Package ‘kpqc’

November 4, 2026

Type Package

Title Korean Post-Quantum Cryptography Algorithms

Version 0.1.0

Copyright see file COPYRIGHTS

Description Provides synchronous interfaces to four algorithms for post-quantum cryptography: 'AIMer', a digital-signature algorithm based on symmetric primitives; 'HAETAE', a module-lattice-based digital-signature algorithm; and 'NTRU+' and 'SMAUG-T', lattice-based key-encapsulation algorithms. These algorithms were selected by the Korean Post-Quantum Cryptography competition. Native reference implementations are compiled into the package and use operating-system entropy for key generation, signing, and encapsulation.

License MIT + file LICENSE

URL https://github.com/KpqC/kpqc-r

BugReports https://github.com/KpqC/kpqc-r/issues

Depends R (>= 3.6.0)

Encoding UTF-8

SystemRequirements C11

NeedsCompilation yes

Author Owen Suolfou [aut, cre]

Maintainer Owen Suolfou <osuolfou@gmail.com>

Repository CRAN

Date/Publication 2026-11-04 21:10:02 UTC

Contents

kpqc-package 2
kpqc_algorithms 2
kpqc_operations 3
print.kpqc_algorithm 4

Index **5**

kpqc-package *Korean Post-Quantum Cryptography Algorithms*

Description

Synchronous R interfaces to AIMer and HAETAE signatures and NTRU+ and SMAUG-T key encapsulation mechanisms (KEMs).

Details

Binary values are represented as raw vectors. Use `charToRaw()` when a text message must be converted explicitly.

kpqc_algorithms *KpqC Algorithms*

Description

Construct algorithm descriptors for every supported KpqC parameter set.

Usage

```
aimer128f()
aimer128s()
aimer192f()
aimer192s()
aimer256f()
aimer256s()
haetae2()
haetae3()
haetae5()
ntruplus768()
ntruplus864()
ntruplus1152()
smaugt128()
smaugt192()
smaugt256()
timer()
signature_algorithms()
kem_algorithms()
```

Value

An accessor returns a `kpqc_algorithm` object containing the algorithm identifier, type, and byte sizes. The collection functions return lists of all signature or KEM algorithms.

Examples

```
algorithm <- aimer128f()
algorithm$id
algorithm$sizes

length(signature_algorithms())
length(kem_algorithms())
```

---

kpqc_operations *KpqC Cryptographic Operations*

Description

Generate key pairs, create and verify detached signatures, and establish shared secrets using a key encapsulation mechanism (KEM).

Usage

```
generate_key_pair(algorithm)

sign_message(algorithm, message, secret_key, context = raw())

verify_signature(
  algorithm,
  message,
  signature,
  public_key,
  context = raw()
)

encapsulate_secret(algorithm, public_key)

decapsulate_secret(algorithm, ciphertext, secret_key)
```

Arguments

algorithm	A KpqC algorithm object returned by an accessor.
message	A raw vector containing the message.
secret_key	A raw vector containing the secret key.
context	An optional raw vector of at most 255 bytes.
signature	A raw vector containing a detached signature.
public_key	A raw vector containing the public key.
ciphertext	A raw vector containing a KEM ciphertext.

Value

`generate_key_pair()` returns a list with `public_key` and `secret_key`. `sign_message()` returns a signature raw vector and `verify_signature()` returns one logical value. `encapsulate_secret()` returns a list with `ciphertext` and `shared_secret`; `decapsulate_secret()` returns a shared-secret raw vector.

Examples

```
algorithm <- aimer128f()
keys <- generate_key_pair(algorithm)
message <- charToRaw("hello")
signature <- sign_message(algorithm, message, keys$secret_key)
verify_signature(algorithm, message, signature, keys$public_key)

kem <- ntruplus768()
recipient <- generate_key_pair(kem)
outbound <- encapsulate_secret(kem, recipient$public_key)
inbound <- decapsulate_secret(kem, outbound$ciphertext, recipient$secret_key)
stopifnot(identical(inbound, outbound$shared_secret))
```

`print.kpqc_algorithm` *Print KpqC Objects*

Description

Print compact summaries without displaying secret key or shared-secret bytes.

Usage

```
## S3 method for class 'kpqc_algorithm'
print(x, ...)

## S3 method for class 'kpqc_key_pair'
print(x, ...)

## S3 method for class 'kpqc_encapsulated_secret'
print(x, ...)
```

Arguments

`x`	A KpqC algorithm or result object.
`...`	Additional arguments, currently ignored.

Value

The input object, invisibly.

Index

aimer128f *(kpqc_algorithms)*, 2
aimer128s *(kpqc_algorithms)*, 2
aimer192f *(kpqc_algorithms)*, 2
aimer192s *(kpqc_algorithms)*, 2
aimer256f *(kpqc_algorithms)*, 2
aimer256s *(kpqc_algorithms)*, 2

decapsulate_secret *(kpqc_operations)*, 3

encapsulate_secret *(kpqc_operations)*, 3

generate_key_pair *(kpqc_operations)*, 3

haetae2 *(kpqc_algorithms)*, 2
haetae3 *(kpqc_algorithms)*, 2
haetae5 *(kpqc_algorithms)*, 2

kem_algorithms *(kpqc_algorithms)*, 2
kpqc *(kpqc-package)*, 2
kpqc-package, 2
kpqc_algorithms, 2
kpqc_operations, 3

ntruplus1152 *(kpqc_algorithms)*, 2
ntruplus768 *(kpqc_algorithms)*, 2
ntruplus864 *(kpqc_algorithms)*, 2

print.kpqc_algorithm, 4
print.kpqc_encapsulated_secret *(print.kpqc_algorithm)*, 4
print.kpqc_key_pair *(print.kpqc_algorithm)*, 4

sign_message *(kpqc_operations)*, 3
signature_algorithms *(kpqc_algorithms)*, 2
smaugt128 *(kpqc_algorithms)*, 2
smaugt192 *(kpqc_algorithms)*, 2
smaugt256 *(kpqc_algorithms)*, 2

timer *(kpqc_algorithms)*, 2

verify_signature *(kpqc_operations)*, 3