Split-key PRFs and a new notion for hybrid KEM security

NTNU
Norwegian University of Science and Technology

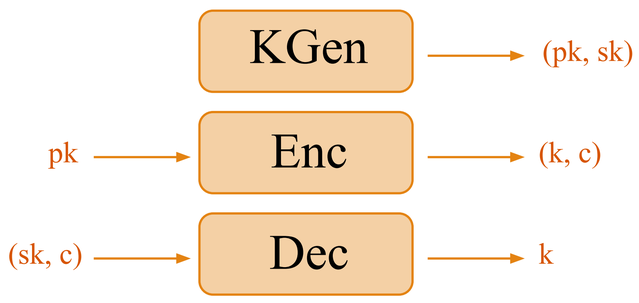
Lise Millerjord, Douglas Stebila, and Camryn Steckel

UNIVERSITY OF WATERLOO

Background

What is a KEM?

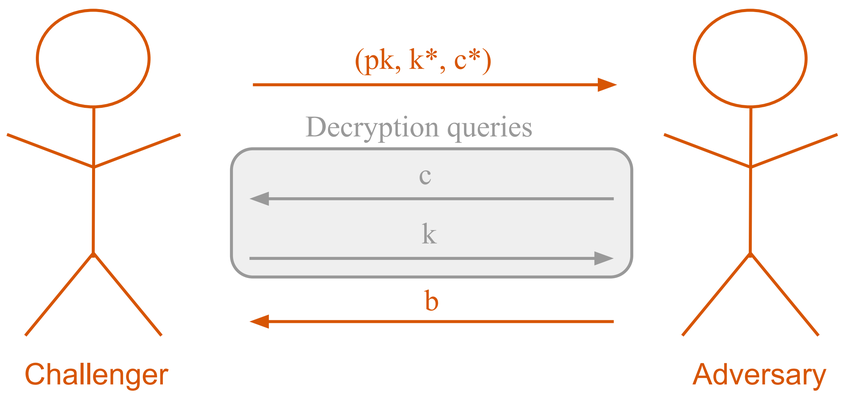
A *key encapsulation mechanism (KEM)* is a tuple of algorithms (KGen, Enc, Dec), that is used to establish a shared secret between two parties.



If Alice generates public and secret keys, then Bob can run the (randomized) encapsulation algorithm on her public key, to produce a shared secret *k* and ciphertext *c*. He can then send Alice *c*, which she can decapsulate using her secret key, to obtain *k*.

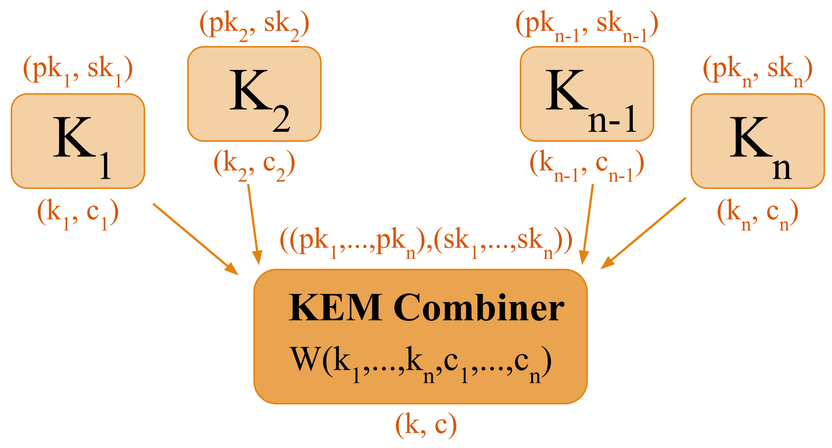
IND-CCA Security for KEMs

We use the notion of *indistinguishability against chosen ciphertext attacks* to define security for KEMs. This is modeled as a game between a challenger and an adversary. The challenger generates a public/secret key pair, and encapsulates the public key to produce the challenge shared secret and ciphertext. They give the adversary the public key, challenge ciphertext, and either the challenge shared secret or a randomly generated one. The adversary's goal is to figure out which one. To do this, they are allowed to query the challenger for the decapsulations of any ciphertexts (other than the challenge ciphertext). Finally, the adversary returns a bit indicating whether they think that the challenge shared secret was real or random.



KEM Combiners

A *KEM combiner* produces a shared secret and ciphertext that are based off of the outputs of multiple "ingredient" KEMs. A KEM combiner works by first running the encapsulation algorithms of the ingredient KEMs, then inputting the resulting shared secrets and ciphertexts into a "core function" to produce a single shared secret and ciphertext.



Security of the KEM combiner is dependent on which core function and ingredient KEMs. Ultimately, the KEM combiner should be secure if at least one of the ingredient KEMs is secure.

Core functions of KEM combiners

In [GHP18], they showed that if the core function of a KEM combiner is a split-key PRF, then the KEM combiner satisfies IND-CCA security for KEMs.

What is a split-key PRF?

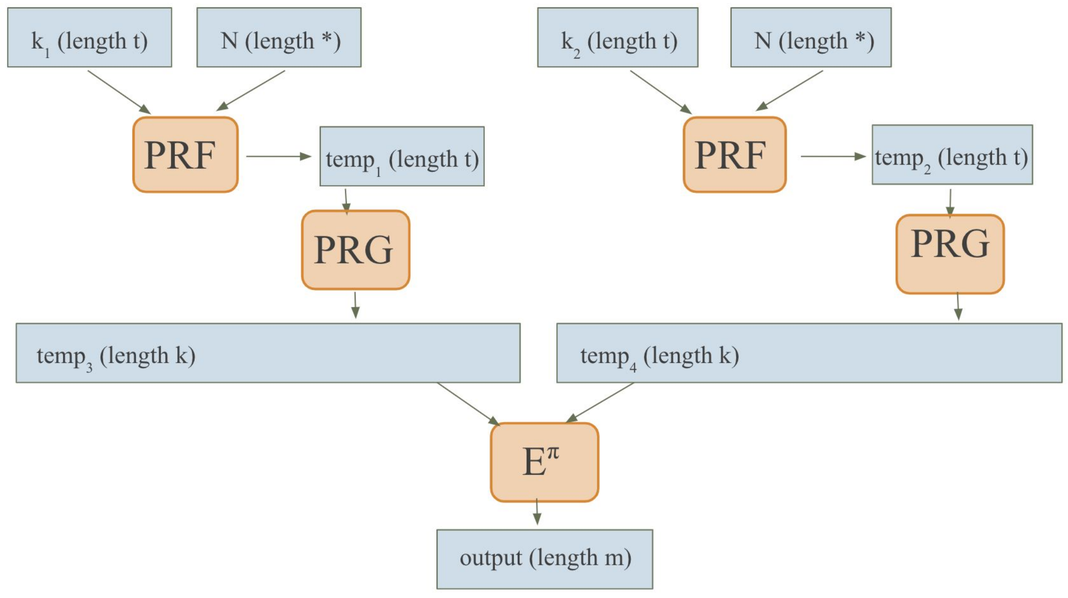
A *split-key pseudorandom function* is a pseudorandom function that takes in multiple keys (and a nonce) as input, and is pseudorandom in each of its input keys.

In [GHP18], they provided a few different constructions of split-key PRFs. Most were proven secure in the random oracle model, and one was secure in the standard model. However, this construction relies on XOR-ing keys together, which makes some people are uncomfortable. We provide a split-key PRF that is secure in the standard model, and does not rely on XOR-ing keys together.

What is a *t-resilient* extractor?

An *extractor* $E^{\pi}: \{0,1\}^n \rightarrow \{0,1\}^m$ on public parameter π takes input from a high entropy source and produces an output that is statistically close to uniform. An extractor is *t-resilient* if it still produces output statistically close to uniform, even if the adversary controls *t* bits of entropy in the input.

Our construction



$$W(k_1, k_2, N) = E^{\pi}(G(PRF(k_1, N)) \| G(PRF(k_2, N)))$$

Our construction requires a (regular) PRF, a PRG, and a *t*-resilient extractor. We assume that the adversary controls *t* of the input keys.

First, each input key is put into the PRF, along with the nonce, to produce an application-specific key. We only really need to do this for the "good" (i.e., non-adversarially-controlled) key, but since we don't know which that is, we do it for all of them.

Each of the application-specific keys is then put into a PRG to expand it, since the *t*-resilient extractor requires long keys as input.

Finally, the expanded keys are put into the *t*-resilient extractor to produce the final output.

Hybrid security

What if we use "regular" IND-CCA security?

Applying the definition of IND-CCA security to a KEM combiner gives the adversary access to a decryption oracle for the KEM combiner… but not for the ingredient KEMs! If a user reuses the same public key for both an ingredient KEM and the KEM combiner, then they might be able to gain information about the KEM combiner by interacting with the ingredient KEM. Our security definition does not capture this case.

Bad solution to the problem

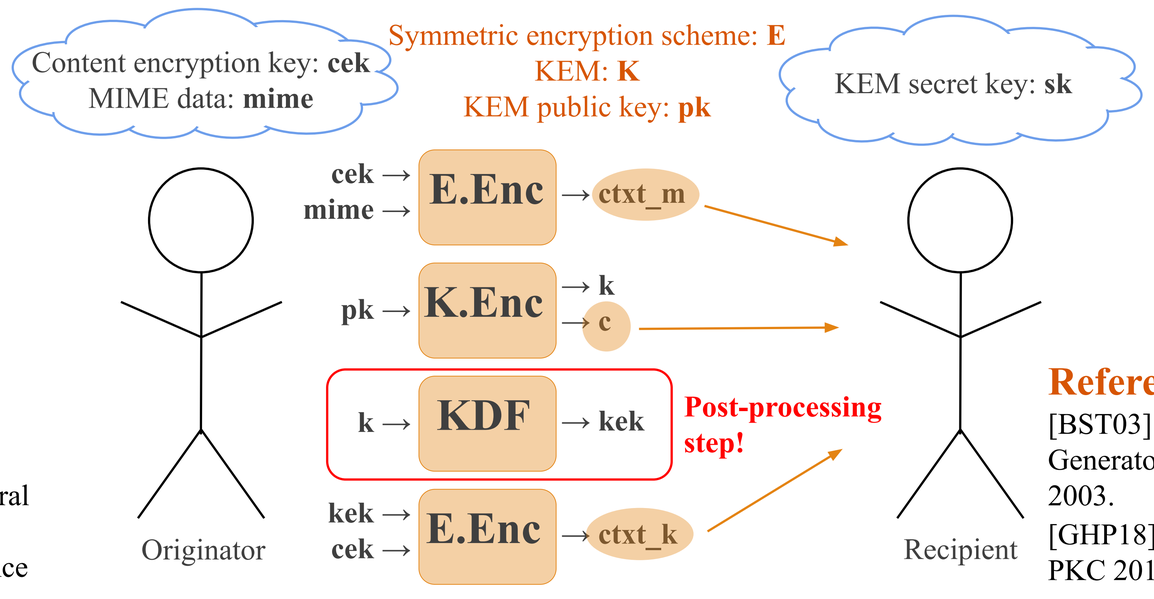
The natural solution would be to expand the security definition to give the adversary unrestricted access to decapsulation oracles for each of the ingredient KEMs. However, this allows them to trivially win the experiment, since they can simulate decapsulation of the challenge ciphertext by querying the ingredient decryption oracles every time they need to use the secret key.

Good solution to the problem

Instead, we can give the adversary unrestricted access to *post-processed* versions of decapsulation oracles for each of the ingredient KEMs. When you think about it, this is also a pretty natural definition, as the core function of a KEM combiner is already a post-processing function on the ingredient KEMs. The only difference here is that the post-processing function will change depending on whether you are interacting with an individual or combined KEM.

Hybrid security model in the wild: S/MIME

Secure/Multipurpose Internet Mail Extensions (S/MIME) is the Internet standard that provides authentication, message integrity, non-repudiation of origin, and confidentiality of MIME data. In the "Composite KEMs in S/MIME" draft, they RECOMMEND (but not REQUIRE) to not reuse keys across individual and combined KEMs. So, this instance is captured by our new security model, but not by the regular IND-CCA definition for KEMs.



Is S/MIME secure?

Yes! One of our main results is that if a KEM combiner has a split-key PRF as its core function, regular PRFs as the post-processing functions for the ingredient KEMs, and at least one of the ingredient KEMs is secure, then the KEM combiner is secure under our model. Since KDFs are made from PRFs, this theorem applies to S/MIME.

References

[BST03] Barak, Shaltiel, Tromer. True Random Number Generators Secure in a Changing Environment. CHES 2003.

[GHP18] Giacon, Heuer, Poettering. KEM Combiners. PKC 2018.

[OGPKS25] Ounsworth et al., 2025. https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-kem/