Project 2

Xiyue Su

May 27, 2022

1 UOV implementation

Implemented.

2 UOV discussion

2.1 memory size

n	v	F	T	P	signature (bytes)
128	64	9842540	177594	13250558	1517
192	128	26874847	399570	30426309	2273

2.2 probability

By experimenting n from 6 to 40, we find out probability of no solutions to equations is really small, and approximate to 0; except for when $m = n - 1$, the probability is rather huge, around $\frac{1}{4}$.

Here we make a rough estimation: we suppose the matrix is randomly chosen(since v variables are randomly chosen) with uniform distribution of (0,3) tuples. Let's denote no solution to equations as event A, G as the coefficient matrix, G^A as the augmented matrix.

the probability $P(A)$ is estimated as follows :

$$
\begin{aligned}
P(A) = 1 - P(\overline{A}) &= 1 - \sum_{k=0}^{m} P(r(G) = k) P(r(G^A) = k | r(G) = k) \\
&\leq 1 - P(r(G) = m) P(r(G^A) = m | r(G) = m) = P(r(G) \leq m) \\
&\leq \sum_{k=1}^{m} P\left(\text{row } j \in \text{span of first } j-1 \text{ rows} | \text{ first } j-1 \text{ rows linearly ind.}\right) \\
&= \sum_{k=0}^{m-1} \frac{|\mathbb{F}|^k}{|\mathbb{F}|^n} = \frac{1}{|\mathbb{F}|^n} \cdot \frac{|\mathbb{F}|^m - 1}{\mathbb{F} - 1} \\
&\leq |\mathbb{F}|^{m-n} = 4^{-(n-m)}
\end{aligned}
$$

3 Guess-and-solve Attack

3.1 best k

Implemented as **GuessAttack()**. Since we need to balance the trade-off between number of random variables and high-rising complexity of Grobner basis. The best k would possibly be 12, with a running time around 24 min.

3.2 adding field equations

Adding field equations like $x^q - x = 0$ will reduce the time of finding of the Grobner basis greatly, because it constrains the solutions only to $\mathbb{F}_q$. By experimenting, we choose several parameters to find out how fast can new field equations bring to finding a Grobner basis. (Given $\mathbb{F}_7$)

n	v	before	after(s)
15	5	0.01	0.01
18	6	0.09	0.07
21	7	0.11	0.09
24	8	0.57	0.53
27	9	1.26	1.39
30	10	26.45	7.64
60	20	>1200	598.74

As is shown in the table, adding field equations will help reduce the time of finding a basis, as expected. With increasing n, the complexity of finding Grobner basis quickly scales up, which can be reduced by adding field equations.(Notice: magma execution time is affected by many uncertainties.)

4 T' that preserves form (1)

The goal is to map $x_1, x_2, \cdots, x_n$ from oil & vinegar space to oil & vinegar space. A family of matrices T that preserves form (1) maps oil & vinegar polynomials to themselves. Obviously, if we only map the vinegar polynomials to themselves, which still preserves form 1, which is denoted as T_1, then we easily derive

$$T = \begin{pmatrix} A_{m\times m} & 0 \\ 0 & I_{m\times m} \end{pmatrix}$$

where $A_{m\times m}$ is a non-singular matrix. Such is why the mapping T' is not unique, but a family of mappings that makes $\mathcal{P} \circ \mathcal{T}'$ oil & vinegar polynomials. To be exact, if we calculate a pair of private keys $(\mathcal{F}, \mathcal{T}')$, $T'^t F_k T' = (T^{-1}T')^t (T^t F_k T')(T^{-1}T')$, hence apparently we find another pair of private keys, i.e. $(\mathcal{F} \circ \mathcal{T}, \mathcal{T}^{-1} \cdot \mathcal{T}')$.

Side question: yes. We can make use of the central map F_k, namely performing elementary transformations on those polynomials without oil variables. By deleting those rows, we finally change them into a form below:

$$T = \begin{pmatrix} A_{r\times r} & \cdots & B \\ \cdots & 0_{(m-r)\times(m-r)} & \\ C & \cdots & 0_{m\times m} \end{pmatrix}$$

which reduces the actual memory size of matrix T.

5 Kipnis-shamir Attack

5.1 explanation

Explanation of form (3):

Assuming F_k is non-singular, F_k has a lower-right zero block that looks like

$$\begin{pmatrix} A_k & B_k \\ C_k & 0 \end{pmatrix}$$

one observation is that when F_k is multiplied by a column vector in oil subspace $(0, \cdots, 0, D_k)^T$, the result is a column vector whose second half is zero. Plus, it is a bilinear map that maps a subspace of dimension m to a subspace of dimension m. In this way, F_k maps the oil subspace into the vinegar subspace, in other words, it maps the oil subspace into the vinegar subspace, i.e. $F_k \cdot \mathcal{O} = \mathcal{V}$. Since all the vinegar subspace is in the range of this mapping, F_k^{-1} maps the vinegar subspace back into the oil subspace, i.e. $F_k^{-1} \cdot \mathcal{V} = \mathcal{O}$.

Explanation of form (4):

now we have P_k and $T^t \cdot F_k \cdot T$ both as symmetric matrices, then for every component $a_{i,j}, b_{i,j}$ in P_k and $T^t \cdot F_k \cdot T$ respectively :

(I) when $i = j$, $a_{i,j}, b_{i,i}$ are the coefficients of x_i^2 in $p_k(x)$, so $a_{i,i} = b_{i,i}$.

(II) when $i \neq j$, $a_{i,j}, b_{i,i}$ are the coefficients of x_i^2 in $p_k(x)$, and $a_{i,j} = a_{j,i}$, $b_{i,j} = b_{j,i}$ so $a_{i,j} = b_{i,j}$.

As a consequence, we have $P_k = T^t \cdot F_k \cdot T$.

5.2 a counter example

suppose in $\mathbb{F}_7$, we have

$$F_1 = \begin{pmatrix} 1 & 2 \\ 2 & 0 \end{pmatrix}, F_2 = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}, T = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

$$T^t F_1 T = \begin{pmatrix} 1 & 3 \\ 3 & 4 \end{pmatrix}, T^t F_2 T = \begin{pmatrix} 2 & 3 \\ 3 & 2 \end{pmatrix}$$

we have an asymmetric public key matrix which says:

$$P_1 = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix}, P_2 = \begin{pmatrix} 2 & 0 \\ 6 & 2 \end{pmatrix}, P_1^{-1} P_2 = \begin{pmatrix} 0 & 4 \\ 5 & 4 \end{pmatrix}, \mathcal{O} = \begin{pmatrix} 0 \\ a \end{pmatrix}, a \in \mathbb{F}_7,$$

$$T^{-1}(\mathcal{O}) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ a \end{pmatrix} = \begin{pmatrix} 0 \\ a \end{pmatrix}$$

, so

$$P_1^{-1}P_2 \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \end{pmatrix} \notin T^{-1}(\mathcal{O})$$

6 KS Attack

6.1 explanation

The attack goes as follows: to find the invariant subspace $\mathcal{T}^{-1}(\mathcal{O})$, we use characteristic polynomials. First we find the oil subspace, then we expand the oil subspace to a dimension of $2m$, from which we can recover a valid T.

To find the oil subspace, we have two important observations: (1)oil subspace is the common eigenspace of all matrices $P_i^{-1}P_j$; (2)$kernel(P'(B))$ is an eigenspace of B given any polynomial P'. To find a usable B, we sample linear combination of $P_i^{-1}P_j$ randomly. Since $F_i^{-1}F_j$ and $P_i^{-1}P_j$ are similar and $F_i^{-1}F_j$ is with zeros in the top right part, the characteristic polynomial of $P_i^{-1}P_j$ can be divided to two m-degree polynomials $P_1(x), P_2(x)$.

Note that $P_1(x)$ or $P_2(x)$ is irreducible with high probability, so we can find the oil space, which is an eigenspace of $P_i^{-1}P_j$ easily. Now that we have the transformation $\mathcal{T}^{-1}$ by expanding the oil subspace, we derive T from obtaining its inverse matrix.

Reason why I choose this particular method: the linearization one looks too abstract.

6.2 Implementation

Implemented.

6.3 Circumvent Non-invertible Matrices

There are two possible solutions:

a) the linear combination of $P_1, P_2, \cdots, P_m$ can be invertible, in such case, we can simply substitute P_i and P_j by the random linear combinations of $P_1, P_2, \cdots, P_m$.

b) another scenario is when certain variables do not appear at all. So to delete one or more of the variables to zero until P_i is invertible. Therefore, m is reduced by one or more(making sure it is even), but it also reduces the dimension of the secret linear subspace by one or more and makes the attack less powerful.

7 intention of S

The intention of S is to mix those polynomials and makes the structure of rainbow layers less visible, compared to the former T whose intention is to mix the oil & vinegar variables. Suppose we know the variable numbers of the first layer v_1, m_1; without linear map S, we can apply Kipnis-Shamir attack to the first layer given that $n_1 \geq 2m_1$ holds water with great possibility. Now that we obtain an $(v_1 + o_1) \times (v_1 + o_1)$ equivalent key T' and F', resulting in a system of $f_{v_1} + ... + f_{v_2}$ equations in the unknown variables $x_1, x_2, \cdots, x_{v_2}$ which can be solved easily. Then these variables can result in a new set of linear equations $f_{v_{v2}+1} + ... + f_n$ in $x_{v_2+1}, \cdots, x_n$ unknowns. Hence we can use such attacks to forge the signature without knowing the whole private key.

No, it does not need to be a fully random invertible map. An immature insight would be we extract only one or more polynomial from each layer(with fixed positions) and mix them together, which reduces the size of S by half but does not affect security(at least we suppose). If we only mix half of polynomials from each layer, then the corresponding S looks like

$$\begin{pmatrix} r(F_q) & \cdots & 0 & \cdots & r(F_q) & 0 \\ 0 & r(F_q) & \cdots & 0 & \cdots & r(F_q) \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix}$$

8 Rainbow implementation

Implemented.

9 Rainbow test

Implemented as **Task9a()**.

References

[1] Kipnis A, Shamir A. Cryptanalysis of the oil and vinegar signature scheme[C]//Annual international cryptology conference. Springer, Berlin, Heidelberg, 1998: 257-266.

[2] Beullens W. Improved cryptanalysis of UOV and rainbow[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Cham, 2021: 348-373.