From: Perlner, Ray A. (Fed)
To: (b) (6)
Subject: RE: GeMSS figures in Asiacrypt paper
Date: Monday, June 15, 2020 5:24:00 PM

One thing to note, playing with some examples on paper: The MinRank problem arising from these systems does not have a unique solution (up to scalar multiplication.) e.g., from a square map over GF3, we can get matrices

A = ((1 0)(0 2)) and B = ((0 1)(1 0)). To get a rank-1 linear combination of these two matrices over the extension field, defined by appending i =sqrt(2), there are multiple linearly independent solutions: In particular A+iB and A-iB are both solutions.

From: Daniel Smith (b) (6)
Sent: Monday, June 15, 2020 5:01 PM
To: Perlner, Ray A. (Fed) <ray.perlner@nist.gov>
Subject: Re: GeMSS figures in Asiacrypt paper

If it helps, I have an old magma script with HFE- programmed into it and I ran the support minors thing on it (my version with the variables of C intact) with one of the linear variables set to 1. I did not get a Grobner basis for the ideal by running the XL-like code, because the unique reduced GB should probably have coefficients from the extension field which are impossible to attain this way, but running F4 on the resulting system, I didn't have to go to a higher degree to resolve the ideal. I think that some of the lower degree terms made more relations with the higher degree terms to help reduce everything to a solution.

This is what made me ask. I see that our paper doesn't address this at all but just seems to apply the numbers of this method without any justification for how it completes the calculation, since it can't occur with XL, it seems. So there is something that I'm missing in here.

Cheers!

On Mon, Jun 15, 2020 at 4:52 PM Daniel Smith (b) (6) wrote:

> There is a problem with language here. The support minors method applied directly to GeMSS produces a system of bidegree (b,r) in the linear variables and the variables of C=(K| I), where we have a restricted monomial set [the component from variables in C is always in the form of a maximal minor of C]. (I think that modeling it this way is strictly better than replacing the minors with new variables so that we would be of bi-degree (b,1) in the linear variables and the minors of C. The reason is that there may be instances where the rank is such that we get more efficiency by targeting something like (b,r+k) in this variable set and we would have many fewer monomials this way than going to something like (b,2) in the linear-minors set. It just requires a much more difficult analysis of how many monomials we are linearizing over.) This system of equations uses variables whose values like in GF(2^n), but the coefficients are all in GF(2).

On the other hand, we can choose a representation of GF(2^n) over GF(2) and use the fact that the operations are over GF(2^n) to generate a much larger set of equations in a much larger variable set, so that the entire system is modeled over GF(2).

I don't know which of these two you are talking about when you say to linearly solve over GF(2).

If you are talking about the latter, then I don't think that there is an extra step, I think that you just have the solution, so that makes me inclined to think that you are stating the former. But in that case, I go back to my original question which is how we got the complexities we obtained to begin with. How were we finishing the attack to get the solution over GF(2^n)?

CHeers!

On Mon, Jun 15, 2020 at 4:13 PM Perlner, Ray A. (Fed) <ray.perlner@nist.gov> wrote:

> Hm. The things we're solving linearly for are actually reasonably high degree in the variables we're really solving for (i.e. they're products of r x r minors and degree-b monomials, which in turn have degree r+b in the things that are supposed to be in the extension field.) This makes me wonder whether we can solve linearly over GF2, and then use the solution to solve for one of the variables of interest in a univariate system (or in a much cheaper GB calculation). Like maybe you could set up an equation in the ratio x1/x2 by looking at all the values for monomials generated by x1, x2, and one of the minors.
>
> If something like that works, then the MinRank attack is actually cheaper than we had been assuming.
>
> **From:** Daniel Smith (b) (6)
> **Sent:** Monday, June 15, 2020 2:17 PM
> **To:** Perlner, Ray A. (Fed) <ray.perlner@nist.gov>
> **Subject:** Re: GeMSS figures in Asiacrypt paper
>
> I don't think that it is a big deal. I think that we should have all of the information necessary to resolve the ideal, but we have to have some equations of positive degree in the variables to be able to achieve solutions in an extension field.
>
> On Mon, Jun 15, 2020 at 2:14 PM Daniel Smith (b) (6) wrote:
>
> > I'm just wondering how we are justifying getting a solution in GF(2^n) when the equations have coefficients in GF(2). There must be an additional Grobner basis step, because we can't get a solution outside of GF(2) linearly.
> >
> > On Mon, Jun 15, 2020 at 2:12 PM Perlner, Ray A. (Fed) <ray.perlner@nist.gov> wrote:
> >
> > > I think I used 2 * 3 * n * <matrix dimension> *<number of potentially nonzero entries in the matrix>. Reasoning was that Wiedemann used 3 *<matrix dimension> matrix vector

multiplications, where the matrix was over the base field and the vector was over the extension field. Let me know if this is wrong.

Cheers,
Ray

From: Daniel Smith (b) (6)
Sent: Monday, June 15, 2020 1:59 PM
To: Perlner, Ray A. (Fed) <ray.perlner@nist.gov>
Subject: GeMSS figures in Asiacrypt paper

Hi, Ray,

How did we get our complexity numbers for GeMSS in the Asiacrypt paper? The equations are over GF(2), but the solution needs to be over GF(2^n) (at least when we express all of the matrices the natural way). So when we solve linearly, we don't get coefficients in the extension field. So I'm just curious where we are getting the numbers from.

Cheers,
Daniel