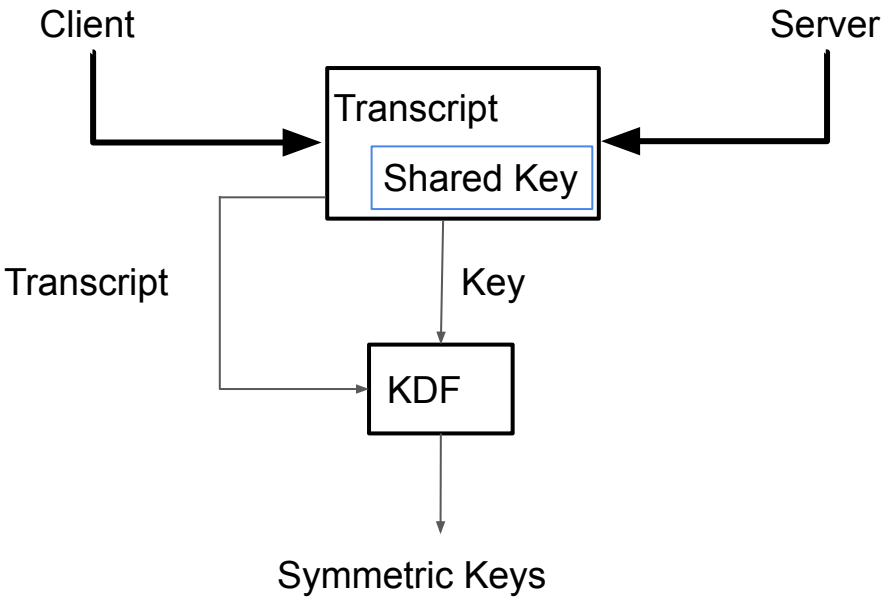
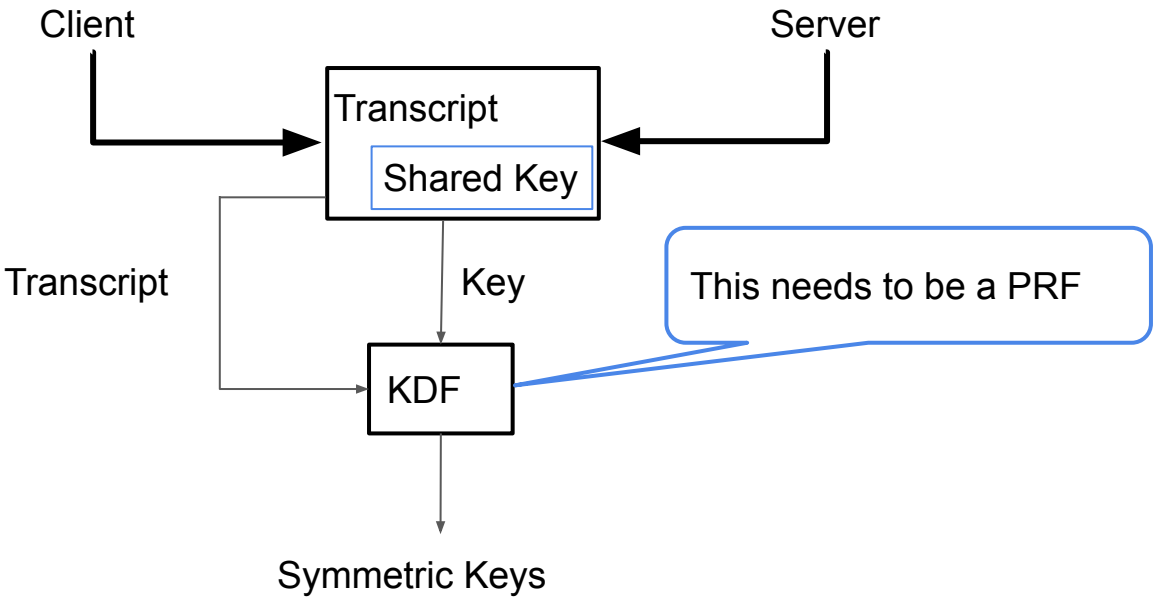
A dual-PRF construction

Nimrod Aviram, Benjamin Dowling, Ilan Komargodski,
Kenny Paterson, Eyal Ronen, Eylon Yogev

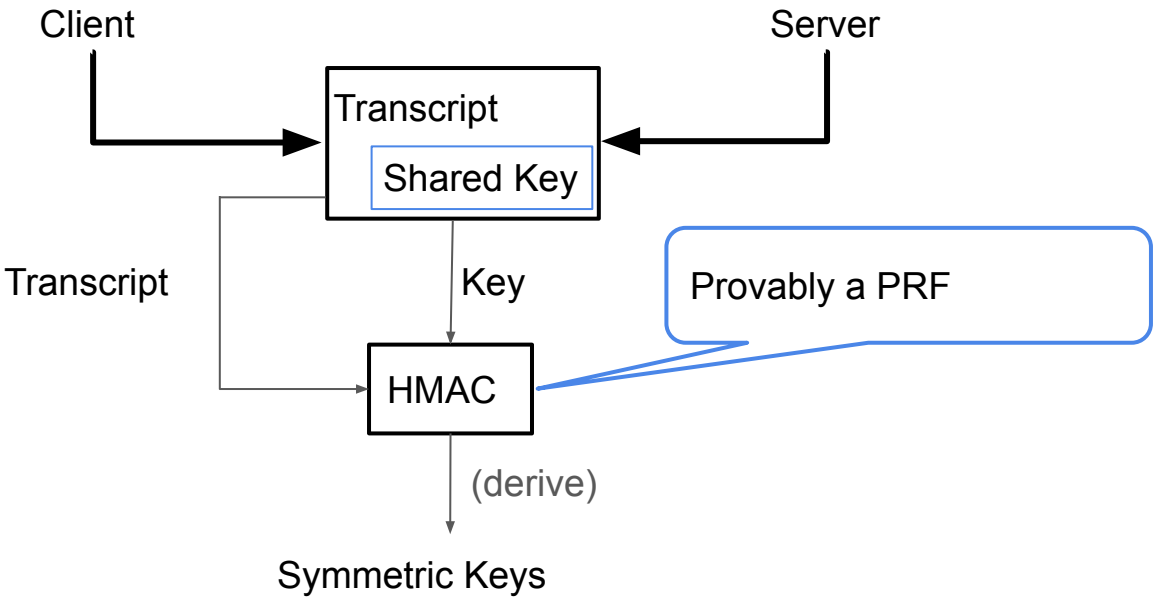
Modern Protocols in a nutshell



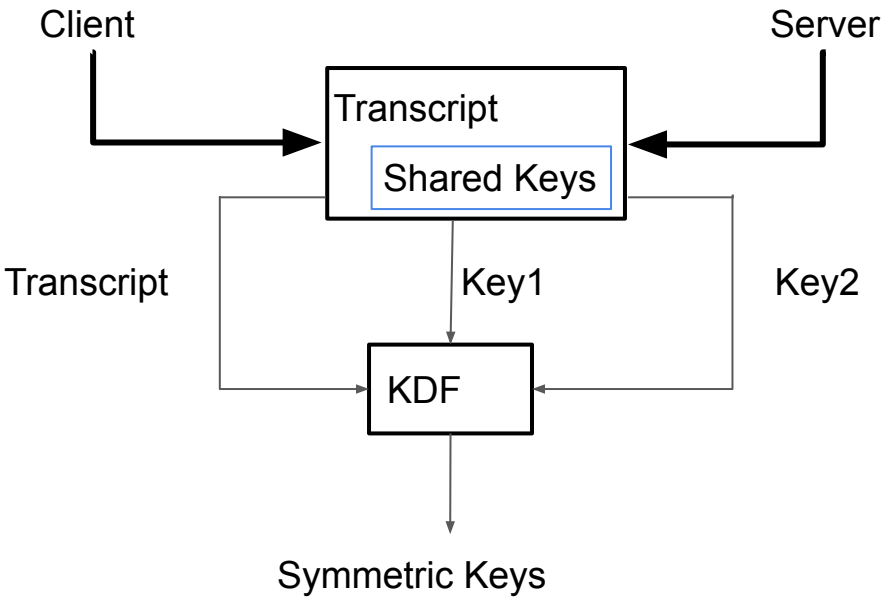
Modern Protocols in a nutshell



Modern Protocols in a nutshell



Modern Protocols - in reality



We Often Use Two Keys

- TLS 1.3, DHE+PSK (resumption)
- Hybrid Key Exchange in TLS 1.3 (Classical + Post-Quantum)
- Signal Double Ratchet

A KDF Taking Two Keys

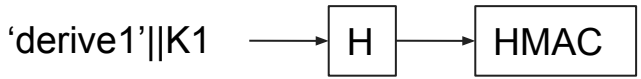
- General approach: k = Combine(key1, key2); output = HMAC(k, transcript)
  - Both for existing constructions, and our proposal.
- “Takes two keys” = Dual-PRF: PRF when keyed by either output.
  - Attacker may realistically control either key1 or key2, other key is random.
  - We do not know in advance which key is controlled by the attacker.

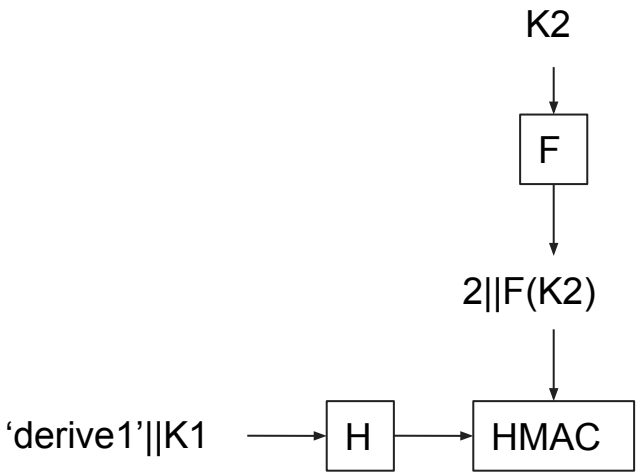
A KDF Taking Two Keys

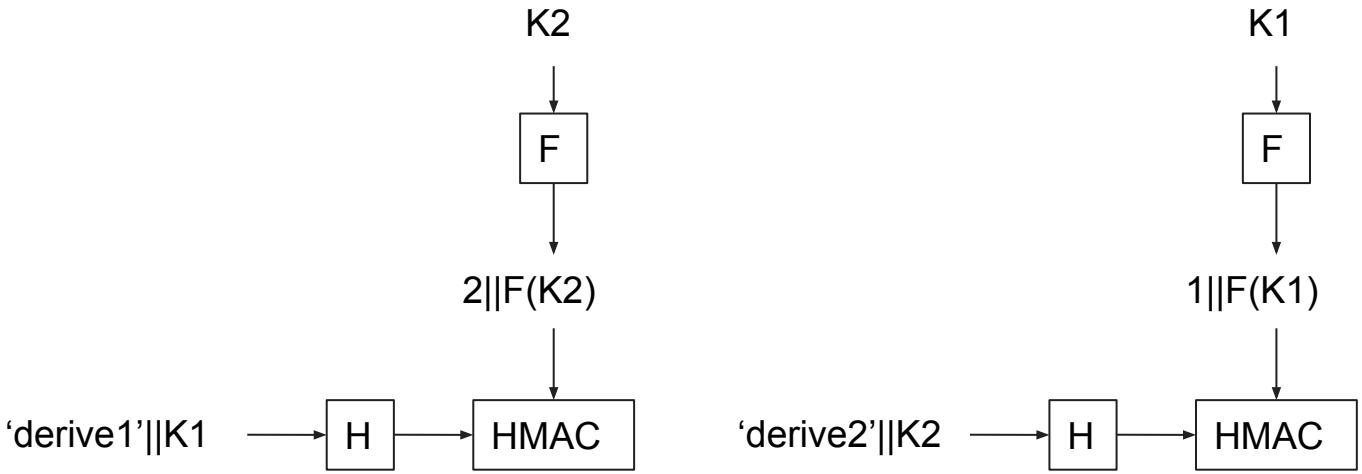
- General approach: k = Combine(key1, key2); output = HMAC(k, transcript)
  - Both for existing constructions, and our proposal.
- “Takes two keys” = Dual-PRF: PRF when keyed by either output.
  - Attacker may realistically control either key1 or key2, other key is random.
  - We do not know in advance which key is controlled by the attacker.
- Can we use HMAC as the key combiner?
- HMAC is generally *not* a dual-PRF.
  - Never intended or proved to satisfy this.
  - Definitely not a dual-PRF if underlying hash function is not CR.

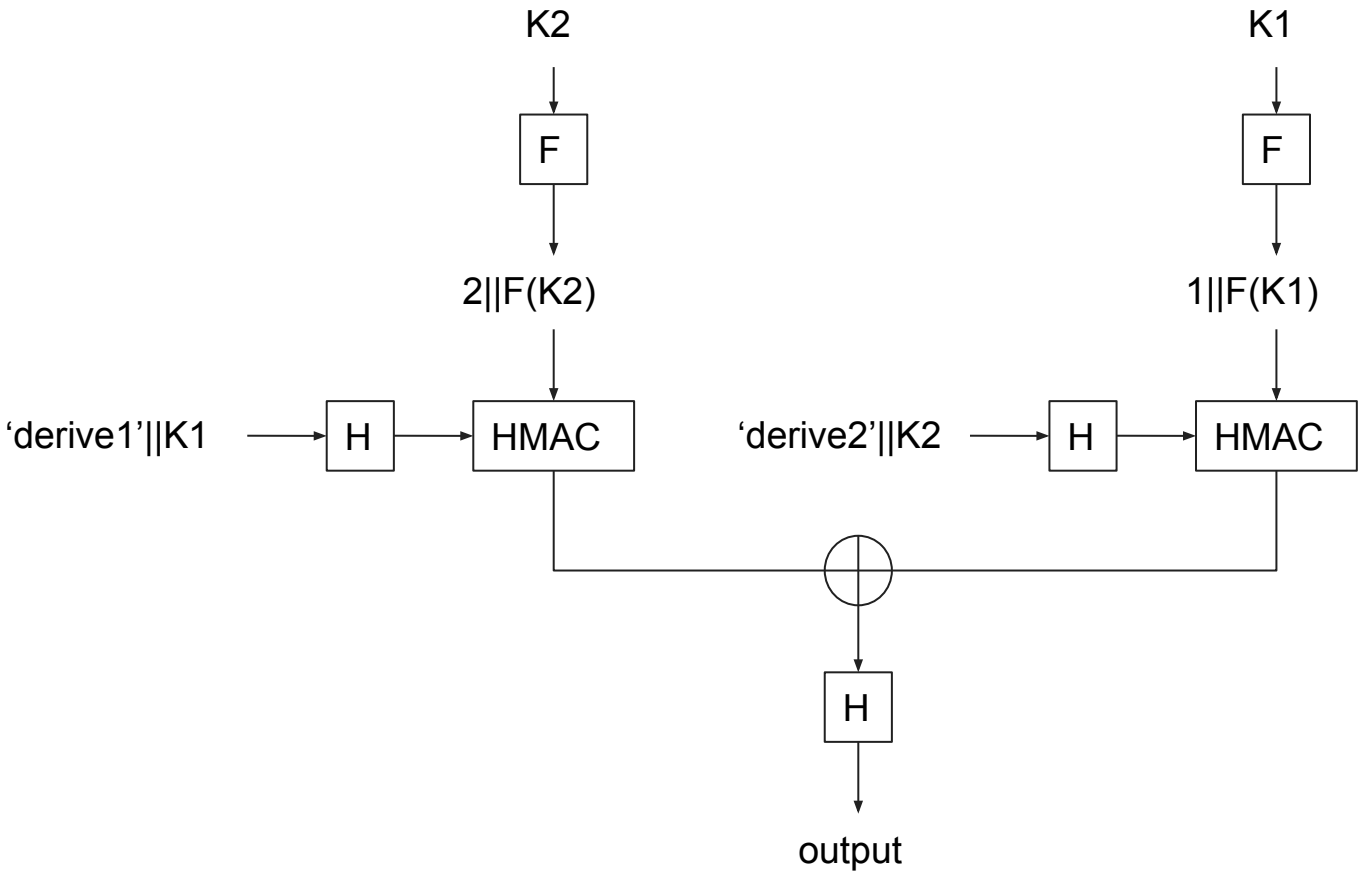
Our construction

- Uses an underlying standard hash function, not necessarily collision-resistant
- Fully practical: “symmetric crypto”, cheap to compute.
- Construction likely to be used alongside asymmetric crypto, so relative cost is minimal.
- Security proof in [ePrint/2022/065]

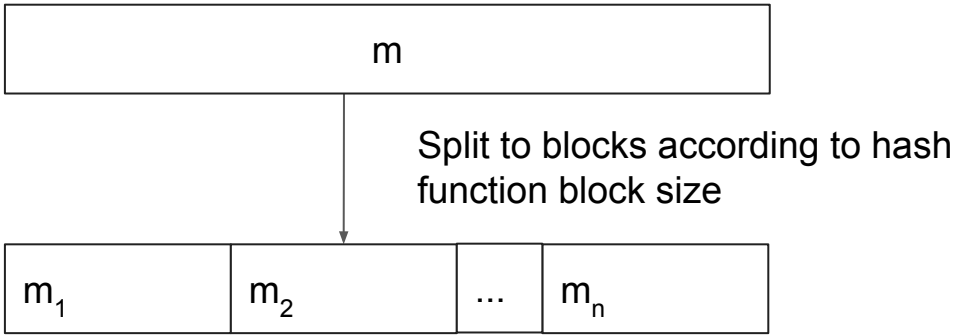

‘derive1’||K1
H
HMAC


K2
F
2||F(K2)
‘derive1’||K1
H
HMAC

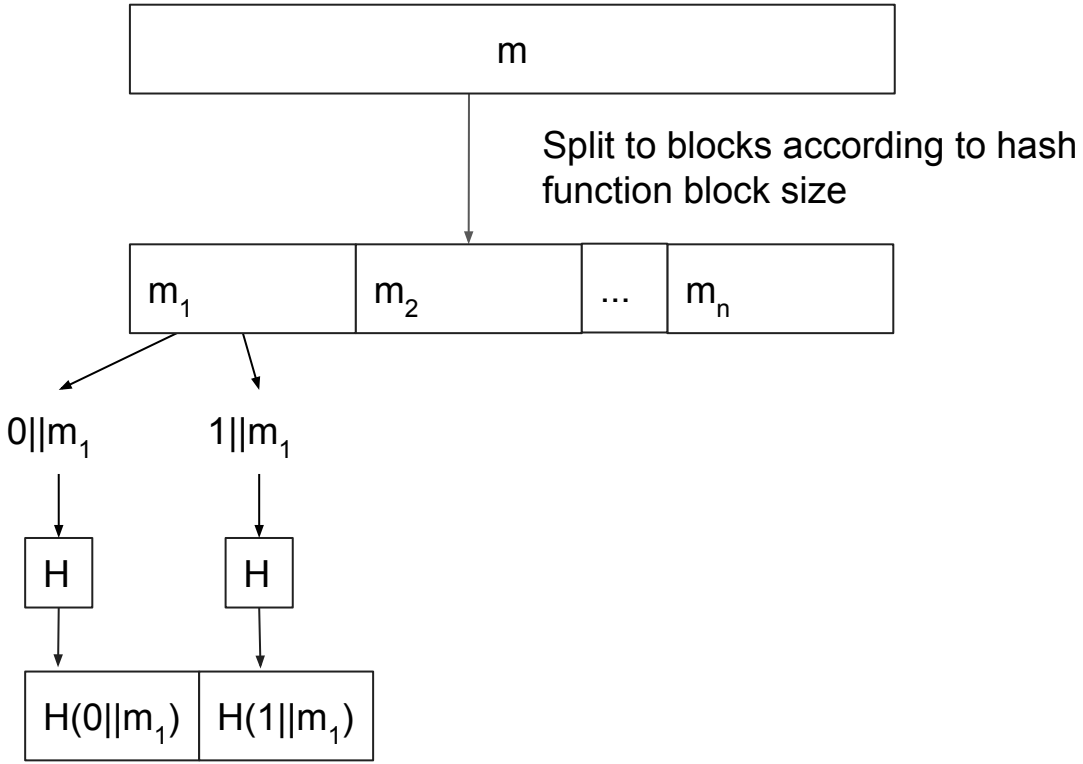

K2
F
2||F(K2)
'derive1'||K1
H
HMAC
K1
F
1||F(K1)
'derive2'||K2
H
HMAC


K2
K1
F
F
2||F(K2)
1||F(K1)
‘derive1’||K1
H
HMAC
‘derive2’||K2
H
HMAC
H
output

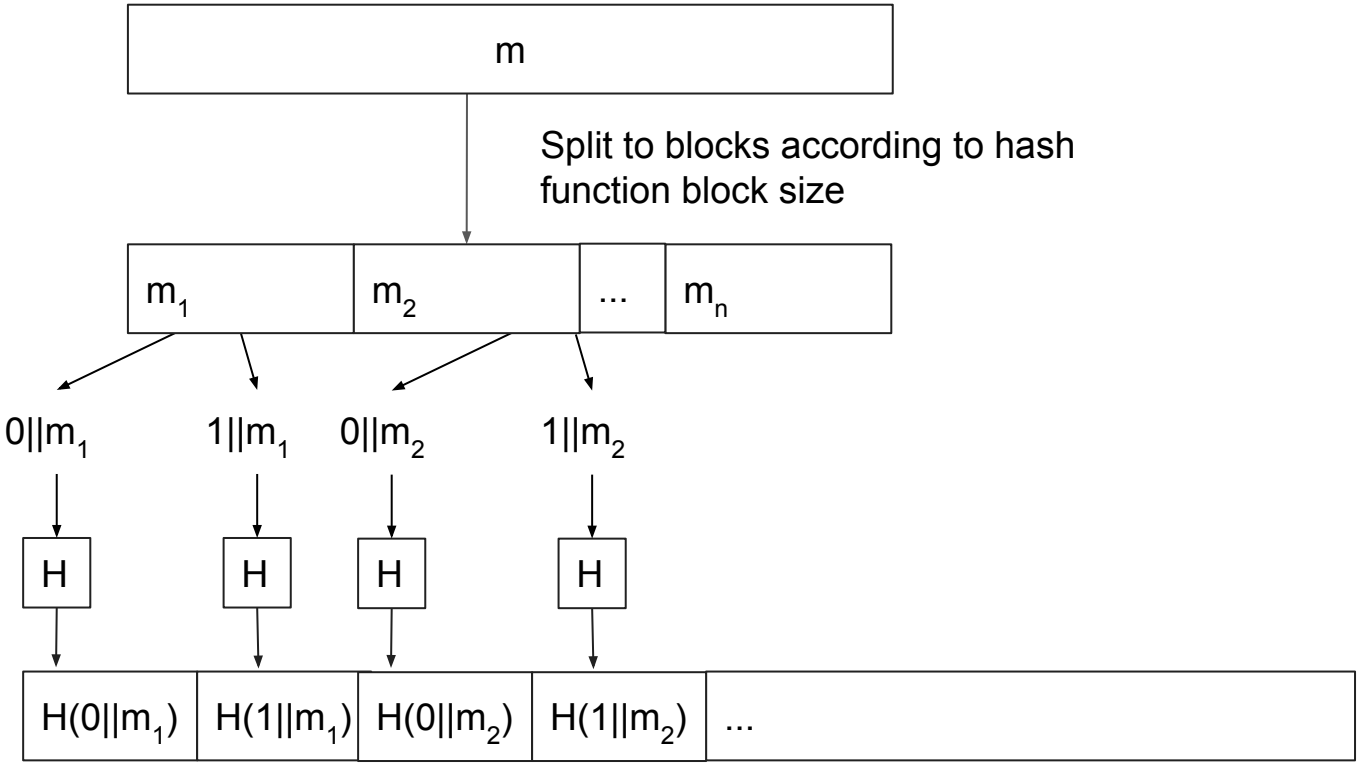
New “Expanding Function” F



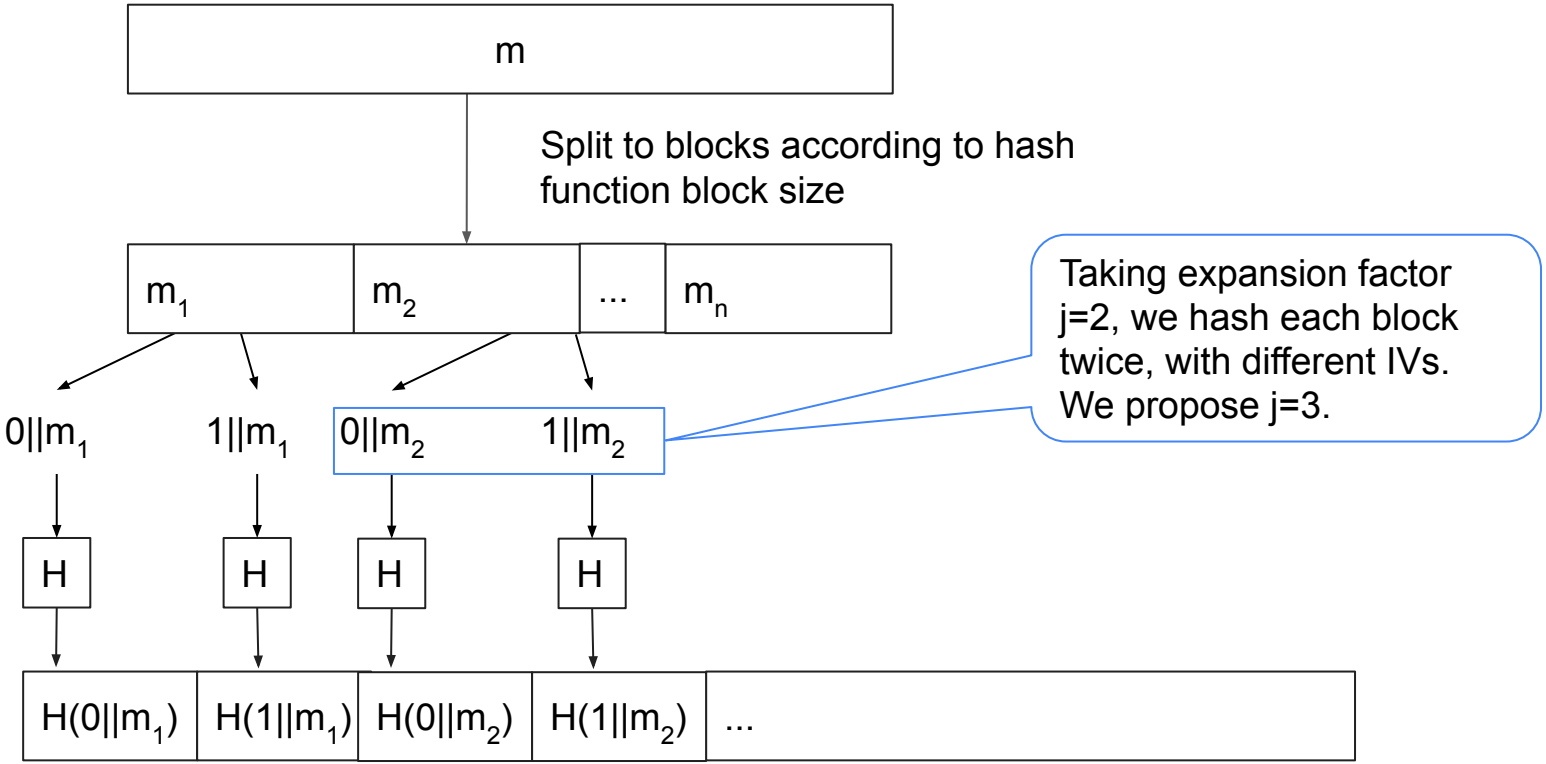
New “Expanding Function” F



New “Expanding Function” F



New “Expanding Function” F



Our construction (cont.)

- H(HMAC(key=H1(k1), data=2||F(k2)) xor HMAC(key=H2(k2), data=1||F(k1)))
  - H1, H2: Hash with prefix, see diagram on previous slides
- F(k) is an expanding injective OWF:
  - Split k into blocks, according to block size of H. k = k1||k2||...||kn
  - Let j≅3 denote an “expansion factor”.
- F(k) = H(0||k1)||H(1||k1)||...||H(j||k1)||H(0||k2)||H(1||k2)||...||H(j||k2)||H(0||Kn)||H(1||Kn)||...||H(j||kn)

Choosing The Expansion Factor

- Why expansion factor j=3?
- Expansion required to ensure OWF is injective. More expansion -> longer output -> higher chance of injectivity.
- KDF is “ossification surface”, hard to upgrade when cryptanalysis (inevitably) improves.
- Conservatively assume underlying hash function is as broken as MD5 is today. Then j=2 is enough, j=3 is robust security margin.
    - (j=2: plausible choice, but seems risky)
    - (higher j, e.g. j=5 also plausible, but seems like overkill)

Benchmarks

- Construction much cheaper than asymmetric crypto.
- Our construction: 7.1 µsec/call.
- HKDF (with concatenated keys): 1.3 µsec/call.
  - Overhead is 5.8 µsec/call.
- Asymmetric crypto:
  - X25519, twice per connection: 44.7 µsec/exponentiation.
  - Secp256r1 ECDSA: 79 / 24.6 µsec for verify/sign.
  - NTRU-HRSS: 17.6 / 11.2 µsec for decaps./encaps.
- Even with only two exponentiations and signature, overhead is only 5%.
  - Likely lower for most use cases, e.g. with verification, NTRU-HRSS.
  - (If higher for some use cases, can consider j=2.)

Key Combiners in Practice

- TLS 1.3 DHE+PSK, Signal Double Ratchet: Combine keys using HKDF/HMAC.
- Hybrid TLS 1.3, [ETSI], [NIST]: Concatenate keys, proceed as usual.
- TLS 1.3, DHE Only: Keyed through “message” input of HMAC (!)
- Standardized dual-PRF would make standards more robust, also with a single key.
- We are thinking of writing an Internet Draft for this technique; would the RG find this useful?

Thanks!

Questions?

References

- https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/
- [ePrint/2022/065] Practical (Post-Quantum) Key Combiners from One-Wayness and Applications to TLS. Nimrod Aviram, Benjamin Dowling, Ilan Komargodski, Kenneth G. Paterson, Eyal Ronen, Eylon Yogev
- [HMAC] Bellare, Mihir. "New proofs for NMAC and HMAC: Security without collision-resistance." Crypto 2006.
- [BelLyn] Bellare, Mihir, and Anna Lysyanskaya. "Symmetric and Dual PRFs from Standard Assumptions: A Generic Validation of an HMAC Assumption." IACR Cryptol. ePrint Arch. 2015 (2015): 1198.
- [ETSI] https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?wki_id=56901
- [NIST] https://csrc.nist.gov/publications/detail/sp/800-56c/rev-2/final