From: Sydney Antonov <ska84@protonmail.com> via pqc-forum <pqc-forum@list.nist.gov>
To: pqc-forum@list.nist.gov
Subject: [pqc-forum] Is there a proven lower bound on the fraction of codes with systematic forms?
Date: Tuesday, March 22, 2022 08:58:00 PM ET

Is there a meaningful proven lower bound on either the fraction of Goppa codes with systematic forms (claimed to be approximately 29% by the Classic McEliece spec) or the fraction of binary matrices which are invertible (which would imply a close bound for Goppa codes if McEliece public keys are pseudorandom), for parameters relevant to Classic McEliece?

Meaningful lower bounds could be probabilistically verified with high confidence* but this would complicate formal verification of Classic McEliece's one-way function's security reduction to the original McEliece cryptosystem.

* Generate 100000 codes. If more than 28000 codes have systematic forms then with statistical significance less than 1 in a googol at least 25% of codes have systematic forms. The laptop I'm writing this email on could perform this computation within hours.

Sydney

From: Maxime Bros <m.bros25000@gmail.com> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] Is there a proven lower bound on the fraction of codes with systematic forms?
Date: Wednesday, March 23, 2022 03:05:22 AM ET

Dear Sydney,

The number of square binary matrices n x n that are invertible is easy
to compute,
for the first row you have 2^n-1 non singular vectors, for the second
2^n-2 because
you remove any linear combination of the vectors you already chose, etc.
The total is then \prod_i (2^n-2^i), i in {0..n-1}.
Once you have this number, you divide it by the total number of binary
square matrices,
that is to say 2^(n^2). You get something which is bounded from below by
0.288 (this is
proven and the proof is pretty easy).

I just computed the first values of this ratio with my computer:
n=4 ⇒ 0.3076
n=10 ⇒ 0.2891
n=20 ⇒ 0.2888

For rectangular matrices, you can count them in a similar way, and in
the end you get
a ratio which tends to the same value, for example for 25 x 50 binary
matrices, the ratio of invertible
ones is 0.2888 if my computations (done quickly on my computer this
morning) are correct.

To conclude, it is very likely that the 0.29 ratio you are mentioning
comes from these
computations; roughly speaking, in code-based cryptography we very often
consider that any
binary matrices is non singular with probability 0.29 as long as "it
looks random".

I hope my answer was useful to you,

Sincerely,

Maxime Bros
(University of Limoges, France)

Le 23/03/2022 à 01:57, 'Sydney Antonov' via pqc-forum a écrit :
> Is there a meaningful proven lower bound on either the fraction of
> Goppa codes with systematic forms (claimed to be approximately 29% by
> the Classic McEliece spec) or the fraction of binary matrices which
> are invertible (which would imply a close bound for Goppa codes if
> McEliece public keys are pseudorandom), for parameters relevant to
> Classic McEliece?
>
> Meaningful lower bounds could be probabilistically verified with high
> confidence* but this would complicate formal verification of Classic
> McEliece's one-way function's security reduction to the original McEliece
> cryptosystem.
>
> * Generate 100000 codes. If more than 28000 codes have systematic forms
> then with statistical significance less than 1 in a googol at least 25%
> of codes have systematic forms. The laptop I'm writing this email on
> could perform this computation within hours.
>
> Sydney
>

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] Is there a proven lower bound on the fraction of codes with systematic forms?
Date: Wednesday, March 23, 2022 07:54:32 AM ET
Attachments: smime.p7m

```
A uniform random dxd matrix over F_2 is invertible with probability
exactly (1-1/2)(1-1/4)(1-1/8)...(1-1/2^d). See, e.g., Theorem 99 in
Dickson's 1901 book on linear groups:

   https://archive.org/details/lineargroupswith00ledi/page/n89/mode/2up

The probability is within 1/2^d of its limit as d->infinity. The limit
is

   prod_{integers d ⩾ 1} (1-1/2^d)
   = sum_{integers k} (-1)^k 2^(-k(3k+1)/2)
   = binary 0.010010011110111000000100001111111101111100000000001000
              000111111111111011111110000000000000010000000011111111
              111111110111111111000000000000000000100000000001111111...
   = 0.288788095086602421278899721929230780088911904840685784114741...

by Euler's pentagonal-number theorem.

Public keys in the original McEliece cryptosystem, with the usual
parameter choices, are commonly conjectured to be indistinguishable from
uniform random matrices of the same size. This indistinguishability
implies indistinguishability of the leading square matrix from uniform,
in turn implying that an invertibility test doesn't distinguish the
leading square matrix from uniform, i.e., that the invertibility chance
is indistinguishable from (1-1/2)(1-1/4)(1-1/8)...(1-1/2^d).

Statistically pinning down the actual probability is a simple matter of
generating many McEliece matrices and seeing how often the leading
square matrix is invertible; or, for the reciprocal of the probability,
running keygen many times, as in the script below. (For experiments
```

using deterministic RNG seeds, change "fast" to "known" in the script.) An experiment generating 1000000 keys for mceliece6960119 used 3466938 matrices in total.

The limited statement that the probability is ≥25% implies that there is a "security difference of at most 2 bits" (to quote the Classic McEliece submission) between systematic-form public keys and arbitrary public keys. For formal verification, it's best to include this limited statement as a hypothesis, since the statement is directly statistically verifiable, rather than deriving the statement from the hypothesis of public-key indistinguishability, which is overkill for the security analysis.

---Dan (speaking for myself)

```
m=mceliece6960119
mkdir goppasystematic
cd goppasystematic
wget https://bench.cr.yp.to/supercop/supercop-20220213.tar.xz
tar -xf supercop-20220213.tar.xz
cd supercop-20220213
sed -i 1q okcompilers/c
: > okcompilers/cpp
chmod +t crypto_kem/$m/ref
touch crypto_kem/$m/used
for opi in crypto_kem/$m/*/
do
  python3 -c '
import sys
gaussstate = 0
print("long long numgauss = 0;")
print("long long numsystematic = 0;")
for line in sys.stdin:
  if gaussstate == 0 and line.find("gauss") >= 0:
    gaussstate = 1
    print("++numgauss;")
```

```
  sys.stdout.write(line)
  if gaussstate > 0:
    gaussstate += line.count("{")
    if line.count("}") > 0:
      gaussstate -= line.count("}")
      if gaussstate == 1:
        print("++numsystematic;")
        gaussstate = -1
  ' < "$opi/pk_gen.c" > "$opi/pk_gen.c.new" \
  && mv "$opi/pk_gen.c.new" "$opi/pk_gen.c"
done
./do-part init
./do-part keccak
./do-part crypto_sort int32
./do-part crypto_hash shake256
./do-part crypto_stream chacha20
./do-part crypto_rng
./do-part crypto_kem $m
(
  echo '#include <stdio.h>'
  echo '#include <stdlib.h>'
  echo '#include "crypto_kem_'"$m"'.h"'
  echo 'unsigned char pk[crypto_kem_mceliece6960119_PUBLICKEYBYTES];'
  echo 'unsigned char sk[crypto_kem_mceliece6960119_SECRETKEYBYTES];'
  echo 'void crypto_declassify(void *x,unsigned long long xlen)'
  echo '{'
  echo '}'
  echo 'void randombytes_callback(void *x,unsigned long long xlen)'
  echo '{'
  echo '}'
  echo 'extern long long numgauss,numsystematic;'
  echo 'int main(int argc,char **argv)'
  echo '{'
  echo '  for (long long loop = 0;loop < atoll(argv[1] ? argv[1] : "100");++loop)'
  echo '    crypto_kem_mceliece6960119_keypair(pk,sk);'
  echo '  printf("gauss %lld systematic %lld\n",numgauss,numsystematic);'
  echo '  return 0;'
```

```
  echo '}'
) > experiment.c
gcc -I bench/*/include/*/constbranchindex -o experiment experiment.c \
bench/*/lib/*/fastrandombytes.o \
bench/*/lib/*/kernelrandombytes.o \
bench/*/lib/*/libsupercop.a \
bench/*/lib/*/libkeccak.a
./experiment 10000
```