UNIVERSITAT POLITÈCNICA DE VALÈNCIA

FAVPQC 2022

VRAIN Valencian Research Institute for Artificial Intelligence

Modeling and verification of the post-quantum key encapsulation mechanism KYBER using Maude

Víctor García
vicgarv2@upv.es

Santiago Escobar
sescobar@upv.es

Universitat Politècnica de València
VRAIN - Valencian Research Institute for Artificial Intelligence

FAVPQC 2022

CONTENTS

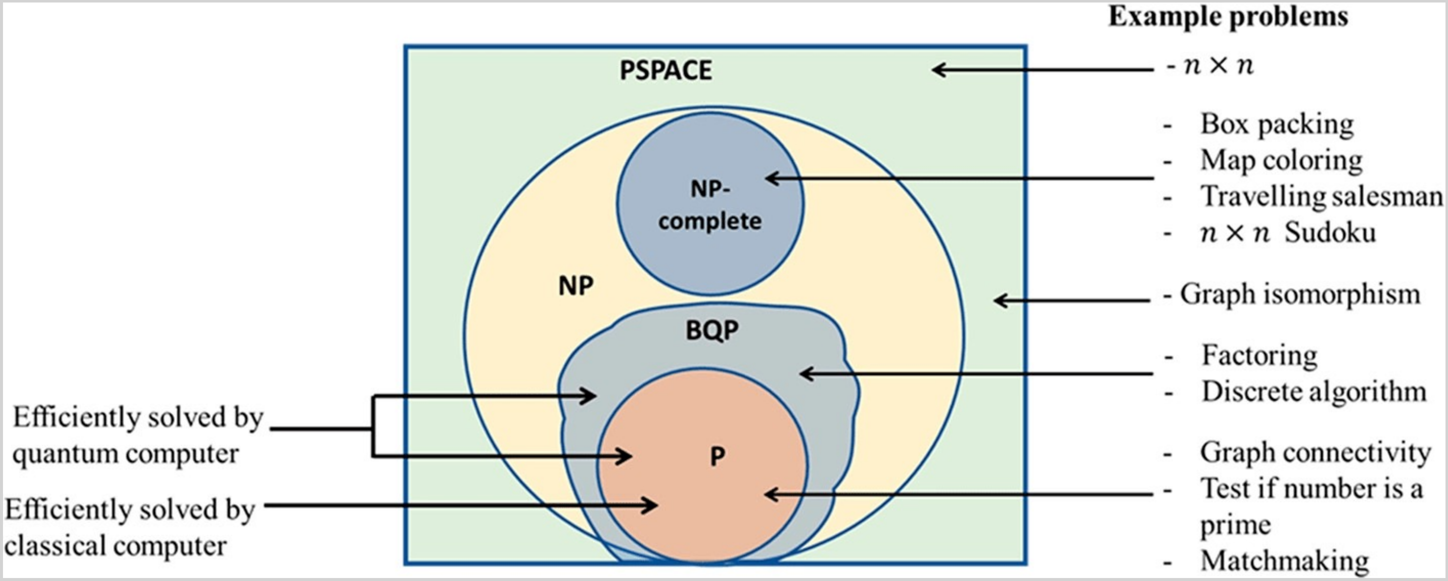
1. Introduction

Security is keystone for today's society

- Security is achieved with the use of protocols and other tools.
- Protocol security is based on the difficulty of solving:
  - Integer factorization: RSA.
  - Discrete logarithm: ElGamal, Diffie-Helman.
  - Elliptic-curve discrete logarithm: EC Diffie-Hellman.

1. Introduction

Quantum computing supposes a threat to security



1. Introduction

Solution proposed by NIST

- Post-Quantum Cryptography project (2017-2022)
- Round 3 finalists:
  - Key-establishment: **CRYSTALS-KYBER**, SABER, Classic McElicee and NTRU
  - Digital signature: CRYSTALS-DILITHIUM, FALCON and Rainbow.

1. Introduction

Security analysis

Computational

- Mathematical proofs and probabilities.
- Keys, messages,... are bit strings.
- Closer to reality, used by cryptographers, already applied to Kyber.

Symbolic

- Cryptographic primitives as black boxes.
- Keys, messages,... are symbols.
- Suitable for automation and easier to understand for non experts of cryptography.

2. Dolev-Yao adversary model

Types of adversaries

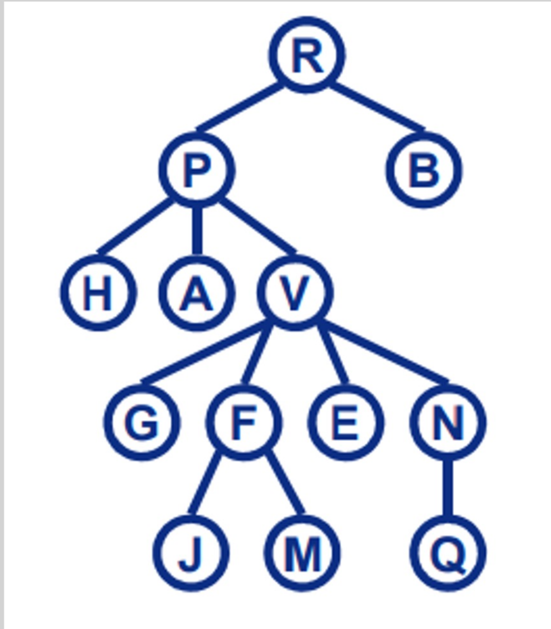
- Passive (eavesdropper).
- Active (total control).

Capabilities

- Intruder can obtain any message that is on the network.
- Intruder is a legitimate user of the network, that is, he/she can do any of the actions an honest participant could do.
- The intruder could interact with honest participants, that is, he/she can receive messages from other participants.

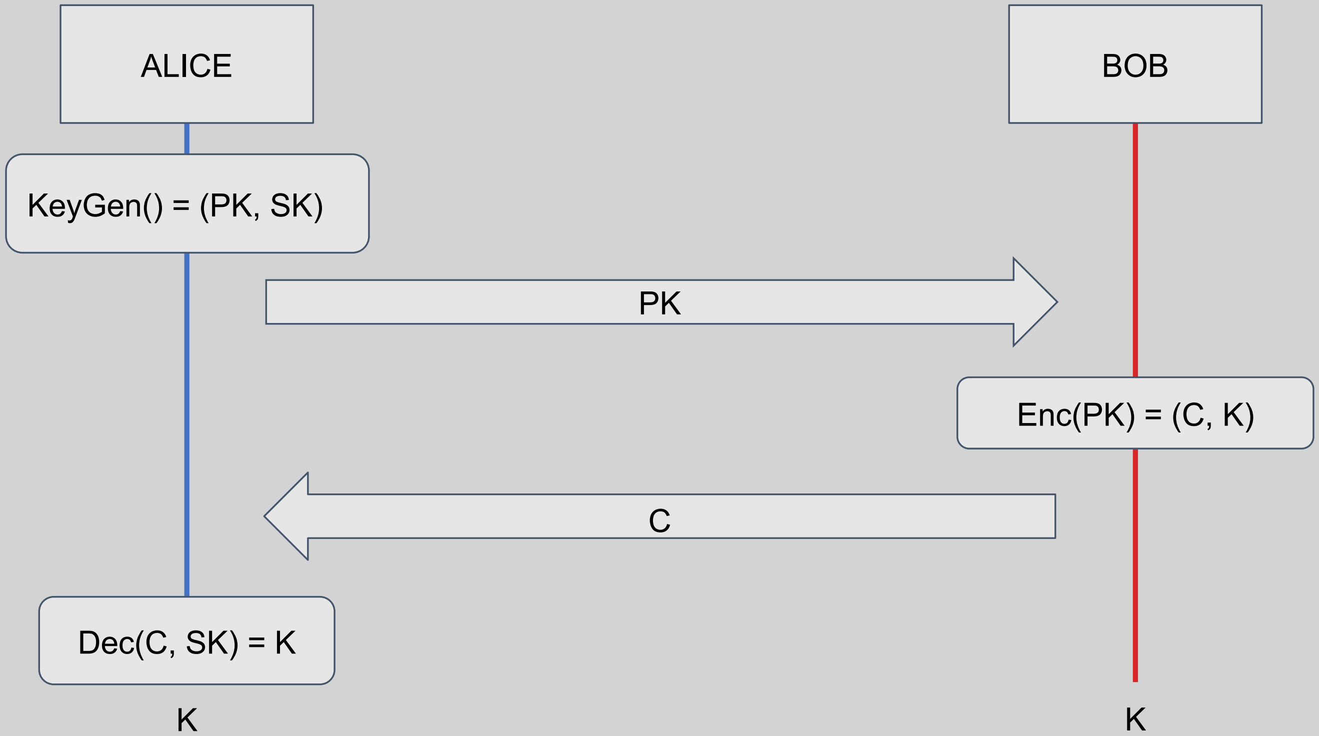
3. Maude

- **Maude** (v3.2.1) is a modeling, programming and verification language.
- Provides explicit state model checking using *search* command or LTL properties.
- Origins at Stanford, California.
- Project members
  - USA
  - Norway
  - Spain



4. Kyber

FAVPQC
2022



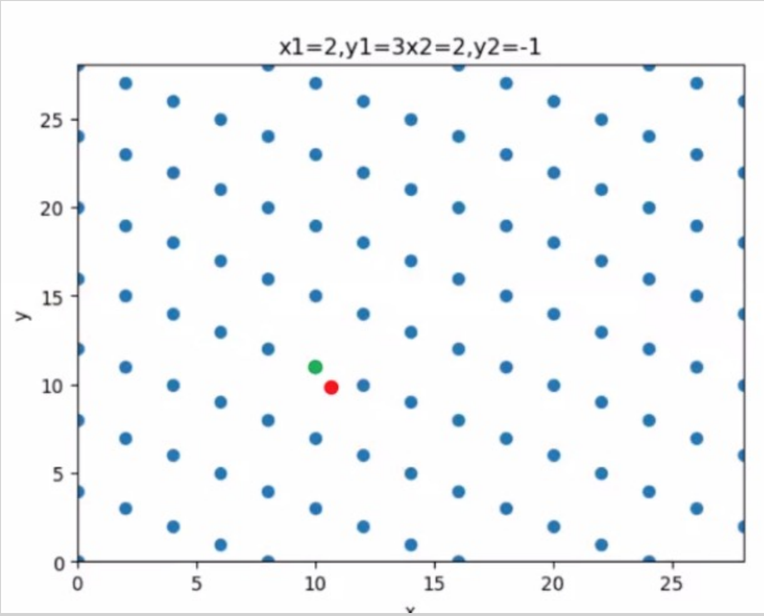
FAVPQC 2022

4. Kyber

- Security basis: Hardness of solving LWE problem over module lattices.

$$t = A * s + e$$

Where *A* is a **square** matrix, *s* and *e* are **column** vectors, all of them being of size **n**.



4. Kyber

- Thanks to the errors introduced on Enc and corrected on Dec.

KEM.KeyGen()
$z \leftarrow \mathcal{B}^{32}$
$(pk, sk') = \text{CPAPKE.KeyGen}()$
$sk = (sk' || pk || \text{H}(pk) || z)$
return (pk, sk)

$\xrightarrow{pk}$

KEM.Enc(pk)
$m_0 \leftarrow \mathcal{B}^{32}$
$m = \text{H}(m_0)$
$(\bar{K}, r) = \text{G}(m || \text{H}(pk))$
$c = \text{CPAPKE.Enc}(pk, m, r)$
$K = \text{KDF}(\bar{K}, \text{H}(c))$
return (c, K)

$\xleftarrow{c}$

KEM.Dec(c, sk)
$(\mathbf{s} || pk || \text{H}(pk) || z) = sk$
$m' = \text{CPAPKE.Dec}(c, \mathbf{s})$
$(\bar{K}', r') = \text{G}(m' || \text{H}(pk))$
$c' = \text{CPAPKE.Enc}(pk, m', r')$
if $c = c'$ **then return** $K = \text{KDF}(\bar{K}', \text{H}(c))$
else return $K = \text{KDF}(z, \text{H}(c))$

CPAPKE.KeyGen()
$d \leftarrow \mathcal{B}^{32}$
$(\rho, \sigma) = \text{G}(d)$
$R_q^{k \times k} \ni \mathbf{A} = \text{generate}(\rho)$
$R_q^k \ni \mathbf{s}, \mathbf{e} \leftarrow \text{sampleCBD}(\sigma)$
$\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$
$pk = (\mathbf{t} || \rho)$
$sk = \mathbf{s}$
return (pk, sk)

CPAPKE.Dec(c, sk)
$(c_1 || c_2) = c$
$\mathbf{u}' = \text{Decompress}_q(c_1, d_u)$
$v' = \text{Decompress}_q(c_2, d_v)$
$m' = \text{Compress}_q(v' - \mathbf{s}^T \mathbf{u}', 1)$
return m'

CPAPKE.Enc(pk, m, r)
$(\mathbf{t} || \rho) = pk$
$R_q^{k \times k} \ni \mathbf{A} = \text{generate}(\rho)$
$R_q^k \ni \mathbf{r}, \mathbf{e}_1 \leftarrow \text{sampleCBD}(r)$
$R_q \ni e_2 \leftarrow \text{sampleCBD}(r)$
$\mathbf{u} = \mathbf{A}^T \mathbf{r} + \mathbf{e}_1$
$v = \mathbf{t}^T \mathbf{r} + e_2 + \text{Decompress}_q(m, 1)$
$c_1 = \text{Compress}_q(\mathbf{u}, d_u)$
$c_2 = \text{Compress}_q(\mathbf{v}, d_v)$
return $c = (c_1 || c_2)$

FAVPQC 2022

5. Symbolic model - Honest

- Protocol operations like *KeyGen*, *Enc* and *Dec,* each represented by a conditional rule.
- Network operations to send and receive public keys and ciphered texts.
- Mathematical assumptions
  - Square matrices.
  - Column vectors.
  - Some samples come from centered binomial distributions.
  - Only consider the cases where error correction does not fail.

5. Symbolic model – Equational theories

Decapsulation property

$$Decompress(Compress(X,N),N) = X$$
$$Compress(Decompress(X,N),N) = X$$

Noise cancelation property

$$\left(V1\ + Decompress(X,N)\right) - V2 = Decompress(X,N)$$

Properties over operations

FAVPQC 2022

5. Symbolic model - Intruder

```
crl [Intercept1] :

{ CONT }
 < (Eve[publicKey(Eve,PK) ; KS1]CONT1) (Alice[publicKey(Alice,PK') ; KS2]CONT2) PS >
net(MSGS msg{(Alice,Bob)[sentPK]PK'})
                                        =>
{ CONT }
< (Eve[publicKey(Eve,PK) ; publicKey(Alice,PK') ; KS1]CONT1) (Alice[publicKey(Alice,PK') ; KS2]CONT2) PS >
net(MSGS msg{(Alice,Bob)[interceptedPK]PK'} msg{(Alice,Bob)[sentPK]PK})

if (msg{(Alice,Bob)[interceptedPK]PK'}) in MSGS == false .
```

FAVPQC 2022

5. Symbolic model - Intruder

```
crl [Intercept2] :

{ CONT }
< (Eve[publicKey(Eve,PK) ; KS1]cl(Eve,C') CONT1) PS >
net(MSGS msg{(Bob,Alice)[sentC]C})
                                                     =>
{ CONT }
< (Eve[publicKey(Eve,PK) ; KS1]cl(Eve,C') cl(Bob,C) CONT1) PS >
net(MSGS msg{(Bob,Alice)[interceptedC]C} msg{(Bob,Alice)[sentC]C'})

if (msg{(Bob,Alice)[interceptedC]C}) in MSGS == false /\
   C =/= C' .
```

6. Verification - Reachability

QUESTION: *Can we reach a state where two participants have shared different keys between them and a third participant has both for each of them?*

```
search in KYBERV2 : init2 =>* {CONT}< (ID1[KS1 ; sharedKey(ID3, K1)]CONT1) (
    ID2[KS2 ; sharedKey(ID1, K1) ; sharedKey(ID3, K2)]CONT2) ID3[KS3 ;
    sharedKey(ID1, K2)]CONT3 >net(MSGS) .
```

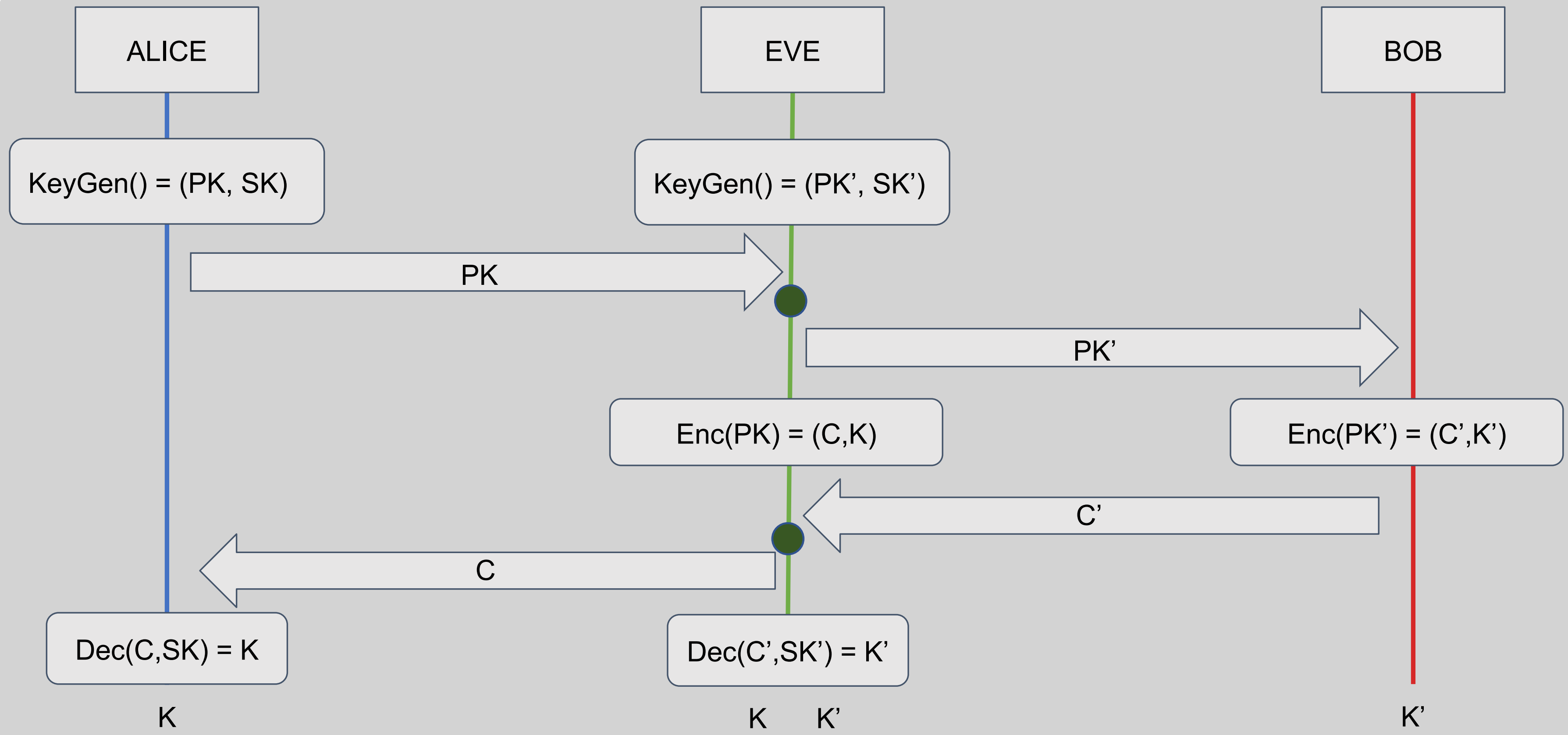
FAVPQC
2022

6. Verification - Reachability

```
Solution 1 (state 34000)
states: 34001  rewrites: 2394038 in 2997ms cpu (3027ms real) (798636
    rewrites/second)
CONT --> ds(emptyS) ms(emptyS) rs(emptyS)
ID1 --> Alice
KS1 --> publicKey(Alice, e v+ (A1 m* s1)) ; secretKey(Alice, s1)
ID3 --> Bob
K1 --> m1
CONT1 --> dI(Alice, d1)
ID2 --> Eve
KS2 --> publicKey(Alice, e v+ (A1 m* s1)) ; publicKey(Eve, e' v+ (A2 m* s2)) ;
    secretKey(Eve, s2)
K2 --> m2
CONT2 --> dI(Eve, d2) mI(Eve, m1) rI(Eve, r1) cI(Eve, Compress(e11 v+ (tpM(A1)
    m* r1'), du),Compress(e21 v+ (tpV(s1) dot (tpM(A1) m* r1')) v+ (tpV(e) dot
    r1') v+ Decompress(m1, 1), dv))
KS3 --> publicKey(Alice, e' v+ (A2 m* s2))
CONT3 --> mI(Bob, m2) rI(Bob, r2) cI(Bob, Compress(e12 v+ (tpM(A2) m* r2'),
    du),Compress(e22 v+ (tpV(s2) dot (tpM(A2) m* r2')) v+ (tpV(e') dot r2') v+
    Decompress(m2, 1), dv))
MSGS --> msg{(Alice,Bob)[interceptedPK]e v+ (A1 m* s1)} msg{(Alice,Bob)[
    receivedPK]e' v+ (A2 m* s2)} msg{(Bob,Alice)[interceptedC]Compress(e12 v+ (
    tpM(A2) m* r2'), du),Compress(e22 v+ (tpV(s2) dot (tpM(A2) m* r2')) v+ (
    tpV(e') dot r2') v+ Decompress(m2, 1), dv)} msg{(Bob,Alice)[
    receivedC]Compress(e11 v+ (tpM(A1) m* r1'), du),Compress(e21 v+ (tpV(s1)
    dot (tpM(A1) m* r1')) v+ (tpV(e) dot r1') v+ Decompress(m1, 1), dv)}
```

FAVPQC 2022

6. Verification - Reachability



FAVPQC
2022

6. Verification - Fairness

Natural language: *Provided that eventually in a future state two honest participants want to share a key, then, there is eventually a future state where both honest participants have shared a key.*

Associated formula in LTL:

$$(GF\ wantsToShareKey(Alice, Bob)) \rightarrow (GF\ sharedAKeyWith(Alice, Bob))$$

```
Maude> red modelCheck(initial1, ([]<> wantsToShareKey(Alice,Bob)) -> ([]<> sharedAKeyWith(Alice,Bob))) .
reduce in KYBERV2-CHECK : modelCheck(initial1, []<> wantsToShareKey(Alice, Bob) -> []<> sharedAKeyWith(Alice, Bob)) .
rewrites: 1469 in 0ms cpu (2ms real) (1883333 rewrites/second)
result Bool: true
Maude> red modelCheck(initial2, ([]<> wantsToShareKey(Alice,Bob)) -> ([]<> sharedAKeyWith(Alice,Bob))) .
reduce in KYBERV2-CHECK : modelCheck(initial2, []<> wantsToShareKey(Alice, Bob) -> []<> sharedAKeyWith(Alice, Bob)) .
rewrites: 8425427 in 14554ms cpu (14577ms real) (578884 rewrites/second)
result Bool: true
```

6. Verification - Security

Natural language: *It is always true that Alice's secret key is not stolen by the intruder Eve.*

Associated formula in LTL:

$$G \neg(stolenSecret(Alice, Eve))$$

```
Maude> red modelCheck(initial1, ([] ~(stolenSecret(Alice,Eve)))) .
reduce in KYBERV2-CHECK : modelCheck(initial1, []~ stolenSecret(Alice, Eve)) .
rewrites: 1404 in 0ms cpu (2ms real) (8886075 rewrites/second)
result Bool: true
Maude> red modelCheck(initial2, ([] ~(stolenSecret(Alice,Eve)))) .
reduce in KYBERV2-CHECK : modelCheck(initial2, []~ stolenSecret(Alice, Eve)) .
rewrites: 8412743 in 10757ms cpu (10820ms real) (782015 rewrites/second)
result Bool: true
```

7. Conclusion

- In this work **we have**:
  - Understood how the KEM Kyber works and learned why is it resistant against quantum computers.
  - Constructed a new model under the previous reviewed specification and applied Dolev-Yao adversary assumptions.
  - Proven the presence of a man-in-the-middle attack on the KEM Kyber with reachability analysis.
  - Specified and applied two LTL formulas, one for fairness and one for security, to carry out a deeper analysis of the model than previous work.

7. Conclusion

- As **future work** we have in mind:
  - Extend the analysis with more properties and formulas in model checking in order to properly verify the model.
  - Use of other tools, such as MaudeNPA to perform unbounded session verification.
  - Combine our specification with other specification (like a signature protocol) to see the combination of behaviours.
  - Apply the same methodology to other protocols in the PQC project.
  - Use of Maude's objects so it is closer to other languages and even more understandable for non experts in formal methods.