Decodificador MLD

Ramon Ribeiro

Instituto de Computação - UNICAMP

27 de Maio de 2019

Introdução

- Vou explicar superficialmente minha implementação do decodificador de Vardy e Berry "Maximum likelihood decoding algorithm", um tipo de CVP que usa glue theory

Introdução

- Vou explicar superficialmente minha implementação do decodificador de Vardy e Berry "Maximum likelihood decoding algorithm", um tipo de CVP que usa glue theory
- A principal preocupação foi em aderir aos requerimentos da criptografia. O tempo de implementação ficará em segundo plano, mas continua sendo importante para praticidade do programa.

Tópicos

- Definições de sistema
- Decodificador
- Pseudocódigos
- Exatidão
- Performance

Definições - Codificação

Codificar

Transformar a mensagem em outro formato mais conveniente para criptografia

Definições - Codificação

Codificar

Transformar a mensagem em outro formato mais conveniente para criptografia

Para o nosso problema

A codificação usada determina quantos bits são extraídos de cada coordenada k.

$$enc(m) := \lceil q/2 \rfloor . m \ , \ m \in \{0, 1\}^k$$

Definições - Decodificação dessa codificação

Decodificar

Reverter dados transformados em uma plataforma diferente.
Qualquer individuo tem essa capacidade.

Definições - Decodificação dessa codificação

Decodificar

Reverter dados transformados em uma plataforma diferente. Qualquer individuo tem essa capacidade.

Para o nosso problema

$$enc^{-1}(enc(.) + erro) = \{0, 1\}^k$$

Onde a saída é igual 0 e 1 quando $\{enc(.) + erro\}$ está mais próximo de 0 e $\lceil q/2 \rfloor$ (mod q), respectivamente.

Definições - Encriptação

Encriptação

Transformar dados em outro formato tal que somente indivíduos especifico possam reverter esse processo. Para manter a confidencialidade do dado.

Definições - Encriptação

No nosso problema

Dada uma distribuição χ, toma-se uma matriz aleatória $\boldsymbol{A} \in \mathbb{Z}_q^{nxm}$, pública.
Para os pares de chaves privadas e públicas, duas matrizes são amostradas de uma distribuição de erro como $\boldsymbol{S}, \boldsymbol{E} \overset{\leftarrow}{\chi} \mathbb{Z}^{nxk}$.
$\boldsymbol{S}$ é a chave privada e a pública é definida como
$\boldsymbol{B} := \boldsymbol{AS} + \boldsymbol{E} \in \mathbb{Z}_q^{nxk}$.

Definições - Encriptação

Encriptação

Concatenamos A e B em uma matriz:

$$A' = \begin{bmatrix} A^t \\ B^t \end{bmatrix}$$

Depois, dois vetores são amostrados da distribuição: $s' \overleftarrow{\chi} \mathbb{Z}^n$, $e' \overleftarrow{\chi} \mathbb{Z}^{n+k}$. Finalmente encriptamos a mensagem $m \in \{0,1\}^t$ usando a chave publica B:

$$c = A's' + e' + (0, enc(m)) \in \mathbb{Z}_q^{n+k}$$

Definições

Decriptar

Reverter dados transformados em uma plataforma diferente.
Somente possível com a chave secreta correspondente.

Definições

Decriptar

Reverter dados transformados em uma plataforma diferente.
Somente possível com a chave secreta correspondente.

No nosso problema

Para decriptar c dado a chave secreta S

$$
\begin{aligned}
[-S^t \; I_k].c &= [-S^t \; I_k].(A's' + e') + enc(m) \\
&= -S^t A^t s' + B^t s' + [-S^t \; I_k].e' + enc(m) \\
&= -S^t A^t s' + (AS)^t s' + [-S^t \; I_k].e' + enc(m) \\
&= [-S^t \; I_k].e' + enc(m) \\
&\approx enc(m) \; mod \; q
\end{aligned}
$$

Maximum likelihood decoder

O decodificador decodifica um ponto no espaço qualquer para o ponto mais próximo do reticulado, logo um CVP, conhecido como MLD.

Decodificando somas diretas

Para decodificar uma decomposição de uma soma direta $\mathcal{L} = \mathcal{L}_1 \oplus \ldots \oplus \mathcal{L}_i$, basta decodificar a projeção ortogonal do target sobre cada componente:

$$CV_t(\mathcal{L}) = \oplus_{i=1}^{k} CV_{\pi_i(t)}(\mathcal{L}_i)$$

Em que π_i é a projeção ortogonal no espaço espanado por $\mathcal{L}_i$.

Decodificando uniões de cosets

- Um algorítimo efetivo de decodificação para um reticulado $\mathcal{L}'$ pode ser facilmente aplicado em um coset $g + \mathcal{L}'$.
- Se $CV_t(\mathcal{L}')$ é o ponto de $\mathcal{L}'$ mais próximo de "t":

$$CV_t(g + \mathcal{L}') = g + CV_{t-g}(\mathcal{L}')$$

- Para um reticulado $\mathcal{L}$ com glue group $G = \mathcal{L}/\mathcal{L}'$, decodificamos:

$$CV_t(\mathcal{L}) = CV_t(\{g + CV_{t-g}(\mathcal{L}') : g \in G)$$

$$CV_t(\mathcal{L}) = CV_t(\{g + \oplus_{i=1}^{k} CV_{\pi_i(t-g)}(\mathcal{L}') : g \in G)$$

Decodificando com glue parity

- Escrevemos um reticulado $\mathcal{L}$ como $\mathcal{L} = G + \mathcal{L}_1 \oplus ... \oplus \mathcal{L}_k$, com os glue groups projetados $G_1 ... G_k$ e os reticulados componentes $\mathcal{L}_i$ que podem ser decodificados eficientemente.
- Assumindo que temos um grupo de paridade P, tal que todo G_i admite uma injeção de $P : G_i \simeq P_i \subset P$. Usa-se o isomorfismo $\mu_i : G_i \mapsto P_i$.
- Assumimos que G é a soma direta dos seus componentes de glue groups, em conjunto com uma condição de paridade:

$$G = \{(g_1, ..., g_k) \in G_k, tal\ que \sum_i u_i(g_i) = 0\}$$

Decodificando com glue parity

- Primeiro, decodificamos $\pi_i(t)$ em $g + \mathcal{L}_i$, para todo i e $g \in G_i$, usando a estratégia passada.
- Seja, assim, $\tilde{g}_i$ o vetor mais perto de $\pi_i(t)$ em $\mathcal{L}_i + G_i$, podemos fazer a soma direta de todos os $\tilde{g}_i$ para obter $\tilde{g}$, o vetor mais próximo do super-reticulado $\tilde{\mathcal{L}} = (\oplus_i^k G_i) + (\oplus_i^k \mathcal{L}_i)$.

Decodificando com glue parity

- Chamamos a paridade do vetor mais próximo à $\mathcal{L}_i$ de síndrome $s \in P$.
- Se $s = 0$, o vetor mais próximo de $\tilde{\mathcal{L}}$ satisfaz a condição de paridade. Logo, o vetor também é o mais próximo à "t"e pertence à $\mathcal{L}$.
- Quando $s \neq 0$, temos que variar a solução e forca-la em $\mathcal{L}$. Como já decodificamos para todo G_i, só resta descobrir a melhor combinação de cosets, tal que a distância com "t"seja mínima.

Projeção no campo de Galois

Esse decodificador utiliza uma construção do reticulado de Leech sobre campo de Galois $\mathbb{F}_4 = \{0, 1, w, w\}$, conhecida como hexacode, H_6.

- Dado um quarteto binário $a = (a_1, a_2, a_3, a_4) \in \{0, 1\}$, Um carácter "x"é considerado uma projeção de "a", quando ele segue a seguinte relacao $(0, 1, w, w) \cdot a = x$.

Construção técnica do reticulado de Leech

- O reticulado bi-dimensional D_2 (rotação escalada de $\mathbb{Z}_{\not\models}$) é particionado em 16 subsets, rotulados com A_{ijk} e B_{ijk}.
- O array possui somente pontos $type - A$ ou $type - B$.
- Para cada coluna $(A_{i_1 j_1 k_1}, A_{i_1 j_1 k_1})^t$ ou $(B_{i_1 j_1 k_1}, B_{i_1 j_1 k_1})^t$, $i_1 \oplus i_2$ é chamado de h-parity e $k_1 \oplus k_2$ de k-paraty.
- o quarteto (i_1, j_1, i_2, j_2) é interpretado como um carácter $x \in \mathbb{F}_4$

Construção técnica do reticulado de Leech

O reticulado de Leech consiste em todos os arrays $2x6$ de D_2, tal que:

- O array é $type - A$ ou $type - B$.
- Consiste de somente colunas pares ou somente impares.
- Se é tipo $type - A$, a $paridade - k$ geral é par. Senão, é impar.
- Se o array consiste somente de colunas pares, a paridade geral é par, senão é impar.
- A processão das seis colunas é uma palavre-chave de H_6.

Construção com glue theory

A nossa construção do reticulado pode ser descrita em termos de 3 níveis.

Construção com glue theory

Nível 1

Usamos um sub-reticulado de Λ_{24} de índice $2^2 8$. Usamos os cosets deste sub reticulado e representamos eles por $0, g_1, g_2, g_1 + g_2$ formando o glue group $G_{1,2}$. Esse grupo é isomorfo a $\mathbb{Z}_2^2$, que definimos como $\mu_{\mathbb{Z}_2^2} : G_{1,2} \mapsto \mathbb{Z}_2^2$. Definido o set com paridade S_6 para um elemento $z \in \mathbb{Z}_2^2$, alcançando:

$$S_6(z) = \{(g_1, ..., g_6) \in G_{1,2} \oplus ... \oplus G_{1,2} \; tal \; que \; \sum_i \mu_{\mathbb{Z}_2^2}(g_i) = z\}$$

Quando $z = (0, 0)$, S_6 é um grupo. Portanto, podemos definir o reticulado:

$$\mathcal{L}' = \cup_{z \in \mathbb{Z}_2^2}(S_6(z)) + (4D_4)^{\oplus 6}$$

Construção com glue theory

Nível 2

Similarmente, usamos representantes dos cosets como $0, g_3, g_4, g_3 + g_4$ formando o glue group $G_{3,4}$. Esse grupo é isomorfo a $\mathbb{Z}_2^2$, mas escolhemos um isomorfismo levemente diferente: $\mu_{\mathbb{F}_4} : G_{1,2} \mapsto \mathbb{F}_4$. Definido o grupo do hexacode G_{H_6}, alcançando:

$$G_{H_6}(z) = \{(g_1, ..., g_6) \in G_{3,4} \oplus ... \oplus G_{3,4} \\ \textit{tal que } (\mu_{\mathbb{F}4}(g_1), ..., \mu_{\mathbb{F}4}(g_6)) \in H_6\} \quad (1)$$

Construção com glue theory

Nível 2

Podemos definir o reticulado:

$$\mathcal{L}'' = \cup_{z \in \mathbb{Z}_2^2} (G_{H_6} \ x \ S_6(z)) + (4D_4)^{\oplus 6}$$

Como no nível 1, obtemos um sub reticulado de Λ_{24} com índice 2^2, se mantermos a paridade $z = (0, 0)$. Esse reticulado é conhecido como Leech quarter-lattice, Q_{24}.

Construção com glue theory

Nível 3

Similarmente, usamos representantes dos cosets como $0, g_5, g_6, g_5 + g_6$ formando o glue group $G_{5,6}$. Esse grupo é isomorfo a $\mathbb{Z}_2^2$, cujo isomorfismo definimos como $\mu'_{\mathbb{Z}_2^2} : G_{5,6} \mapsto \mathbb{Z}_2^2$. Definido o set de repetição como:

$$S_6^{\perp}(z) = \{(g_1, ..., g_6) \in G_{5,6} \oplus ... \oplus G_{5,6} \; tal \; que \; \mu'_{\mathbb{Z}_2^2}(g_i) = z\}$$

Para um $z \in \mathbb{Z}_2^2$, o tamanho desse set é um. Portanto, podemos definir o reticulado de Leech:

$$\Lambda_{24} = \cup_{z \in \mathbb{Z}_2^2}(S_6^{\perp}(z) \times G_{H_6} \times S_6(z)) + (4D_4)^{\oplus 6}$$

Construção com glue theory

Usando os 5 requisito da construção técnica, podemos restringir como essas colunas, ou cosets, são "colados" juntos.

Pseudocódigos - L24

Decodes the L24 Lattice - *t, *cv, *d

for $coset_h \leftarrow 0, 1$ do ▷ Decodes the Leech lattice by means of four Leech quarter-lattice decoders
    $PrecomputationofH24(t, d_{IJ}, offsets, coset_h)$;
    for $coset_q \leftarrow 0, 1$ do
        $coset \leftarrow 2 * coset_h + coset_q$;
        $q = init_Q 24(d_{ij}, delta_{ij}, offsets, coset_h, coset_q)$;
        $decoder_Q 24(q, cv_q[coset], d_q[coset])$;
    end for
end for
$min_m etric(d_q, cv_q, 4)$; ▷ Calculate the smallest metric and places it in index 0
$*d = d_{q[0]}$
$*cv = cv_{q[0]}$

Pseudocódigos - Q24

Decodes the Q24 Lattice - Q24 *q, *cv, *d

1: Compute the confidence values and penalties;
2: Sort the penalties;
3: Compute the confidence values of each point of q;
4: Finds images of the hexacodewords and calculates its metrics;
5: Calculates the min metric of all the 64 points o "q"and places it at index 0
6: Finalize the output and add the offset bitts of information to "cv" ▷ Outputting 12 offset bits, plus 36 point bits (of which 12 are redundant)

Pseudocódigos - H24

Calculates the d_{ij}s and $delta_{ij}$s

Offsets are wiped to '0'
for $n \leftarrow 0, 11$ do
    for $i \leftarrow 0, 1$ do
        for $j \leftarrow 0, 1$ do
            $decodesubset(t + 2 * n, coset_h, i, j, 0, d_{ij0}, offsets[n]);$
            $decodesubset(t + 2 * n, coset_h, i, j, 1, d_{ij1}, offsets[n]);$
            $d_{ij}[n][i][j] \leftarrow min_v alue(d_{ij0}, d_{ij1});$
            $delta_{ij}[n][i][j] \leftarrow d_{ij1} - d_{ij0};$
        end for
    end for
end for

Pseudocódigos - Decode Subset

Decodes a X_{ijk} subset, for the 32-QAM constellation

1: $offset[2][2] = \{\{0, 0\}, \{4, 4\}\};$
2: $distances[2];$
3: for $index \leftarrow 0, 1$ do
$distances[index] = enclidiandist(t[0], t[1], SCALE * (p[coset_h][i][j][k][0] + offset[index][0])\%Q, SCALE * (p[coset_h][i][j][k][1] + offset[index][1])\%Q);$
4: end for
5: Move the min values of arrays and places it at index 0 for both offset and distance;
6: Set pointers given for offset and distance to its respective index 0 place;

Pseudocódigos - Decode Point

Decodes the output from a decoder to a point in space

$coset_h = 0;$
for $l \leftarrow 0; 11$ do
    $coset_h \hat{=} (cv >> 3 * l) \& 1;$ ▷ determine if A-type or if B-type by counting k-parity ($coset_h ?\ B\ :\ A$)
end for
for $l \leftarrow 0; 5$ do
    $col = cv >> (6 * l);$
    $o1 = cv >> (36 + 2 * l);$
    $o2 = cv >> (36 + 2 * l + 1);$ ▷ offsets

Pseudocódigos - Decode Point

Decodes the output from a decoder to a point in space

$out[4 * l] = p[coset_h][col >> 5\&1][col >> 4\&1][col >> 3\&1][0] + 4 * o1;$ ▷ Line 9

$out[4 * l + 1] = p[coset_h][col >> 5\&1][col >> 4\&1][col >> 3\&1][1] + 4 * o1;$

$out[4 * l + 2] = p[coset_h][col >> 2\&1][col >> 1\&1][col\&1][0] + 4 * o2;$

$out[4 * l + 3] = p[coset_h][col >> 2\&1][col >> 1\&1][col\&1][1] + 4 * o2;$

end for;

for $l = 0; 23$ do

    $out[l] = SCALE * out[l]\%Q;$

end for=0

Suposições

Assumimos que as instruções de adição, subtração, multiplicação e bit-shift são de tempo constante. Isso não é verdade para todas arquiteturas computacionais.

Corretude

Nessa seção, veremos a teoria dos testes que verificam a corretude do código.

Error-correction radius

- Nesse teste, geramos pontos aleatórios do reticulado de Leech, para cada ponto adicionamos um erro aleatório dento do raio de erro do reticulado ($\lambda_1(\Lambda_{24})/2 = 1$).
- O ponto criado é decodificado e verificado contra o ponto inicial.
- Assim, verificamos também a distancia entre o ponto inicial mais o erro e o vetor mais próximo que a decodificação retorna.

Consistência

- Nesse teste, geramos pontos contidos no espaço $\mathbb{R}_n$. Para cada ponto, usa-se dois decodificadores diferentes e verifica-se se eles divergem em resultados.
- Para os pontos que divergem, checa-se a distancia entre os pontos do reticulado encontrados e o inicial.

Segurança por tempo

- Nesse teste, geramos pontos contidos no espaço $\mathbb{R}_n$. Para cada ponto, usa-se o decodificador N vezes.
- Depois, calculamos a media dos tempos e as varianças delas.
- Idealmente, todas as decodificações devem terminar em tempos iguais.

Performance em codificação

- Examinaremos a performance de codificar e decodificar as informações em comparação com os métodos já conhecidos para o reticulado inteiro.
- Para isso compararemos seu tempo médio de uso em cada processo.

Performance em encriptação

Além disso, analisaremos o funcionamento do reticulado de Leech em interação com métodos de encriptação LWE, devido a sua dimensão e características únicas.