ML-KEM for TLS 1.3

draft-connolly-tls-mlkem-key-agreement

https://datatracker.ietf.org/doc/draft-connolly-tls-mlkem-key-agreement/
https://github.com/dconnolly/draft-connolly-tls-mlkem-key-agreement
A pure-PQ ciphersuite for TLS 1.3

- No purely post-quantum ciphersuites
- Fills in the other side of draft-ietf-tls-hybrid-design
- Needed because there are no documents that describe KEM-only key agreement in TLS
- If PQ-only works for your applications, clean key agreement, no hybrid duplicate shares or mixing and matching logic
- ML-KEM-1024 supports FIPS users who need to comply with the CNSA 2.0 draft
- I want to be able to do it 🤡
New NamedGroups: MLKEM768, MLKEM1024

```c
enum {

    ..., 

    /* ML-KEM Key Agreement Methods */
    mlkem768(0x0768),
    mlkem1024(0x1024)

    ..., 

} NamedGroup;
```
Client sends encaps key, server replies with ciphertext

```
struct {
    NamedGroup group;
    opaque key_exchange<1..2^16-1>;
} KeyShareEntry;
```

These are transmitted in the `extension_data` fields of `KeyShareClientHello` and `KeyShareServerHello` extensions:

```
```
KEM shared secret is input to Handshake Secret derivation

```
shared_secret -> HKDF-Extract = Handshake Secret

Derive-Secret(., "derived", "")

-----------
```

```
FAQs

Can’t you just get a codepoint instead of adopting a document?
There are no documents that describe KEM-only key agreement in TLS, so not really

Should this be Recommended = Y? MTI?
No. It should be optional

What about PQ signatures too?
Nah; this is easy, adopting PQ signatures is looking like a hard design problem and are out of scope here

Isn’t this too early?
Considering the long timelines for adoption, I don’t think so

Just use hybrid!
Some users cannot use hybrid, and some will not do more than one PQ transition. Having a PQ-only option seems necessary eventually, let’s make a start

I don’t trust PQ crypto, it’s too young!
CRYSTALS-Kyber was published 7 years ago, LWE schemes are older than that. Elliptic curves were first published in ~1985, wg adopted for TLS in 1998, NIST curves standardized in 1999, RFC in 2006! We’re older and wiser now, but even so this timeline seems in line with major crypto assumption changes in TLS in the past.
Questions?
ML-KEM for TLS 1.3

draft-connolly-tls-mlkem-key-agreement

https://datatracker.ietf.org/doc/draft-connolly-tls-mlkem-key-agreement/
https://github.com/dconnolly/draft-connolly-tls-mlkem-key-agreement