Lattice Signatures, Commitments, and Zero-Knowledge Proofs

Rick

2020 年 10 月 23 日

1 Lattice Signatures, Commitments, and Zero-Knowledge Proofs

The problem here is when trying to prove multiple SIS-like equations, this is achieved by using only one column of C, which makes a linear combination of the secret vectors (each one corresponding to an equation). Is that secure and soundness-keeping ? Anyway, here is the complete talk content.

First Vadim reviewed the Schnorr proof in the discrete log world. He then moved on to the lattice world, where hard problems can be summarized as given $A, b \in \mathbb{Z}_q^{n\times m}, \mathbb{Z}_q^n$, find a small vector s such that $As = b$. The LWE problem can be viewed in this way, *in my opinion*, by merging the e vector as an additional row in A. He also mentioned the parameter choice for different applications.

表 1: Typical Modulus for Different Applications

Application	Modulus
Encryption	$\approx 2^{12}$
Signature	$\approx 2^{20}$
FHE	$> 2^{30}$

Plugging this Ax=b into the Schnorr's framework would get a scheme that has correctness, HVZK, but no soundness. This is due to the knowledge extractor cannot extract a *small* secret x from $z - z'/c - c'$, where c is a challenge scalar. Besides, he mentioned the soundness error is too small – 2^{-20} or the challenge space is too small. There is nevertheless some doubt about this part, though, where does the soundness error come from? Is that the case where $c = 0$? If so, why not let the challenge space to be non-zero and avoid this problem once and for all?

Vadim then proposed another modified scheme that address the soundness problem. The way to do this is by making y a binary vector and the challenge c also binary. In this way, the verifier can actually verify that $||s|| \leq \sqrt{n}$ by checking the response z have -2 to 2 coordinates. The challenge space is further reduced to $\{0, 1\}$, which means more repetition. But the biggest challenge comes from the zero-knowledge property, and that is where *rejection sampling* comes into play.

Notice that in the aforementioned scheme, the response z_i's are basically random binary vectors with mean $(1/2)^n + c_i \cdot s$, which results different distribution with 0/1 choices of c_i. This is a problem since proof of HVZK will not work since s is unknown. The simple modification then proceeds only when the sent z coordinates do not reveal the respective coordinates of s.

In the example Vadim chose $y \in \{0,1,2,3,4,5\}$, and thus when $z_i = 0 \vee z_i = 6$, the response is rejected. Ergo, (simplified) rejection sampling.

Choice of parameter: when we set $b = mk$ where m is the dimension and k is the repetition count, the success rate is $(1-1/mk)^{mk} \approx 1/e$, a constant. Vadim chose to use this setting throughout the talk.

A extractor can also extract the secret $s \in \{-mk, \ldots, mk\}$ from a prover that can answer more than one distinct query. I wrote "but still no HVZK" on my notebook, probably regarding the case pointed out in the slides that when the third message $z = \perp$, the first messages cannot be simulated from the lack of s. This is also complimented with a comment "cannot simulate it but does not matter", which I think correspond to the next slide showing we can replace the w's by an RO output $r = H(w_1, \ldots, w_k)$ – a simple commitment to the commitments. And then since the verified relation is linear, the verifier can reconstruct w_i's from the z_i's and check valid opening. In this case, if the final message is $\perp$, just send uniform randomness in the first step. In this protocol, $k = 128 \sim 256$.

Vadim then listed three possible optimizations that could make this basic scheme practically useful.

1. Proof size for 1 equation $\approx$ proof size for many equations (amortization with log growth)
2. Working over polynomial rings instead of $\mathbb{Z}_q$ allows for "1 shot" approximate proofs – digital signatures)
3. More advanced ZK techniques allow for almost 1-shot exact proofs (i.e. prove that coefficients of s are in $\{0,1\}$)

2 Amortized Proof

The first technique to use is amortizing many equations over one shared A. The setup is j equations. The verifier samples a matrix $C \in \{0,1\}^{j \times k}$ and checks that $T \cdot C + W = A \cdot Z$, where T is the target matrix. A simple calculation reveals that the rejection rate (when coefficients of Z is smaller than j or larger than mkj) still converges to $1/e$, and notice that the dimension of Z is $m \times k$ which does not depend on j.

The extractor for soundness proof extracts one column of S at one time. We do this by setting the two challenges C, C' only different at the i^{th} column we want to extract ($i \in \{1, \ldots, j\}$).

This means that we can acquire the following equation:

$$T \cdot (C - C') = A \cdot (Z - Z'),$$

where any column of $Z - Z'$ is in $[j - mkj, -j + mkj]$. Using any non-zero column of $C - C'$ would get will complete the proof.

3 Working over Larger Rings

The paradox with working over rings is that we want the challenge space to be large, but with smaller value so that the extracted witness would be "short". In order to do this, Vadim says we can *work over larger rings that have many small elements*. One example is polynomial ring $\mathbb{Z}_{17}[X]/(X^4+1)$. One fact about these types of rings is that reduction modulo x^4+1 does not increase coefficient size very much (at least l_1 norm), so they are good for crypto. In the ring case, there are several advantages against the plain version:

- Dimension and sample number can be divided by d for the same conjectured hardness;
- There are now many 0/1-coefficient elements in the ring (2^d).

Vadim then presented a analog scheme over ring $\mathbb{Z}_q[X]/(X^d+1)$. Throughout, he used the notation $[B]$ to denote ring elements with coefficient value from $-B$ to B. The challenge space is $[1]$, the space of y is $[\beta + d]^m$, and the space for z is $[\beta]^m$. The reason for this (an intuitive idea, without further enquiry to ring multiplication details) is that when modulo reduction occurs, one -1 appears replacing the original one. And therefore a binary s times a binary c can at most incur d bounded term, making the space reduce by d on both sides.

There is one big problem, though, being the quotient of two small polynomials does not necessarily be small. This would make the knowledge extraction process fail, but Vadim said in the slides that this is good enough for commitment and digital signature.

Applying the Fiat-Shamir transform to the scheme would yield a digital signature scheme. More specifically, the signing key is (A, s), the verification key is (A, t).

- The signing algorithm first samples short $y \in [\beta + d]^m$, compute $w = A \cdot y$, $c = H(w, \mu)$, and $z = s \cdot c + y$. If $z \notin [\beta]^m$, the algorithm rewinds. The signature for μ is $\sigma = (c, z)$.
- The verification algorithm accepts if $z \in [\beta]^m$ and $H(A \cdot z - t \cdot c, \mu) = c$.

The proof sketch for EUF-CMA is as follows: first program the RO such that whenever $H(w,\mu)$ is queried, samples small z and solves c for $A\cdot z = t\cdot c + w$. Answer the signing query by using HVZK proof – basically the aforementioned process. Whenever a forgery is submitted, the simulator is able to find $A\cdot(z-z') = t\cdot(c-c')$, where (c',z') refers to the forged signature. And this constitutes as a solution to SIS problem with matrix $\bar{A} = [A|t]$, and the solution being $\bar{s} = [z-z'|-c+c']$.

Several additional points are also mentioned in the talk.

- Can also get homomorphic commitment schemes with ZK opening (used in the next section)
- Digital signatures can be made very efficient (pk $\approx$ 1.5KB, sig $\approx$ 2.7KB)
- Other ways to do "rejection sampling". Can save *square root* in the dimension in the output norm by using gaussians

4 Proving Knowledge of the Exact Relation $As = t$

Previously, the range of s extracted is not satisfying, we see if that can be improved in this part of the talk. In this part of the talk, Vadim uses $\odot$ to denote component-wise multiplication. Homomorphic commitment with zero-knowledge proof is also a required component for this kind of proof. The four properties required are 1) hiding, 2) binding, 3) homomorphic addition, and 4) practical ZKPoK of homomorphic relations. The observation here is that in order to prove the binary property of s, we only have to prove that $s\odot(s-\mathbf{1}) = \mathbf{0}$. This is enabled by a closely related z:

$$z = y + s\cdot c \quad z\odot(z-c\cdot\mathbf{1}) = \underbrace{y\odot y}_{h} + c\cdot\underbrace{(2s-\mathbf{1})\odot y}_{g} + c^2\cdot\underbrace{s\odot(s-\mathbf{1})}_{0}$$

This can be proved using (additive) homomorphism of commitments and ZKPoKs. Altogether, we want to prove three equations:

1. $A\cdot z = w + c\cdot t$, this is done in plaintext;
2. $z = y + c\cdot s$, this is done using homomorphism and ZKPoK;
3. $z\odot(z-c\cdot\mathbf{1}) = h + c\cdot g$, this is done using homomorphism and ZKPoK.

2&3 implies s is binary, and 1&2 implies (with rewinding) that $As = t$, proving the relation we want. Moreover, the Schwartz-Zippel lemma implies that by using a random $c \in \mathbb{Z}_p$, the probability that a non-binary s passes the check is at most $2/p$. The overall scheme proceeds as follows: in the first step, besides sending $w = A \cdot y$, the prover also send commitments of s, y, g, h, denoted using the capital letters. The challenge c is a random element from $\mathbb{Z}_q$. In the final step, the prover sends $z = y + c \cdot s$, in addition to the proof of valid opening of Y, S, G, H, and proof of relation

$$z = y + c \cdot sz \odot (z - \mathbf{1}) = c \cdot g + h$$

The S-Z Lemma states that (*in my understanding*) when z is statistically independent with c, the probability that $z \odot (z - \mathbf{1}) = c \cdot g + h$ for non-binary s (i.e. a non-zero degree-two polynomial over $\mathbb{Z}_p$) is at most $2/p$, which ensures soundness of the 0/1 proof.

The requirement for c is that (listed in the slides):

- Must be in a field with 0/1 coefficients;
- Must be compatible with multiplication with s;
- Must be compatible with multiplication with S (the commitment).

I think the final conclusion is that Z_p is good enough?

Vadim then moved on to homomorphic commitments. He first introduced Lattice-based commitment in [BLS19] and [YAZXYW19]. They have the required properties of homomorphism and efficient ZKPoK. In these schemes, there are two public matrices B_1, B_2, and commitment to message m is $(B_1 \cdot r, B_2 \cdot r + m)$. The ZKPoK property is proved in [BDLOP18], I believe? I did not left much note in the notebook for this part.

The SISRS (SIS relation using Reed-Solomon code) is a work in progress, and shows sublinear asymptotic property when proving several related SIS relations (sharing the same A). In the last few minutes of this talk he introduced the scheme, but with much confusion on my side. Here they used a linear code as commitment, and used PCP to ensure hiding property. In addition, if verifier is given $z = y + c \cdot s$ and $r_z = r_y + c \cdot r_s$ (the commitment for m is $G \cdot [m|r]^T$ where G is the generator matrix), then there is a lemma that states if $c \cdot S + Y$ is a codeword then with probability $1 - 1/p$, S and Y are close to codewords and $c \cdot \mathsf{Decode}(S) + \mathsf{Decode}(Y) = z$, which proves soundness?

He then introduced the use of Merkle tree in the proof, specifically, if many related instances are being proved, then the hash tree can be reused, bring down communication. Vadim himself

mentioned in the slides that this part uses techniques that are quite involved, so I guess I will get a better understanding after reading the paper.

Vadim left a few parting words at the end of the talk.

- Lattice based signatures over polynomial rings are already practical (See a recent tutorial on my IBM web page for how to construct lattice based encryptions / signatures with many optimizations)
- Work on exact zero knowledge proofs for SIS is just beginning
- Can we have use another commitment scheme in the generic proof approach?
- Maybe have a lattice based (instead of code based) PCP