FAVPQC 2023

Formal specification of the post-quantum signature scheme FALCON in Maude

Víctor García[1] (vicgarval@upv.es)

Santiago Escobar[1] (sescobar@upv.es)

Kazuhiro Ogata[2] (ogata@jaist.ac.jp)

[1]Universitat Politècnica de València (UPV), Camí de Vera, s/n, 46022 València, Valencia, Spain

[2]Japan Advanced Institute of Science and Technology (JAIST), Ishikawa 923–1292, Japan

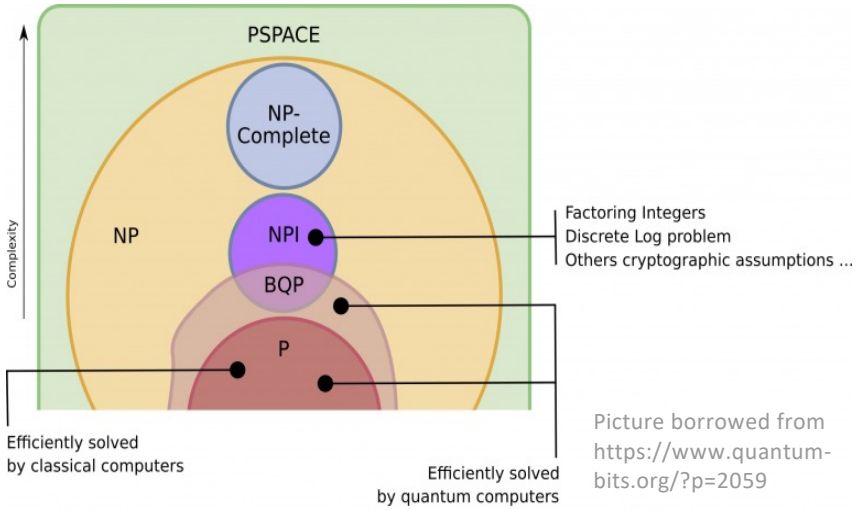
Contents

Contents

Motivation

Threat of adversaries with quantum capabilities

- Shor's algorithm (1994)
- Grover's search algorithm (1996)



Picture borrowed from https://www.quantum-bits.org/?p=2059

Search of solutions by NIST with the PQC project (round 3)

- Key Encapsulation Mechanisms: CRYSTALS-Kyber
- Digital Signature Schemes: CRYSTALS-Dilithium, **FALCON**, SPHINCS+

Motivation

Types of security analysis

- Computational
  - Mathematical proofs and probabilities
  - Keys, messages,... are bit strings
  - Closer to reality, used by cryptographers
- Symbolic
  - Cryptographic primitives as black boxes
  - Keys, messages,… are symbols
  - Suitable for automation and easier to understand for non-experts of cryptography

Contents

Maude

What is Maude?

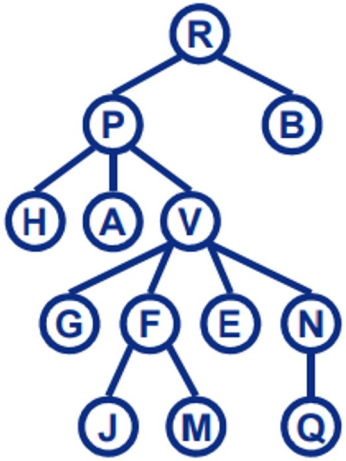
Maude is a modelling, programming and verification language based on rewriting logic.

Why Maude?

Because it allows us to specify and execute systems in a simple and intuitive way.

Which verification tools are provided?

Reachability analysis using the *search* command from an initial state to a target state. Moreover, under the assumption of a finite number of reachable states from a given initial state, one can use Maude's LTL model checket to prove any properties with LTL formulas.

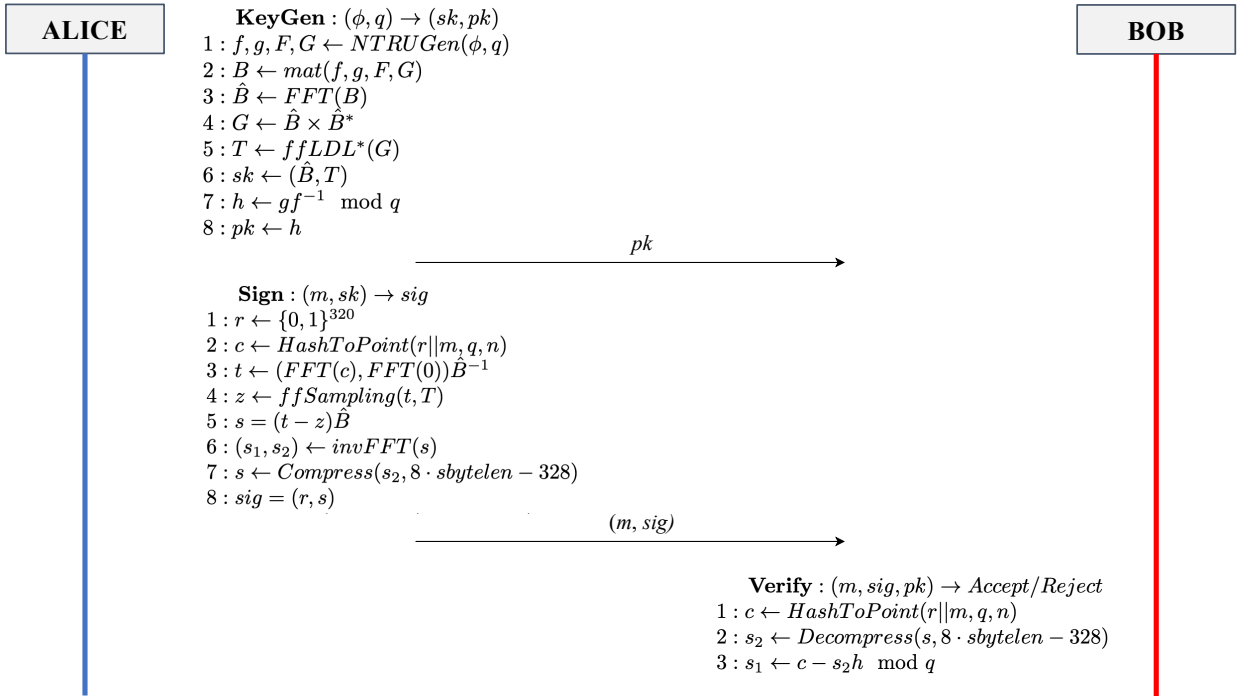


Contents

FALCON

What is FALCON?

Falcon is a signature scheme based on lattices to sign and verify messages.



Contents

MODEL

Main components

Message

```
op msg{(_,_)[_]_} : Identifier Identifier MsgState Content -> Msg .
```

Principal

```
op _[_]_ : Identifier Keys Content -> Principal [ctor] .
```

Environment

```
op {_}<_>net(_) : Content Principals Msgs -> GlobalState
```

MODEL

$$\textbf{KeyGen} : (\phi, q) \rightarrow (sk, pk)$$

$1 : f, g, F, G \leftarrow NTRUGen(\phi, q)$

$2 : B \leftarrow mat(f, g, F, G)$

$3 : \hat{B} \leftarrow FFT(B)$

$4 : G \leftarrow \hat{B} \times \hat{B}^*$

$5 : T \leftarrow ffLDL^*(G)$

Collapsed

$6 : sk \leftarrow (\hat{B}, T)$

$7 : h \leftarrow gf^{-1} \mod q$

$8 : pk \leftarrow h$

```
crl [KeyGen] :  {phis(SAM1, CONT1), qs(SAM2, CONT2), CONT3}
                <
                 (ID1[emptyK]peer(ID2))
                 PS
                >net(MSGS)
                =>
                {phis(CONT1), qs(CONT2), CONT3}
                <
                 (ID1[publicKey(ID1,PK) ; secretKey(ID1,SK)]
                  qI(ID1,SAM2), phiI(ID1,SAM1), peer(ID2))
                 PS
                >net(MSGS)
                if  L := NTRUGen(SAM1,SAM2) /\
                SK := ([FFT(mat(L)),ffLDL(FFT(mat(L)) x FFT(mat(L)))]) /\
                PK := (elem(2,L) p* inv(elem(1,L))) mod SAM2 /\
                ID1 =/= ID2 .
```

MODEL

$$\textbf{Sign} : (m, sk) \rightarrow sig$$

$1 : r \leftarrow \{0,1\}^{320}$
$2 : c \leftarrow HashToPoint(r||m, q, n)$
$3 : t \leftarrow (FFT(c), FFT(0))\hat{B}^{-1}$
$4 : z \leftarrow ffSampling(t, T)$
$5 : s = (t - z)\hat{B}$
$6 : (s_1, s_2) \leftarrow invFFT(s)$
$7 : s \leftarrow Compress(s_2, 8 \cdot sbytelen - 328)$
$8 : sig = (r, s)$

```
crl [Sign] :    {ms(str(STR), CONT1), rs(SAM1, CONT2), CONT3}
                < (ID1[secretKey(ID1,SK) ; KS1]qI(ID1,Q), CONT4) PS >net(MSGS)
                =>
                {ms(CONT1), rs(CONT2), CONT3}
                < (ID1[KS1]sig(STR,SAM1,Compress(second(Ss),SBYTELEN)) ,
                    mI(ID1,STR), qI(ID1,Q), CONT4) PS >
                net(MSGS)
                if c := HashToPoint(SAM1 || STR, Q, n) /\
                   t := [FFT(c), FFT(0)] p* inv(first(SK)) /\
                   z := ffSampling(t, second(SK)) /\
                   s := (t p- z) p* first(SK) /\
                   Ss := invFFT(s) .
```

MODEL

Verify : $(m, sig, pk) \rightarrow Accept/Reject$

1 : $c \leftarrow HashToPoint(r||m, q, n)$

2 : $s_2 \leftarrow Decompress(s, 8 \cdot sbytelen - 328)$

3 : $s_1 \leftarrow c - s_2 h \mod q$

```
crl [Verify] :  {CONT1}
                <
                 (ID1[publicKey(ID2, p mod Q) ; KS1]
                 mI(ID2,STR), sig(STR,R,P), peer(ID2), CONT2) PS >
                net(MSGS)
                =>
                {CONT1}
                < (ID1[KS1]mI(ID2,STR), peer(none), CONT2) PS >
                net(MSGS)
                if  c := HashToPoint(R || STR, Q, n) /\
                S2 := Decompress(P, SBYTELEN) /\
                   h := p mod Q /\
                S1 := (c p- (S2 p* h)) mod Q /\
                 (S1 == s1) /\ (S2 == s2) .
```

Contents

Experiments


```
eq init1 =  {phis(phi), qs(q), ms(str(m)), rs(r)}
            <   (Alice[emptyK]peer(Bob))
                (Eve[emptyK]peer(none))
                (Bob[emptyK]peer(Alice))
            >net(emptyM) .
```

Experiments

Executable?

```
==========================================
rewrite in FALCON : init1 .
rewrites: 30 in 0ms cpu (0ms real) (254237 rewrites/second)
result [GlobalState]:
{phis(emptyC),qs(emptyC),ms(emptyC),rs(emptyC)}
<
(Alice[emptyK]peer(none),phiI(Alice, phi),qI(Alice, q))
(Eve[emptyK]peer(none))
Bob[emptyK]peer(none),mI(Alice, m)
>net(msg{(Alice,Bob)[received](g p* inv(f)) mod q}
       msg{(Alice,Bob)[received]str(m),[r,Compress(s2, SBYTELEN)]})
```

Experiments

Terminating?

```
==========================================
search in FALCON : init1 =>!
{CONT1}
<
PS
ID1[KS1]peer(none),mI(ID2, STR)
>net(MSGS) such that ID1 =/= ID2 = true .

Solution 1 (state 27)
states: 29  rewrites: 126 in 0ms cpu (0ms real) (320610 rewrites/second)
CONT1 --> phis(emptyC),qs(emptyC),ms(emptyC),rs(emptyC)
PS --> (Alice[emptyK]peer(none),phiI(Alice, phi),qI(Alice, q))
Eve[emptyK]peer(none)
ID1 --> Bob
KS1 --> emptyK
ID2 --> Alice
STR --> m
MSGS --> msg{(Alice,Bob)[received](g p* inv(f)) mod q}
        msg{(Alice,Bob)[received]str(m),[r,Compress(s2, SBYTELEN)]}
```

```
Solution 2 (state 28)
states: 29  rewrites: 127 in 0ms cpu (0ms real) (293981 rewrites/second)
CONT1 --> phis(emptyC),qs(emptyC),ms(emptyC),rs(emptyC)
PS --> (Eve[emptyK]peer(none))
Bob[emptyK]peer(none),phiI(Bob, phi),qI(Bob, q)
ID1 --> Alice
KS1 --> emptyK
ID2 --> Bob
STR --> m
MSGS --> msg{(Bob,Alice)[received](g p* inv(f)) mod q}
        msg{(Bob,Alice)[received]str(m),[r,Compress(s2, SBYTELEN)]}

No more solutions.
states: 29  rewrites: 127 in 0ms cpu (0ms real) (276688 rewrites/second)
Maude>
```

Contents

Conclusion

Concluding remarks

- We specified the digital signature scheme FALCON in Maude.
- We checked that the specification is executable and terminating, capturing the behaviour of FALCON.

Future work

- We will use Maude tools like the LTL model checker to verify properties like liveness or security.
- We will explore other signature schemes and specify them, so we can compare our models in terms of satisfied properties.
- We could adapt the specification to use the object oriented notation provided by Maude.