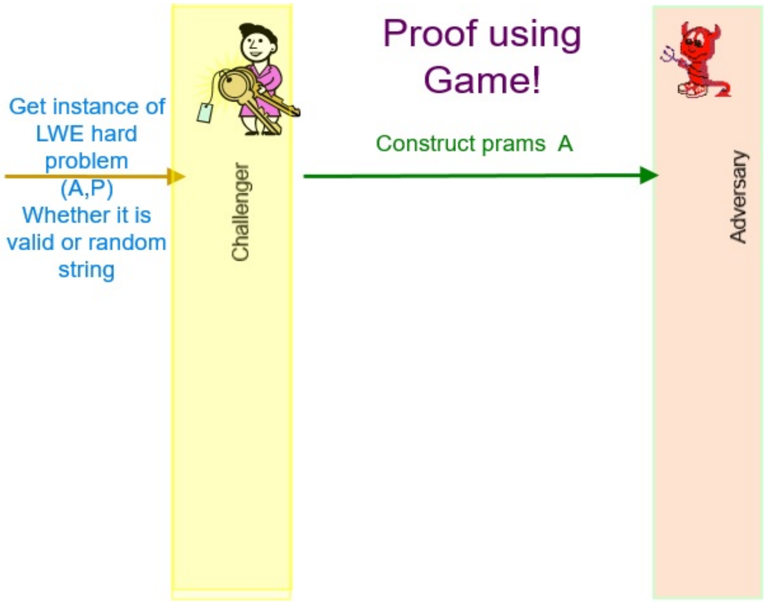
$A \in_R Z_q^{n\times m}$ and $b = Ax \in Z_q^n$.

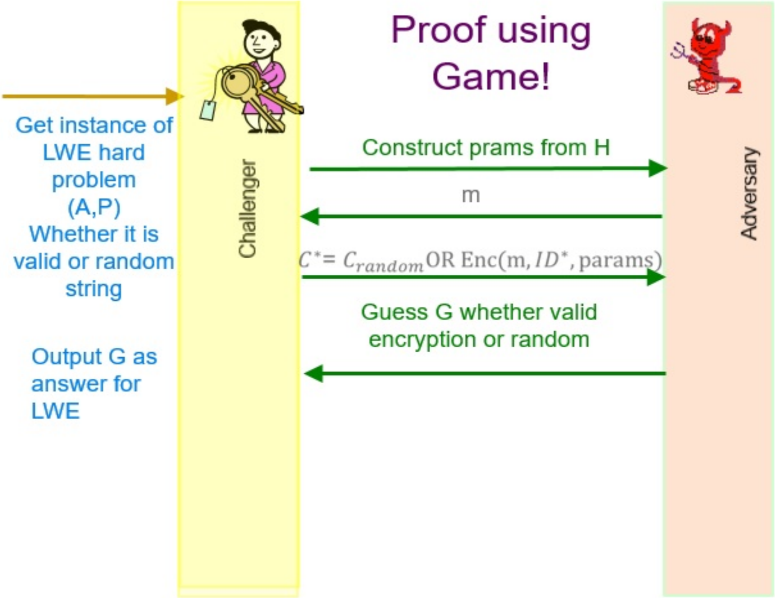
Encryption:

To encrypt each bit of message, do the following.
Choose a $s \in_R Z_q^n$
Compute $u = s^T A + e \in Z_q^m$, $u' = bit \left\lfloor \frac{q}{2} \right\rfloor + s^t b + e'$.

Decryption:

Compute $u' - ux = bit \left\lfloor \frac{q}{2} \right\rfloor + ex$. Output is 0 if $u' - ux$ is closer to 0 than $\left\lfloor \frac{q}{2} \right\rfloor$ and 1 otherwise.


Proof using Game!
Get instance of LWE hard problem (A,P) Whether it is valid or random string
Challenger
Construct prams A
Adversary


Proof using Game!
Get instance of LWE hard problem (A,P) Whether it is valid or random string
Output G as answer for LWE
Challenger
Adversary
Construct prams from H
m
$C^* = C_{random}$ OR Enc(m, ID^*, params)
Guess G whether valid encryption or random

Identity Based Cryptosystem

History:

- Concept was proposed by Adi Shamir in 1984.
- Scheme was independently discovered by Boneh-Franklin and Cocks in 2001.

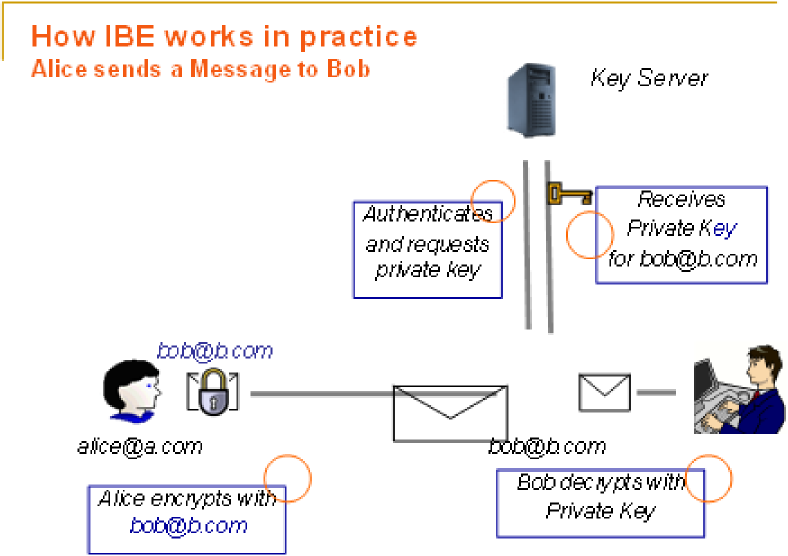
Definition:

- IBE is a public key encryption system in which an arbitrary string which uniquely identifies the user can be used as public key.
- For example an email or phone number can be a public key.

Benefits:

- No certificates are required. A recipient's public key is derived from his identity.

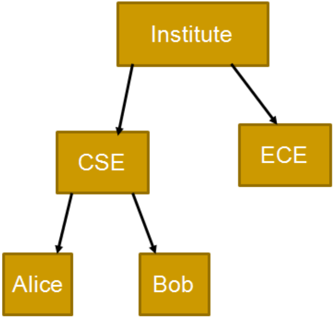
Identity Based Cryptosystem



Hierarchical IBE (HIBE)

HIBE primitive [HL02, GS02]

- PKG (root) delegates the capability for providing private key generation and identity authentication to lower level PKGs.
- There are no lower level public parameters. Only the PKG (root) has public parameters.
- Alice can obtain her private key from her "local" key generation centre.



CSE : Lower level KGC

ID_{Alice} = (Institute,CSE,Alice)

HIBE Scheme

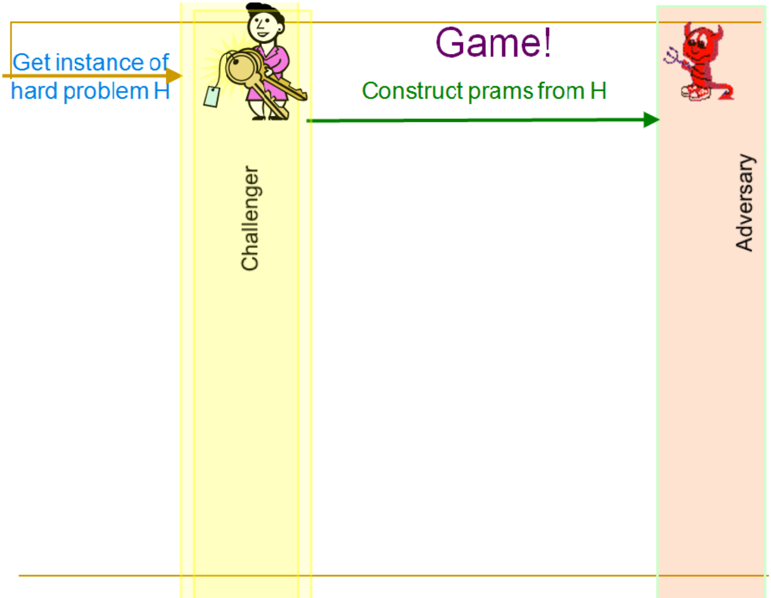
HIBE

- **Setup**(d, λ): outputs the public parameters and master key of root PKG.
- **Derive**$(PP, (id/id_l), SK_{\text{prefix of } (id/id_l)})$: outputs private key for the identity (id/id_l) at depth l.
  If $l = 1$ then $SK_{(id/id_0)}$ is defined to be master key of root PKG.
- **Encrypt**$(PP, (id/id_l), M)$: outputs ciphertext C.
- **Decrypt**$(PP, SK_{(id/id_l)}, C)$: outputs message M.

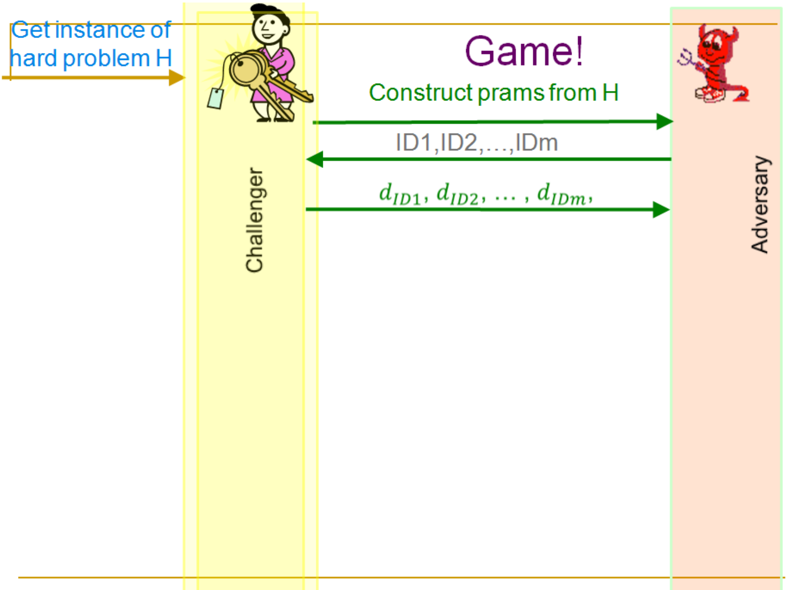
IBE

IBE is special case of HIBE when depth is one.

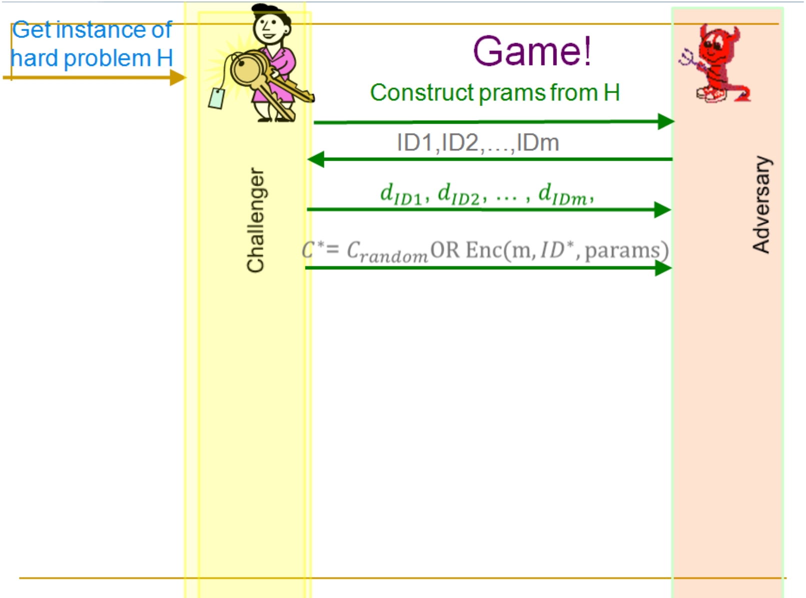
Adaptive-ID Semantic Security Model of HIBE



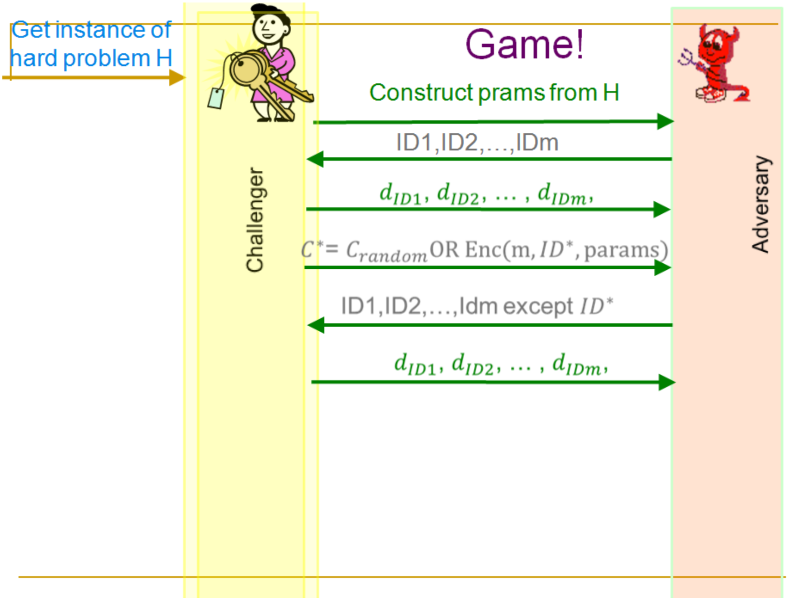
Adaptive-ID Semantic Security Model of HIBE



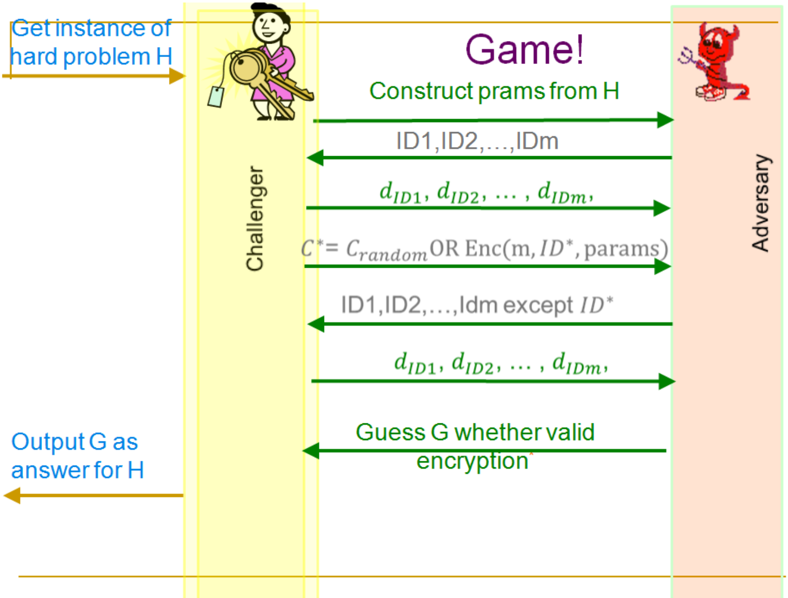
Adaptive-ID Semantic Security Model of HIBE



Adaptive-ID Semantic Security Model of HIBE



Adaptive-ID Semantic Security Model of HIBE



Identity Based Cryptosystem under LWE Assumption:

Theorem (Ajtai and Peikert):

Let $q \geq 3$ be odd and $m = \lceil 6n \; log \; q \rceil$. There is probabilistic polynomial-time algorithm $TrapGen(q, n)$ that outputs a pair $(A \in Z_q^{n \times m}, S \in Z^{n \times m})$ such that A is statistically close to a uniform matrix in $Z_q^{n \times m}$ and S is a short basis for $\Lambda_q^{\perp}(A)$ $(\Lambda_q^{\perp}(A) := \{e \in Z^m \;\; s.t. \;\; Ae = 0 \; (mod \; q)\})$.

SetUp:

Hash function $H : \{0, 1\}^* \longrightarrow Z_q^n$. Generate a trapdoor function f_A with trapdoor T (short basis) by running Algo TrapGen(q, n). $f_A(e) = Ae \; mod \; q$. The master public key is A and master secret key is T.

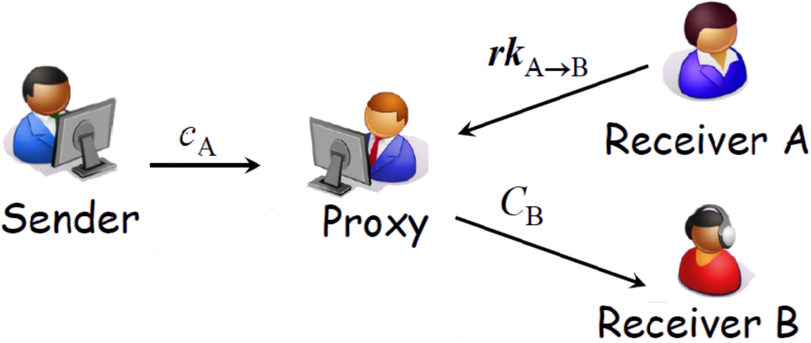
Extract:

Public Key is $id \in \{0, 1\}^*$. Compute $u = H(id)$. Using Preimage Sampler algorithm with trapdoor T compute decryption key short $e \longleftarrow f_A^{-1}(u)$ such that $Ae = u \; mod \; q$. Return e.

Proxy Re-Encryption

Proxy Re-Encryption

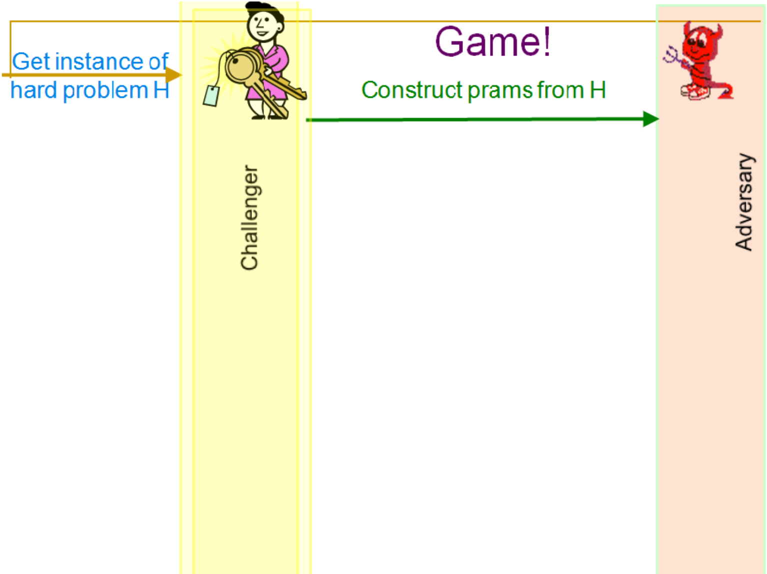
Blaze, Bleumer and Strauss (BBS) presented a new primitive called proxy re-encryption in 1998. *PRE* allows semi trusted proxy to convert a ciphertext for Alice into a ciphertext for Bob without knowing the message.



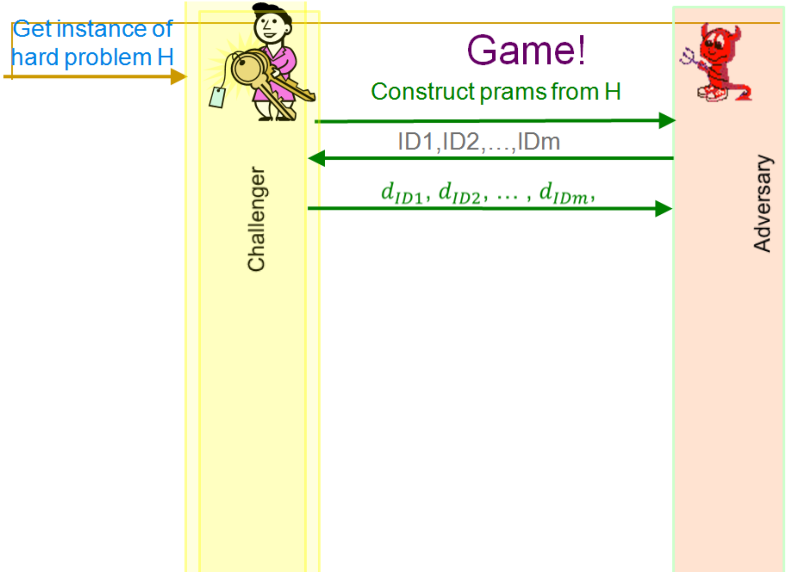
Unidirectional Proxy Re-Encryption Scheme

- **PublicParameters(n):** outputs public parameters.
- **KeyGeneration(n):** outputs a secret key sk and the corresponding public key pk of the user.
- **Encrypt(pk, M):** outputs ciphertext C.
- **Re-Encryption Key(sk_i, pk_i, pk_j):** outputs unidirectional reencryption key $rk_{i,j}$.
- **Re-Encryption($rk_{i,j}, C_i$):** outputs a re-encrypted ciphertext C_j.
- **Decrypt(sk_j, C_j):** outputs message m.

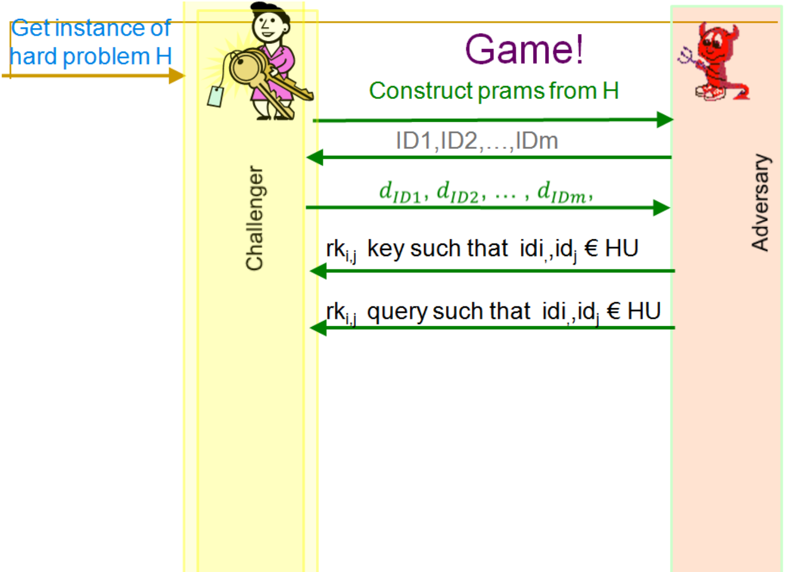
Semantic Security Model for Unidirectional Proxy Re-Encryption Scheme



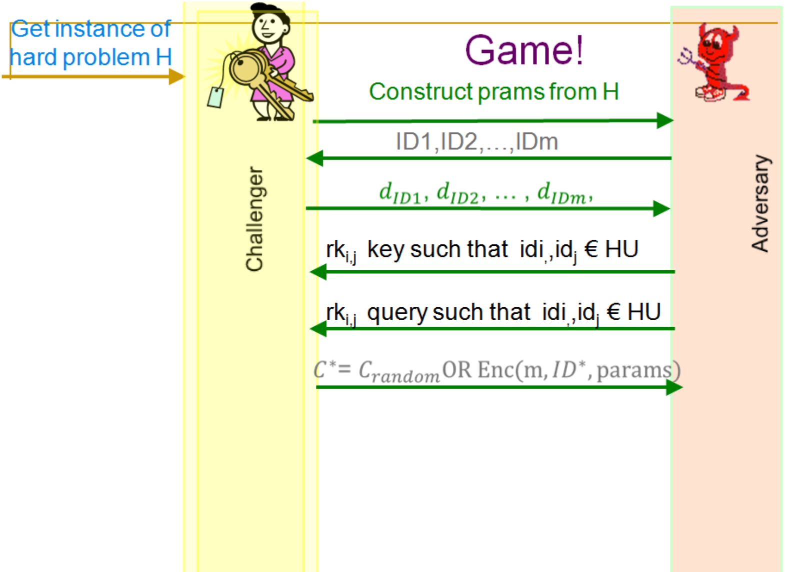
Semantic Security Model for Unidirectional Proxy Re-Encryption Scheme



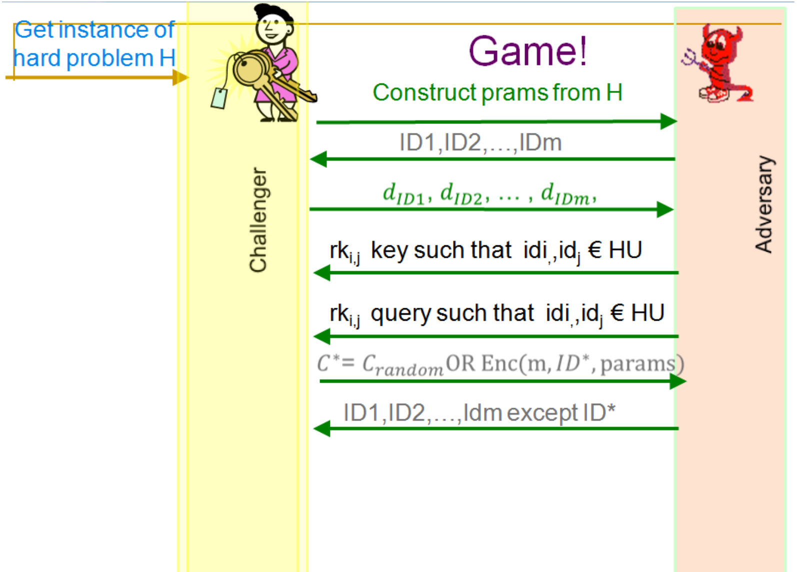
Semantic Security Model for Unidirectional Proxy Re-Encryption Scheme



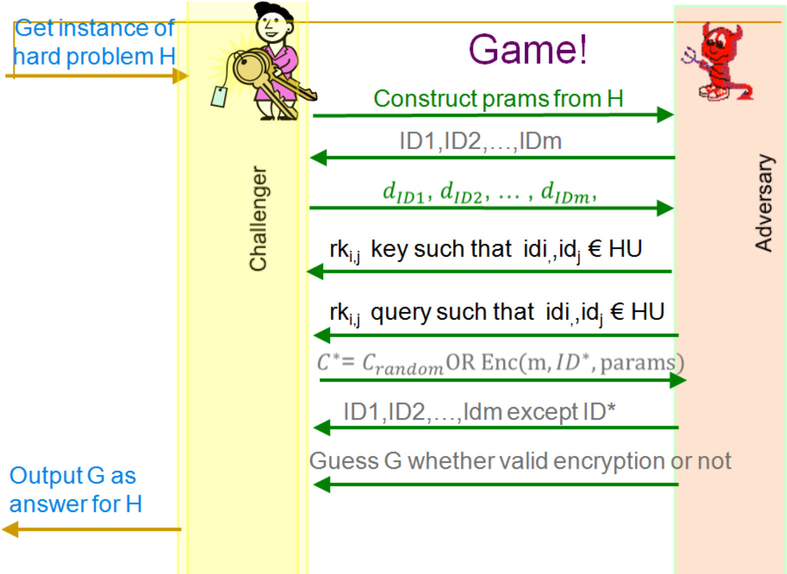
Smantic Security Model for Unidirectional Proxy Re-Encryption Scheme



Semantic Security Model for Unidirectional Proxy Re-Encryption Scheme



Semantic Security Model for Unidirectional Proxy Re-Encryption Scheme



Bi-Directional Proxy Re-Encryption

- **Key Generation**
- $< g >= G$ of prime order q.
- $SK_a = a \in \ Z_q^*$, $SK_b = b \in \ Z_q^*$
- $PK_a = g^a$, $PK_b = g^b$
- $RK_{A \to B} = b/a = b.a^{-1}(modq)$
- **Encryption:**
- $m \in \ G$, random $r \in \ Z_q^*$
- $C_a = (g^r.m, g^{ar})$
- **Decryption:**
- $m = \frac{g^r.m}{(g^{ar})^{1/a}}$.
- **Re-encryption:**
- $C_a = (g^r.m, g^{ar})$
- $C_b = (g^r.m(g^{ar})^{RK_{A\to B}})$
- $= (g^r.m(g^{ar})^{b/a})$
- $= (g^r.m, g^{br})$

Uni-Directional Proxy Re-Encryption

- **Key Generation:**
- $< g > = \mathrm{G}_1$ of prime order q.
- $SK_a = a \in Z_q^*$, $SK_b = b \in Z_q^*$
- $PK_a = g^a$, $PK_b = g^b$
- **Re-encryption key:**
- $RK_{A \to B} = (g^b)^{1/a} = g^{b/a}$
- **Encryption:**
- $m \in G_2$
- $C_a = (Z^r.m, g^{ra})$ compute $Z = e(g, g)$ where $e(g,g)$ is a bilinear pairing
- **Re-Encryption:**
- $C_a = (Z^r.m, g^{ra})$
- $C_b = (Z^r.m, e(g^{ra}, RK_{A\to B})) = (Z^r.m, e(g^{ra}, g^{b/a}) = (Z^r.m, Z^{rb})$
- **Decryption:**
- $m = \frac{Z^r.m}{(Z^{rb})^{1/b}}$.

Bi-Directional Proxy Re-Encryption (Keita Xagawa)

Setup(n)

On input a security parameter n, we set the parameters $q = poly(n)$ and $m = O(nlg\ n)$ accordingly. We choose a matrix $A \in Z_q^{n\times m}$ randomly. Public parameters (PP) is matrix A.

KeyGeneration(n):

We choose noise matrices $S \in \psi_s^{n\times l}$ and $E \in \psi_s^{n\times l}$. We compute $P = S^T A + E$.
So private key is S and public key $P \in (Z_q^{l\times m})$.

RKGen(PP, S_A, P_B):

On input of Alice's private key S_A and Bob's private key S_B, re-encryption key $rk_{A,B} = S_A - S_B$.

Encrypt(mpk, b):

To encrypt a bit $b \in \{0, 1\}$, we do the following.

- We choose $s \leftarrow Z_q^n$ uniformly.
- Compute $p = A^T s + e$, where $e \leftarrow \chi^m$. Here χ^m is error (Gaussian) distribution.
- Compute $c_i = u^T s + b\lfloor \frac{q}{2} \rfloor + \overline{e}$, where $\overline{e} \leftarrow \chi$. Here χ is error (Gaussian) distribution.
- Output the ciphertext $C = (p, c_i) \in (Z_q^m \times Z_q)$.

.

Re-Encrypt($PP, rk_{i,j}, ((p, c_i) = C_i)$

. We compute

$$\begin{aligned} c_j &= c_i - rk_{i,j}^T p \\ &= u_i^T s + b\lfloor \frac{q}{2} \rfloor + \bar{e} - (sk_i^T - sk_j^T)(A^T s + e) \\ &= u_j^T s + b\lfloor \frac{q}{2} \rfloor + \bar{e}' \end{aligned}$$

where $\bar{e}' = \bar{e} - ({sk_i}^T - sk_j^T)e$.
New error $\bar{e}'$ may be greater than $\bar{e}$ but error after decrypting C_j will be around same as error after decrypting C_{id_i}.

Decrypt(PP, x_j, C_j):

To decrypt $C_j = (p, c_j)$, we do the following.

- We compute $b' = c_j - sk_j^T p$.
- If b' is closer to 0 than $\lfloor \frac{q}{2} \rfloor$ mod q output 0 otherwise output 1.

Master Secret Security

Ateniese et al. introduced *master secret security* as another security requirement for unidirectional *PRE*. *Master secret security* demands that no coalition of dishonest proxy and malicious delegatees can compute the master secret key (private key) of the delegator. Ateniese et al. gave following motivation for *master secret security*.

1. Some PRE may define two or more type of encryption schemes. In one encryption scheme ciphertext may be decrypted by only master secret (private key) of the delegator. Other encryption scheme re-encrypted ciphertext may be decrypted by private key of the delegatee.
2. Delegator may want to delegate decryption rights to delegatee but may not want to delegate signing rights to delegatee. With this security it is possible.

Aono et al.'s Unidirectional Proxy Re-Encryption Scheme

Let $S = [S_1|...|S_l] \in Z_q^{n \times l}$ where S_i are columns. Then Power2(S) is defined as

$$Power2(S) = \begin{bmatrix} S_1 \dots S_l \\ 2S_1 \dots 2S_l \\ \vdots \quad\quad \vdots \\ 2^{k-1}S_1 \dots 2^{k-1}S_l \end{bmatrix} \in Z_k^{nk \times l}$$

Here first n rows are S. So if we know Power2(S) then we can find S.

Proxy Key Gen(PP, S_A, P_B):

On input of Alice's private key S_A and Bob's public key P_B, do the following.

1. Bob chooses matrices $X \in \psi_s^{nk \times l}$ ($k = \lceil lg\ q \rceil$) randomly and noise Matrix $E \in \psi_s^{nk \times l}$ where ψ_s is a Gaussian distribution. Bob computes $-XS_B + E$ and sends X, $-XS_B + E$ secretly to the Alice.
2. Alice compute proxy re-encryption key $rk_{A,B} = (P_B, Q)$ where

$$Q = \begin{bmatrix} X & -XS_B + E + Power2(S_A) \\ 0_{l \times n} & I_{l \times l} \end{bmatrix}$$

Cryptanalysis...

In Aono et al's scheme, if proxy and delegatee collude they can compute delegator's private key. It works as follows.

Now let us see the expression of proxy key Q

$$Q = \begin{bmatrix} X & -XS_B + E + Power2(S_A) \\ 0_{l\times n} & I_{l\times l} \end{bmatrix},$$

where S_B is private key of Bob (delegatee). Bob (delegatee) creats X, E and securely sends $X, -XS_B + E$ to Alice. Basically Bob knows $X, -XS_B + E$.

Our Lattice Based Unidirectional Proxy Re-Encryption Scheme

Set(n):

On input a security parameter n, we set the parameters $q = poly(n)$ and $m = O(nlg\ n)$ accordingly. We choose a matrix $A \in Z_q^{n\times n}$ and matrix $X \in Z_q^{nk\times n}$ randomly, where $k = \lceil lg\ q \rceil$. Public parameters (PP) are matrix A and matrix X.

KeyGeneration(n):

Let $s = \alpha q$ for $0 < \alpha < 1$. We choose noise matrices $R, S \in \psi_s^{n\times l}$ and $E \in \psi_s^{nk\times l}$ where l is message length. We compute $P_1 = R - AS$ and $P_2 = -XS + E$.
So private key is S and public key $P = (P_1, P_2) \in (Z_q^{n\times l}, Z_q^{nk\times l})$.

Continued...

Encrypt(PP, m, P_1, P_2):

To encrypt a message $m \in \{0,1\}^l$, we do the following.

1. We choose noise vectors $e_1, e_2 \in \psi_s^{1\times n}$ and $e_3 \in \psi_s^{1\times l}$ where ψ_s is a gaussian distribution.
2. Compute $c_1 = e_1 A + e_2 \in Z_q^{1\times n}$, $c_2 = e_1 P_1 + e_3 + m\lfloor \frac{q}{2} \rfloor$.
3. Output the ciphertext $C = (c_1, c_2) \in Z_q^{1\times(n+l)}$.

RKGen(PP, S_A, P_B):

On input of Alice's private key S_A and Bob's public key P_B, we do the following.

1. We choose noise vectors $e_4 \in \psi_s^{nk\times nk}$ and $e_5 \in \psi_s^{nk\times l}$ where ψ_s is a gaussian distribution.
2. We compute proxy re-encryption key $rk_{A,B} = Q$ where

$$Q = \begin{bmatrix} e_4 X & e_4 P_2 + e_5 + Power2(S_A) \\ 0_{l\times n} & I_{l\times l} \end{bmatrix}$$

Continued...

Re-Encrypt$(PP, rk_{A,B}, C_A)$:

On input of re-encryption key $rk_{A,B}$, proxy transforms Alice'ciphertext C_A to Bob's ciphertext C_B by the following equation.

$$C_B = (c_{1B}, c_{2B}) = [Bits(c_1)|c_2].rk_{A,B} \in Z_q^{1\times(n+l)}$$

Decrypt(PP, S_B, C_B):

To decrypt $C_B = (c_1, c_2)$, we do the following.

1. We compute
$$m = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} S_B \\ I_{l\times l} \end{bmatrix}$$
2. Let $m = (m_1, \ldots, m_l)$. If m_i is less than $\lfloor \frac{q}{4} \rfloor$ mod q than $m_i = 0$ otherwise $m_i = 1$.

Correctness:

First we decrypt the normal ciphertext

$$c_1 S_A + c_2 = e_2 S_A + e_3 + m\lfloor \frac{q}{2} \rfloor,$$

which will yield m if $e_2 S_A + e_3$ is less than $\lfloor \frac{q}{4} \rfloor$. Now we decrypt the re-encrypted ciphertext

$$Bits(c_1)|c_2]rk_{A,B} \begin{bmatrix} S_B \\ I_{l\times l} \end{bmatrix}$$

$$\begin{aligned}
&= [Bits(c_1)|c_2] \begin{bmatrix} e_4 X & e_4 P_2 + e_5 + Power2(S_A) \\ 0_{l\times n} & I_{l\times l} \end{bmatrix} \begin{bmatrix} S_B \\ I_{l\times l} \end{bmatrix} \\
&= [Bits(c_1)|c_2] \begin{bmatrix} e_4 E + e_5 + Power2(S_A) \\ I_{l\times l} \end{bmatrix} \\
&= Bits(c_1)e_4 E + Bits(c_1)e_5 + Bits(c_1)Power2(S_A) + c_2 \\
&= Bits(c_1)e_4 E + Bits(c_1)e_5 + c_1 S_A + c_2 \\
&= Bits(c_1)e_4 E + Bits(c_1)e_5 + e_2 S_A + e_3 + m\lfloor \frac{q}{2} \rfloor
\end{aligned}$$

Shamir's Secret Sharing

Shamir's Secret Sharing:

Shamir's secret sharing divides the data D into u pieces (shares) in such a way that:

- At least t- shares are required to construct the data D.
- No information about data D is revealed from t-1 shares or less.

Theorem:

Given t points in the 2- dimensional plane $(x_1, y_1), \ldots, (x_t, y_t)$ with distinct x's, there is one and only one polynomial of degree t-1 such that $q(x_i) = y_i$ for all i.

Shamir's sharing protocol:

Shamir's sharing protocol:

Goal is to create n- secret shares of the secret s such that at least t shares are required to compute D.

1. Dealer D pick a random $t-1$ degree polynomial $q(x) = a_0 + a_1x +, \dots, +a_{t-1}x^{t-1}$ in which $s = a_0$. Here all coefficients a_i $(0 \leq i \leq t-1)$ are from field $(F_p : prime\ p)$.
2. Dealer D computes $q(1), q(2), \dots, q(u)$ and secretly distributes each player j the share $q(j)$. Hence the shares are denoted as $q(1), q(2), \dots, q(u)$.
3. From these t- points we can construct polynomial $q(x)$ of degree $t-1$ and can find secret $s = q(0)$. One can construct polynomial by using Lagrange interpolation.

Lagrange polynomial:

Lagrange polynomial:

Given t points in the 2- dimensional plane $(x_1, y_1), \ldots, (x_t, y_t)$ with distinct x's, then the unique polynomial passing through these points in the Lagrange form is a linear combination

$$q(x) = L(x) = \sum_{i=0}^{t-1} y_i l_i(x)$$

Lagrange basis polynomials:

$$l_i(x) = \prod_{0 \le m \le t-1, m \ne i} \frac{(x - x_m)}{(x_i - x_m)} = \frac{(x - x_0)}{(x_i - x_0)} \cdots \frac{(x - x_{i-1})}{(x_i - x_{i-1})} \frac{(x - x_{i+1})}{(x_i - x_{i+1})} \cdots \frac{(x - x_{t-1})}{(x_i - x_{t-1})}$$

Homomorphic Property of Shamir's secret sharing:

Homomorphic Property:

- If we multiply a constant to all secret shares (y-values) then a constant will be multiplied to secret to get new secret.
- Suppose we have one secrets s and their corresponding shares are $f(1), \ldots, f(n)$ for polynomial $f(x)$. We have another secret t and their corresponding shares are $g(1), \ldots, g(n)$ for polynomial $g(x)$.
  For new function $h(x) = f(x) + g(x)$ and their corresponding shares $h(1)(f(1) + g(1)), \ldots, h(n)(f(n) + g(n))$, then new secret will be $s + t$ since $h(0) = f(0) + g(0)$.

Homomorphic Property of Shamir's secret sharing:

Homomorphic Property:

- If we multiply a constant to all secret shares (y-values) then a constant will be multiplied to secret to get new secret.
- Suppose we have one secrets s and their corresponding shares are $f(1), \ldots, f(n)$ for polynomial $f(x)$. We have another secret t and their corresponding shares are $g(1), \ldots, g(n)$ for polynomial $g(x)$.
  For new function $h(x) = f(x) + g(x)$ and their corresponding shares $h(1)(f(1) + g(1)), \ldots, h(n)(f(n) + g(n))$, then new secret will be $s + t$ since $h(0) = f(0) + g(0)$.

CHINESE REMAINDER THEOREM

To solve a set of congruent equations with one variable but different modulo, which are relatively prime to each other.

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\dots \\ x &\equiv a_k \pmod{m_k} \end{aligned}$$

METHOD TO SOLVE CHINESE REMAINDER THEOREM

1. $M = m_1 \times m_2 \times \ldots \times m_k$.
2. $M_1 = M/m_1$, $M_2 = M/m_2$, …, $M_k = M/m_k$.
3. Since $\gcd(M_i, m_i) = 1$. Inverses $M_i^{-1} \bmod m_i$ exists for $1 \le i \le k$.

4. The solution to the simultaneous equations is

$$\mathbf{x = (a_1 M_1(M_1^{-1} \bmod m_1) + a_2 M_2(M_2^{-1} \bmod m_2) + \ldots + a_k M_k(M_k^{-1} \bmod m_k)) \bmod M}$$

Sun- Tsu's Puzzle

Find an integer that has a remainder of 2 when divided by 3, a remainder of 3 when divided by 5 and a remainder of 2 when divided by 7.

$$\begin{aligned} x &\equiv 2 \pmod 3 \\ x &\equiv 3 \pmod 5 \\ x &\equiv 2 \pmod 7 \end{aligned}$$

Solution is $x = 23$.
This value satisfies all equations:
$23 \equiv 2 \pmod 3$, $23 \equiv 3 \pmod 5$, and $23 \equiv 2 \pmod 7$.

We follow the four steps.

1. $M = 3 \times 5 \times 7 = 105$

2. $M_1 = 105 / 3 = 35$, $M_2 = 105 / 5 = 21$, $M_3 = 105 / 7 = 15$

3. The inverses are $M_1^{-1} \bmod 3 = 35^{-1} \bmod 3 = (35 \bmod 3)^{-1} \bmod 3 = 2^{-1} \bmod 3 = 2$,

   Similarly $M_2^{-1} \bmod 5 = 1$, $M_3^{-1} \bmod 7 = 1$

4. $x = (2 \times 35 \times 2 + 3 \times 21 \times 1 + 2 \times 15 \times 1) \bmod 105 = 23 \bmod 105$

Another Example of Chinese remainder theorem

Suppose

$$x \equiv 1 \mod 3$$

$$x \equiv 6 \mod 7$$

$$x \equiv 8 \mod 10$$

By the Chinese remainer theorem, the solution is:

$$x \equiv 1 \times 70 \times (70^{-1} \bmod 3) + 6 \times 30 \times (30^{-1} \bmod 7) + 8 \times 21 \times (21^{-1} \bmod 10)$$

$$\equiv 1 \times 70 \times (1^{-1} \bmod 3) + 6 \times 30 \times (2^{-1} \bmod 7) + 8 \times 21 \times (1^{-1} \bmod 10)$$

$$\equiv 1 \times 70 \times 1 + 6 \times 30 \times 4 + 8 \times 21 \times 1 \bmod 210$$

$$\equiv 958 \bmod 210$$

$$\equiv 118 \bmod 210$$

Example:

Let $S = 9$, $p = 41$ and the threshold value be $k = 3$

Choose at random a_1 and a_2 in $\mathbb{Z}_{41}$. For example $a_1 = 2$ and $a_2 = 31$

Now we have $P(x) = 9 + 2x + 31x^2$ over $\mathbb{Z}_{41}$

Then generate as many share as we wish. For example if $n = 7$ we have

$$(1, P_1) = (1, 1), \qquad (2, P_2) = (2, 14), \qquad (3, P_3) = (3, 7),$$
$$(4, P_4) = (4, 21) \qquad (5, P_5) = (5, 15), \qquad (6, P_6) = (6, 30),$$
$$(7, P_7) = (7, 25)$$

Reconstruction:

Suppose we have 3 shares $H = \{(1,1),(3,7),(7,25)\}$ then we can reconstruct the secret from

$$S = \sum_{i \in H} P_i \prod_{j \in H, j \neq i} \frac{-j}{i-j}$$

over $\mathbb{Z}_{41}$. Hence,

$$S = 1\frac{(-3)(-7)}{(1-3)(1-7)} + 7\frac{(-1)(-7)}{(3-1)(3-7)} + 25\frac{(-1)(-3)}{(7-1)(7-3)}$$

$$S = \frac{7}{4} + \frac{49}{-8} + \frac{25}{8} = \frac{48}{4} - \frac{8}{8} - \frac{16}{8} = 9$$

Fields

Def (field): *A set* F *with two binary operations* + (addition) and · (multiplication) *is called a field if*

1. $\forall\ a,b \in F,\ a+b \in F$
2. $\forall\ a,b,c \in F,$ $(a+b)+c=a+(b+c)$
3. $\forall\ a,b \in F,\ a+b=b+a$
4. $\exists\ 0 \in F,\ \forall\ a \in F,\ a+0=a$
5. $\forall\ a \in F,\ \exists\ -a \in F,\ a+(-a)=0$
6. $\forall\ a,b \in F,\ a \cdot b \in F$
7. $\forall\ a,b,c \in F,\ (a \cdot b) \cdot c=a \cdot (b \cdot c)$
8. $\forall\ a,b \in F,\ a \cdot b=b \cdot a$
9. $\exists\ 1 \in F,\ \forall\ a \in F,\ a \cdot 1=a$
10. $\forall\ a,b,c \in F, a \cdot (b+c)=a \cdot b+a \cdot c$

11 $\quad \forall\ a \neq 0 \in F,\ \exists\ a^{-1} \in F,\ a \cdot a^{-1}=1$

Ex. Field of Real numbers, Z_p = (1, …, p-1), +mod p, *mod p

4.1 Vectors in R^n

- An ordered n-tuple

  a sequence of n real numbers $(x_1, x_2, \cdots, x_n)$

- R^n-space:

  the set of all ordered n-tuples

$n = 1$ R^1-space = set of all real numbers
(R^1-space can be represented geometrically by the x-axis)

$n = 2$ R^2-space = set of all ordered pair of real numbers (x_1, x_2)
(R^2-space can be represented geometrically by the xy-plane)

$n = 3$ R^3-space = set of all ordered triple of real numbers (x_1, x_2, x_3)
(R^3-space can be represented geometrically by the xyz-space)

$n = 4$ R^4-space = set of all ordered quadruple of real numbers (x_1, x_2, x_3, x_4)

$$\mathbf{u}=\left(u_1,u_2,\cdots,u_n\right),\quad \mathbf{v}=\left(v_1,v_2,\cdots,v_n\right)\qquad \text{(two vectors in } R^n\text{)}$$

- Equality:

  $\mathbf{u}=\mathbf{v}$ if and only if $u_1=v_1,\ u_2=v_2,\ \cdots,\ u_n=v_n$

- Vector addition (the sum of **u** and **v**):

  $$\mathbf{u}+\mathbf{v}=\left(u_1+v_1,u_2+v_2,\cdots,u_n+v_n\right)$$

- Scalar multiplication (the scalar multiple of **u** by *c*):

  $$c\mathbf{u}=\left(cu_1,cu_2,\cdots,cu_n\right)$$

- Notes:

  The sum of two vectors and the scalar multiple of a vector in R^n are called the standard operations in R^n

- Difference between **u** and **v**:

$$\mathbf{u}-\mathbf{v} \equiv \mathbf{u}+(-1)\mathbf{v} = (u_1-v_1,\ u_2-v_2,\ u_3-v_3, \ldots,\ u_n-v_n)$$

- Zero vector :

$$\mathbf{0} = (0, 0, \ldots, 0)$$

4.2 Vector Spaces over Field V(F)

- Vector spaces :

Let V be a set on which two operations (addition and scalar multiplication) are defined. If the following ten axioms are satisfied for every element u, v, and w in V and every scalar c and d *in* F, then V is called a vector space over the field F, and the **elements** in $\boldsymbol{V}$ are called **vectors**

Addition:

(1) $\mathbf{u}+\mathbf{v}$ is in V

(2) $\mathbf{u}+\mathbf{v} = \mathbf{v}+\mathbf{u}$

(3) $\mathbf{u}+(\mathbf{v}+\mathbf{w}) = (\mathbf{u}+\mathbf{v})+\mathbf{w}$

(4) V has a zero vector $\mathbf{0}$ such that for every $\mathbf{u}$ in V, $\mathbf{u}+\mathbf{0} = \mathbf{u}$

(5) For every $\mathbf{u}$ in V, there is a vector in V denoted by $-\mathbf{u}$ such that $\mathbf{u}+(-\mathbf{u}) = \mathbf{0}$

Scalar multiplication:

(6) $c\mathbf{u}$ is in V

(7) $c(\mathbf{u}+\mathbf{v})=c\mathbf{u}+c\mathbf{v}$

(8) $(c+d)\mathbf{u}=c\mathbf{u}+d\mathbf{u}$

(9) $c(d\mathbf{u})=(cd)\mathbf{u}$

(10) $1(\mathbf{u})=\mathbf{u}$

- Notes: To show that a set is not a vector space, you need only find one axiom that is not satisfied

- Ex 6: The set of all integers over the field of real numbers is not a V(R)

Pf:

$$1 \in V, \text{ and } \tfrac{1}{2} \text{ is a real-number scalar}$$

$$\underset{\substack{\uparrow\\ \text{scalar}}}{(\tfrac{1}{2})}\underset{\substack{\uparrow\\ \text{integer}}}{(1)} = \underset{\substack{\uparrow\\ \text{noninteger}}}{\tfrac{1}{2}} \notin V$$

(it is not closed under scalar multiplication)

- Ex 7: The set of all (exact) second-degree polynomial functions is not a vector space

Pf: Let $p(x) = x^2$ and $q(x) = -x^2 + x + 1$

$$\Rightarrow p(x) + q(x) = x + 1 \notin V$$

(it is not closed under vector addition)

Spanning Sets and Linear Independence

- Linear combination :

A vector $\mathbf{u}$ in a vector space V is called a linear combination of the vectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k$ in V if $\mathbf{u}$ can be written in the form

$$\mathbf{u} = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \ldots + c_k\mathbf{v}_k,$$

where $c_1, c_2, \ldots, c_k$ are real-number scalars

- The span of a set: span(S)

If $S = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ is a set of vectors in a vector space V, then the span of S is the set of all linear combinations of the vectors in S,

- The span of a set: span(S)

If $S = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ is a set of vectors in a vector space V, then the span of S is the set of all linear combinations of the vectors in S,

$$\text{span}(S) = \left\{ c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k \;\middle|\; \forall c_i \in R \right\}$$

(the set of all linear combinations of vectors in S)

- Definition of a spanning set of a vector space:

If every vector in a given vector space V can be written as a linear combination of vectors in a set S, then S is called a **spanning set** of the vector space V

- Notes: The above statement can be expressed as follows

$$\begin{aligned} & \operatorname{span}(S) = V \\ \Leftrightarrow\ & S \text{ spans (generates) } V \\ \Leftrightarrow\ & V \text{ is spanned (generated) by } S \\ \Leftrightarrow\ & S \text{ is a spanning set of } V \end{aligned}$$

- Ex 4:

(a) The set $S = \{(1,0,0), (0,1,0), (0,0,1)\}$ spans R^3 because any vector

$\mathbf{u} = (u_1, u_2, u_3)$ in R^3 can be written as

$$\mathbf{u} = u_1(1,0,0) + u_2(0,1,0) + u_3(0,0,1)$$

(b) The set $S = \{1, x, x^2\}$ spans P_2 because any polynomial function

$p(x) = a + bx + cx^2$ in P_2 can be written as

$$p(x) = a(1) + b(x) + c(x^2)$$

- Definitions of Linear Independence (L.I.) and Linear Dependence (L.D.) :

$$S=\{\mathbf{v}_1,\mathbf{v}_2,\cdots,\mathbf{v}_k\}\text{: a set of vectors in a vector space } V$$

For $c_1\mathbf{v}_1+c_2\mathbf{v}_2+\cdots+c_k\mathbf{v}_k=\mathbf{0}$

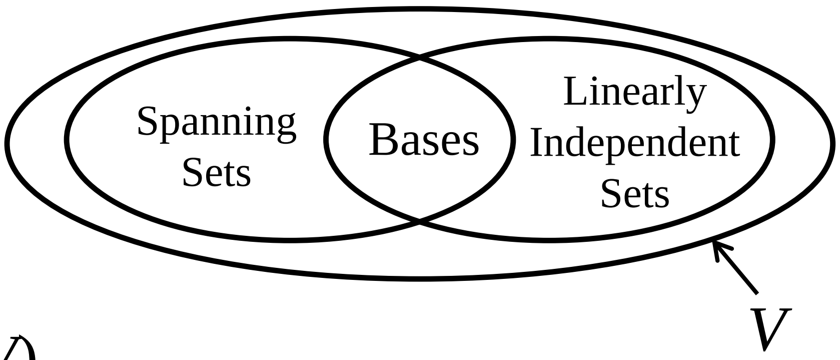
- $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ are Linear Independence (L.I.) if and only if $c_1 = \mathbf{c}_2 = \ldots = c_n = 0$

4.5 Basis and Dimension

- Basis :

  V: a vector space

  $S = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\} \subseteq V$

  S spans V (i.e., span(S) = V)

  S is linearly independent

  $\Rightarrow$ S is called a basis for V

- Notes:

(1) the **standard basis** for R^3:

$\{i, j, k\}$, for $i = (1, 0, 0)$, $j = (0, 1, 0)$, $k = (0, 0, 1)$

(2) the **standard basis** for R^n :

$\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$, for $\mathbf{e}_1 = (1,0,\ldots,0)$, $\mathbf{e}_2 = (0,1,\ldots,0), \ldots,$ $\mathbf{e}_n = (0,0,\ldots,1)$

Ex: For R^4, $\{(1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1)\}$

- Theorem: Number of vectors in a basis

If a vector space V has one basis with n vectors, then every basis for V has n vectors

Any n linearly independent vectors will be basis for the vector space V