From: (b) (6)
To: Moody, Dustin (Fed)
Subject: Re: Simple Matrix
Date: Monday, December 2, 2019 11:05:42 PM

Awesome, thanks!

I am going to rest for a bit. Then I can add it. Looks good.

On Mon, Dec 2, 2019 at 23:00 Moody, Dustin (Fed) <dustin.moody@nist.gov> wrote:

> Here's the data for Experiment 1:
>
> q=3
> 2.45915697674419 13760
> [q,s,u,v,r,n]= [3, 3, 4, 4, 5, 20]
> [mean,std dev]= [2.45915697674419, 1.83850977513046]
>
> 6.59226305609284 2585
> [q,s,u,v,r,n]= [3, 4, 5, 5, 6, 30]
> [mean,std dev]= [6.59226305609284, 6.32908922808728]
>
> 19.0932779456193 2648
> [q,s,u,v,r,n]= [3, 5, 6, 6, 7, 42]
> [mean,std dev]= [19.0932779456193, 18.5086585721677]
>
> 53.8666666666667 1305
> [q,s,u,v,r,n]= [3, 6, 7, 7, 8, 56]
> [mean,std dev]= [53.8666666666667, 50.2200892269537]
>
> 173.047438330171 527
> [q,s,u,v,r,n]= [3, 7, 8, 8, 9, 72]
> [mean,std dev]= [173.047438330171, 167.120000908034]
>
> 551.388429752066 242
> [q,s,u,v,r,n]= [3, 8, 9, 9, 10, 90]
> [mean,std dev]= [551.388429752066, 690.475521656382]
>
> q=5
> 4.27184071515644 2461
> [q,s,u,v,r,n]= [5, 3, 4, 4, 5, 20]
> [mean,std dev]= [4.27184071515644, 3.58696480167274]

20.5344086021505 930
[q,s,u,v,r,n]= [5, 4, 5, 5, 6, 30]
[mean,std dev]= [20.5344086021505, 20.8804428695659]

99.3520179372197 446
[q,s,u,v,r,n]= [5, 5, 6, 6, 7, 42]
[mean,std dev]= [99.3520179372197, 93.5903545061826]

520.798927613941 373
[q,s,u,v,r,n]= [5, 6, 7, 7, 8, 56]
[mean,std dev]= [520.798927613941, 512.776486970926]

q=7
6.51416666666667 2400
[q,s,u,v,r,n]= [7, 3, 4, 4, 5, 20]
[mean,std dev]= [6.51416666666667, 6.53216999351918]

40.5374429223744 2190
[q,s,u,v,r,n]= [7, 4, 5, 5, 6, 30]
[mean,std dev]= [40.5374429223744, 41.8938620048050]

281.457910447761 1675
[q,s,u,v,r,n]= [7, 5, 6, 6, 7, 42]
[mean,std dev]= [281.457910447761, 280.626800758365]

1873.02987421384 636
[q,s,u,v,r,n]= [7, 6, 7, 7, 8, 56]
[mean,std dev]= [1873.02987421384, 1884.11896093337]

q=11
9.75156950672646 1115
[q,s,u,v,r,n]= [11, 3, 4, 4, 5, 20]
[mean,std dev]= [9.75156950672646, 8.57400902949422]

113.625698324022 895
[q,s,u,v,r,n]= [11, 4, 5, 5, 6, 30]
[mean,std dev]= [113.625698324022, 120.199971379439]

1318.79750778816 321
[q,s,u,v,r,n]= [11, 5, 6, 6, 7, 42]

[mean,std dev]= [1318.79750778816, 1337.03371993520]

q=13
11.7200772200772 1036
[q,s,u,v,r,n]= [13, 3, 4, 4, 5, 20]
[mean,std dev]= [11.7200772200772, 10.7997505067721]

157.689292543021 1046
[q,s,u,v,r,n]= [13, 4, 5, 5, 6, 30]
[mean,std dev]= [157.689292543021, 153.099254855776]

2026.73991031390 223
[q,s,u,v,r,n]= [13, 5, 6, 6, 7, 42]
[mean,std dev]= [2026.73991031390, 1832.88349220909]

From: Daniel Smith (b) (6)
Sent: Monday, December 2, 2019 9:39 AM

To: Moody, Dustin (Fed) <dustin.moody@nist.gov>
Subject: Re: Simple Matrix

Hi, Dustin,

These data match our expected values very closely. In fact, they are very slightly better than our estimates, so that is good for the attack. They are very close, though. For example, the q=13 and s=3 data point matches the theoretical value of about 151. The way to get the theoretical value is to multiply the inverse of the probability that a 2s x 2s matrix over GF(q) is singular by the inverse of the probability in section 4.2. (Do you have access to the overleaf document?) These values are really tight, so that's good. Thanks!

Cheers,
Daniel

On Mon, Dec 2, 2019 at 8:48 AM Moody, Dustin (Fed) <dustin.moody@nist.gov> wrote:

> Daniel,
>
> I have finished running a bunch of the experiments for Experiment 2. The data is below. I have a busy day today, but I'm going to try and do more of Experiment 1. And then I will try and add in even q and run both experiments.
>
> Dustin
>
> It seems to hold that the mean (i.e. expected number of trials) is about q*(q-1). The same is true for the standard deviation. Increasing s seems to multiply by a factor of q. Thus

perhaps it is q^(s-2)*(q-1).

5.89217919514047 1317 iterations
[q,s,u,v,r,n]= [3, 3, 4, 4, 5, 19]
[mean,std dev]= [5.89217919514047, 5.11959474359889]

16.3192526401300 1231 iterations
[q,s,u,v,r,n]= [3, 4, 5, 5, 6, 29]
[mean,std dev]= [16.3192526401300, 15.6841677126768]

47.0008110300081 1233 iterations
[q,s,u,v,r,n]= [3, 5, 6, 6, 7, 41]
[mean,std dev]= [47.0008110300081, 44.7181196676965]

138.925619834711 605 iterations
[q,s,u,v,r,n]= [3, 6, 7, 7, 8, 55]
[mean,std dev]= [138.925619834711, 135.977876395491]

q=5
17.8899341486359 1063 iterations
[q,s,u,v,r,n]= [5, 3, 4, 4, 5, 19]
[mean,std dev]= [17.8899341486359, 17.1927732566781]

86.1897196261682 1070 iterations
[q,s,u,v,r,n]= [5, 4, 5, 5, 6, 29]
[mean,std dev]= [86.1897196261682, 80.6311427746200]

500.284615384615 260 iterations
[q,s,u,v,r,n]= [5, 5, 6, 6, 7, 41]
[mean,std dev]= [500.284615384615, 510.330035306707]

2137.33846153846 130 iterations
[q,s,u,v,r,n]= [5, 6, 7, 7, 8, 55]
[mean,std dev]= [2137.33846153846, 2269.52575230371]

q=7
36.5502901353965 1034 iterations
[q,s,u,v,r,n]= [7, 3, 4, 4, 5, 19]
[mean,std dev]= [36.5502901353965, 35.4661651737622]

277.723136495644 1033 iterations

[q,s,u,v,r,n]= [7, 4, 5, 5, 6, 29]
[mean,std dev]= [277.723136495644, 275.766737347640]

2092.32692307692 104 iterations
[q,s,u,v,r,n]= [7, 5, 6, 6, 7, 41]
[mean,std dev]= [2092.32692307692, 2046.08949515429]

q=11
100.421209117939 1009 iterations
[q,s,u,v,r,n]= [11, 3, 4, 4, 5, 19]
[mean,std dev]= [100.421209117939, 94.7370943702996]

1174.99335548173 301 iterations
[q,s,u,v,r,n]= [11, 4, 5, 5, 6, 29]
[mean,std dev]= [1174.99335548173, 1289.92022232218]

q=13
148.352589641434 502 iterations
[q,s,u,v,r,n]= [13, 3, 4, 4, 5, 19]
[mean,std dev]= [148.352589641434, 157.878086518874]

1855.37313432836 201iterations
[q,s,u,v,r,n]= [13, 4, 5, 5, 6, 29]
[mean,std dev]= [1855.37313432836, 1928.81276827862]

From: Daniel Smith (b) (6)
Sent: Wednesday, November 27, 2019 10:23 AM
To: Moody, Dustin (Fed) <dustin.moody@nist.gov>
Subject: Re: Simple Matrix

I would expect it to be higher, but I'm a bit surprised it is so high. I don't really have a well-formed expectation, though.

On Wed, Nov 27, 2019 at 10:13 AM Moody, Dustin (Fed) <dustin.moody@nist.gov> wrote:

> I'm running the experiments now.
>
> FYI - on experiment 2 (n=ru-1), the std deviation appears to be roughly the same as the number of trials, e.g. q(q-1). I don't know if that's expected or not.
>
> ---
>
> **From:** Daniel Smith (b) (6)
> **Sent:** Wednesday, November 27, 2019 7:43 AM
> **To:** Moody, Dustin (Fed) <dustin.moody@nist.gov>

Subject: Re: Simple Matrix

I would say we need more of the last two types you did, (1) with u=v=s+1, r=s+2 n=r*u, and (2) the same but with n=r*u-1 and throw away two equations. It would be nice to get data as large as possible for these settings to support our argument of the behavior. It might also be nice to collect the data points as well so we can say something about the consistency, the standard deviation or something. Also, I suspect that the standard deviation should be fairly high for these small instances, but we might still get useful data even if we can only do 100 experiments at a certain size. So you can save time by doing fewer than 2000 at once. Another thing is that it would be good to see this for non-prime fields. I don't know how easy that is for you to program, but doing this for q=16 would be a little more in line with what we want. And then of course there is increasing s. Whatever you can accomplish will help.

Thanks!

Cheers,
Daniel

On Wed, Nov 27, 2019 at 7:16 AM Moody, Dustin (Fed) <dustin.moody@nist.gov> wrote:

> Tell me the parameters you want experiments for, and I'll run them.

From: Daniel Smith (b) (6)
Sent: Tuesday, November 26, 2019 2:29 PM
To: Moody, Dustin (Fed) <dustin.moody@nist.gov>
Cc: Perlner, Ray A. (Fed) <ray.perlner@nist.gov>; Apon, Daniel C. (Fed) <daniel.apon@nist.gov>; Javier Alfonso Verbel Herrera <javerbelh@unal.edu.co>
Subject: Re: Simple Matrix

good! As expected. Now we just need bigger and bigger experiments of these sorts and we will have good data.

Thanks!

On Tue, Nov 26, 2019 at 12:12 PM Moody, Dustin (Fed) <dustin.moody@nist.gov> wrote:

> Daniel suggested another experiment:
>
> "I have another experiment that would be good to run
>
> instead of choosing n=r*u, let it be r*u-1 and then still choose r two bigger than s. but then the solving strategy will be different. You would throw away two of the equations you generate. (Are you mixing the equations with a linear map?) In this case I think that the probability should be something like 1/(q(q-1) again."

Doing this, with r=u=v=4, s=3, n=r*u-1=15 yields

q=3 yields 5.6

q=5 gives 17.8

q=7 gives 37.2

q=11 gives 104.1

This basically fits the 1/q(q-1) model.

Dustin

From: Daniel Smith (b) (6)
Sent: Monday, November 25, 2019 3:24 PM
To: Moody, Dustin (Fed) <dustin.moody@nist.gov>
Cc: Perlner, Ray A. (Fed) <ray.perlner@nist.gov>; Apon, Daniel C. (Fed) <daniel.apon@nist.gov>; Javier Alfonso Verbel Herrera <javerbelh@unal.edu.co>
Subject: Re: Simple Matrix

So I guess that the next step is to try it with even bigger r and see if we can get a probability greater than 1!

On Mon, Nov 25, 2019 at 3:17 PM Moody, Dustin (Fed) <dustin.moody@nist.gov> wrote:

> Also, for r=u=v=4, s=3, n=r*u=16 and q=13, the average was 161 (recall 13*12=156). Perhaps with more than 2000 trials the average would be closer.
>
> Re-running with r=5, u=v=4, n=r*u=20, and several values of q:
>
> For q = 3, the average is 3

For q = 5, the average is 4.4

For q = 7, the average is 6.7

For q = 11, the average is 10.7

For q = 13, the average is 12

For q = 17, the average is 15.7

This is approximately 1/(q-1), which is what Daniel predicted.

From: Daniel Smith (b) (6)
Sent: Monday, November 25, 2019 3:02 PM
To: Moody, Dustin (Fed) <dustin.moody@nist.gov>
Cc: Perlner, Ray A. (Fed) <ray.perlner@nist.gov>; Apon, Daniel C. (Fed) <daniel.apon@nist.gov>; Javier Alfonso Verbel Herrera <javerbelh@unal.edu.co>
Subject: Re: Simple Matrix

That is excellent. Can you do it with r one bigger and everything else the same? Thanks!

On Mon, Nov 25, 2019 at 2:34 PM Moody, Dustin (Fed) <dustin.moody@nist.gov> wrote:

> I think I have the code up and running. Using r=u=v=4, s=3, and n=r*u=16. Ray says that over Fq, the probability should be about 1/(q*(q-1)), hence q*(q-1) trials needed.
>
> For q=3, the average was 8.
>
> For q=5, the average is 20.
>
> For q=7, the average is 42.
>
> For q=11, the average is 108.
>
> (running the experiment 2000 times)
>
> Seems to fit pretty well. Any other experiments you want ran?

From: Daniel Smith (b) (6)
Sent: Monday, November 25, 2019 1:57 PM
To: Perlner, Ray A. (Fed) <ray.perlner@nist.gov>
Cc: Apon, Daniel C. (Fed) <daniel.apon@nist.gov>; Moody, Dustin (Fed) <dustin.moody@nist.gov>; Javier Alfonso Verbel Herrera <javerbelh@unal.edu.co>
Subject: Re: Simple Matrix

Okay. It seems that what he is using is the following. Number of variables n is s^2+8s and number of equations m is 2s^2+14s+24. The reason he gives in his paper for the choice of ratio between n and m is "security reasons". He also says that he needs m<= 2n, which he doesn't have.

At the 128 bit security level, he would need at least s=13, according to his analysis, and at that level, the number of variables is more than half the number of equations. Then we only need to find one vector. I still think that there might be something in the smaller cases... some sort of trade off in the probabilities. While it is true that ignoring equations is the same as choosing zero for the coefficients in the minrank modeling we have degrees of freedom in the fact that u+v is 8 larger than 2s.

In fact, I don't think that there are any other problems than that. The other probability conditions only depend on properties of A, so I think that we're good as long as u+v-a is at least 2s, which for the 80-bit parameters we can get exactly with no guessing at the cost of the factor of q from the last step I mentioned above, and in the larger cases we already get for free since the quantity m-2n decreases by 2 as parameters are increased. So that makes our 80-bit attack a factor of q slower and the others have no effect.

On Mon, Nov 25, 2019 at 12:29 PM Daniel Smith (b) (6) wrote:

> Forgot about arithmetic. m is 264, so still slightly larger than 256.
>
> On Mon, Nov 25, 2019, 12:24 PM Daniel Smith (b) (6) wrote:
>
> > Hi, Ray,
> >
> > I think you're mostly right. But...

I forgot to mention that the number of variables they publish is larger. (I also misreported the numbers before. I can't remember anything.) The values I want to attack are

q=2^8

r=11

s=8

u=v=12

n=128

m=254

So you can see that since m=254<2*(128), we only need two kernel vectors, so the complexity is exactly as I said.

They provide some reason why (perhaps related to direct attack) that the number of variables needs to be higher. I know it doesn't affect the probability stuff at all, but it keeps the number of kernel vectors at 2 with no extra cost.

We can also investigate what happens when we have larger numbers of equations. I think that we can get a positive trade off by eliminating a few public equations for a cost in probability that is less severe than the cost in guessing. Specifically, since u+v=2s+8, we should be able to reduce our expected intersection between the band space and the span of the public maps by removing some public maps and still have this probability be not too bad. So it should be a bit better than q^10 times the inverse of the probability, even in that case. With the above values, if we had 3 times as many equations as variables we would have a complexity of 2^80 times linear algebra overhead. So still not break the scheme.

As it is, though, I think that we are still legitimately breaking their proposals. Also, I need to investigate more why they need so many variables. It is certainly not for efficiency.

Cheers,

Daniel

On Mon, Nov 25, 2019 at 11:40 AM Perlner, Ray A. (Fed) <ray.perlner@nist.gov> wrote:

> I don't think the linear algebra search attack works as well as you think it does in the q=2^8 and s=7, r=10, u=v=11 case
>
> Assuming we have r*s=70 variables and r*(u+v) = 220 equations, we need 4 kernel vectors to uniquely solve for linear coefficients for all the equations
>
> In such a case our probability of success is only q^(r-1-4s)*q^(u+v-1-4s) =q^-26 = 2^-208
>
> We can do a little better by only choosing 3 kernel vectors and searching through the expected 10-dimensional solution space for a low rank map
>
> Then our probability of success is q^(r-1-3s)*q^(u+v-1-3s) =q^-12 = 2^-96, but we need to do about q^9=2^72 work to search for a low rank map, so our complexity is still 2^168 times linear algebra.
>
> (unless the plaintext space is larger)
>
> Cheers,
>
> Ray
>
> **From:** Daniel Smith (b) (6)
> **Sent:** Monday, November 25, 2019 11:05 AM
> **To:** Apon, Daniel C. (Fed) <daniel.apon@nist.gov>; Perlner, Ray A. (Fed) <ray.perlner@nist.gov>; Moody, Dustin (Fed) <dustin.moody@nist.gov>; Javier Alfonso Verbel Herrera <javerbelh@unal.edu.co>
> **Subject:** Simple Matrix

Hi, everyone,

I want to give an update on the reaction attack to the ABC Simple Matrix encryption scheme. I would like to submit this to PQCrypto 2020, which has a deadline of tomorrow. I don't think that I'm crazy for this. (There is another week for revisions before it goes to review.)

This is not exactly a central topic right now in post-quantum, and considering the fact that the conference is in Paris, there may be more submissions and make the chances of the paper getting accepted a little lower than normal. I still think that it is worthwhile, though. It is interesting to have a reaction attack on a multivariate scheme. I can't think of any other off the top of my head. Also, if rejected, we will likely get one useful review and two worthless reviews, and we can get more insight into the state of mind of any negative reviewer. Sometimes the difficulty is just in perspective, and our presentation can be altered to better offer our understanding of the significance for a future submission... maybe to SAC2020.

Anyway, here is the summary of the attack. We have A B and C which are matrices of linear forms in the variables x_1,...,x_n. (For simplicity we can use n=rs, though it doesn't matter if n is bigger.) To address decryption failures one needs the number of rows (r) of A to be larger than the number of columns (s). (Also, the dimensions of B and C are s x u and s x v, respectively.) There is an advantage in the linear algebra search minrank attack when r is greater than s. Basically, the attack is faster by a factor of roughly q for each additional row. So we get an advantage of q^(r-s). With decryption failures we get an extra factor of q for free because in effect A has one fewer column. Also, whenever u+v is greater than 2s we get an extra factor of q speedup. The reason is that once we discover two vectors within the union of the kernels of maps coming from the same linear combination of the rows of A the condition that they are in a common kernel is modeled by membership in the kernel of a (u+v) x 2s matrix. When u+v=2s as in the square case, this condition is satisfied with probability q^-1, but when u+v>2s, it is always satisfied. For proposed parameters r=s+3 and both u and v are larger than s. So the probability of finding a second vector in the kernel of a common map of the appropriate form when we have already detected a decryption failure is q^(r-2s), and so the complexity is something like r^3s^4q^(2s-r)\approx s^7q^(s-3). One set of parameters at the 80 bit security level uses q=2^8 and s=7, r=10, u=v=11 so that we get a complexity of about 2^53.

Anyway. I am going to create an overleaf document with the notes I have assembled, and we should try to put a bit of time into getting something reasonably complete as far as the scientific content into easychair by tomorrow.

Daniel, are you still interested in joining us on this one? (The idea for the attack was inspired by you pointing out a different scheme that looked easy to break, so I think it is appropriate to include you if you can contribute to the proofs and writing.) If you want, I can send some explicit statements to prove that we can include in some parts once I get the document up.

Dustin, I am not sure if you have had time to look at your code yet, but it would be nice to use the same thing we already built instead of having to develop a new one. If you need any more details of what experiments we need and how to implement them, please let me know.

Ray, if I recall correctly, you handled the part of the complexity analysis in our original paper that was concerned with all of the fine details of the linear algebra being performed. Do you have time to look at that and correct my r^3s^4 above? I bet it is wrong since I just made it up and didn't think about the solving and rank calculations explicitly.

Javier, as you have time, please expand your notes on the probability of finding a second vector in the kernel of a band space map when the first is known.

I will try to handle everything else and compose an introduction stating that analysis of rank methods seems to be where the most motion in post-quantum security analysis is happening right now. Also, that this seems to be the first time that we have a cryptanalysis boosted by a reaction attack and that it affects the minrank complexity, so we need to consider such attacks for any scheme with a rank defect.

Cheers,

Daniel