Improved ChaCha-Based AEAD

Jean Paul Degabriele, Jan Gilcher, Jérôme Govinden, Kenneth G. Paterson

Technology Innovation Institute

ETH zürich

TECHNISCHE UNIVERSITÄT DARMSTADT

24th July 2025 IETF 123

ChaCha-Poly1305 (RFC 8439)

- ChaCha-Poly1305 is the second most popular AEAD scheme after **AES-GCM**.
- The default in **OpenSSH**, **Wireguard**, **OTR v4**, and supported in **TLS 1.3**, **DTLS**, **QUIC**, **IPsec**, **MLS** and many cryptographic libraries.

ChaCha-Poly1305 (RFC 8439)

- ChaCha-Poly1305 is the second most popular AEAD scheme after **AES-GCM**.
- The default in **OpenSSH**, **Wireguard**, **OTR v4**, and supported in **TLS 1.3**, **DTLS**, **QUIC**, **IPsec**, **MLS** and many cryptographic libraries.
- Both ChaCha and Poly1305 were **designed** by Bernstein **as independent primitives.** They were later combined into an AEAD scheme in 2015 by Nir and Langley in RFC 7539 (now RFC 8439).
- Attractive Features: **very efficient** (especially on generic hardware), is **not based on AES** (alternative trust assumption), and ChaCha is an **established and well-studied** streamcipher.

ChaCha-Poly1305 (RFC 8439)

- ChaCha-Poly1305 is the second most popular AEAD scheme after **AES-GCM**.
- The default in **OpenSSH**, **Wireguard**, **OTR v4**, and supported in **TLS 1.3**, **DTLS**, **QUIC**, **IPsec**, **MLS** and many cryptographic libraries.
- Both ChaCha and Poly1305 were **designed** by Bernstein **as independent primitives.** They were later combined into an AEAD scheme in 2015 by Nir and Langley in RFC 7539 (now RFC 8439).
- Attractive Features: **very efficient** (especially on generic hardware), is **not based on AES** (alternative trust assumption), and ChaCha is an **established and well-studied** streamcipher.
- Widely perceived as the **main alternative to GCM** and other **AES-based AEAD**.

Then Why Replace it?

- We can improve **Performance** and **Security** in numerous ways, for example...
- ChaCha and Poly1305 have **mismatching security levels** (**256 vs 106/$\log_2(L_{max})$**) where the **polynomial hash is the bottleneck** for multi-user AEAD security [eprint/2023/085].

Then Why Replace it?

- We can improve **Performance** and **Security** in numerous ways, for example...
- ChaCha and Poly1305 have **mismatching security levels** (**256 vs 106/$\log_2(L_{max})$**) where the **polynomial hash is the bottleneck** for multi-user AEAD security [eprint/2023/085].
- **CPUs have changed** – Poly1305 was optimised for **floating-point arithmetic** on **32-bit** architectures. However, today integer arithmetic is faster and CPUs are mostly 64-bit, making it a **wasteful design**.
- **AEAD has changed** – The requirements on AEAD have evolved, e.g. **commitment security**, **nonce hiding**, **RUP security**, see **RFC 9771** for a more extensive list.
- **GCM is changing** – NIST is looking to update GCM to increase its **security level** and **nonce size**, and standardise **new AES modes** with advanced security (Rijndael256-GCM, DNDK-GCM, Accordion).

Then Why Replace it?

- We can improve **Performance** and **Security** in numerous ways, for example...
- ChaCha and Poly1305 have **mismatching security levels** (**256 vs 106/$\log_2(L_{max})$**) where the **polynomial hash is the bottleneck** for multi-user AEAD security [eprint/2023/085].
- **CPUs have changed** – Poly1305 was optimised for **floating-point arithmetic** on **32-bit** architectures. However, today integer arithmetic is faster and CPUs are mostly 64-bit, making it a **wasteful design**.
- **AEAD has changed** – The requirements on AEAD have evolved, e.g. **commitment security**, **nonce hiding**, **RUP security**, see **RFC 9771** for a more extensive list.
- **GCM is changing** – NIST is looking to update GCM to increase its **security level** and **nonce size**, and standardise **new AES modes** with advanced security (Rijndael256-GCM, DNDK-GCM, Accordion).

ChaCha-based AEAD needs to keep up if it is to remain a viable alternative.

RFC 9771 and ChaCha-Based AEAD

- Comparing with RFC 9771 we observe that ChaCha-Poly1305 **lacks many of the listed properties**.
- Of course, no single AEAD scheme can achieve them all and a **dichotomy** exists separating **single-pass** from **multi-pass** AEAD.

Table of Contents

RFC 9771 and ChaCha-Based AEAD

- Comparing with RFC 9771 we observe that ChaCha-Poly1305 **lacks many of the listed properties**.
- Of course, no single AEAD scheme can achieve them all and a **dichotomy** exists separating **single-pass** from **multi-pass** AEAD.
- ChaCha-Poly1305 is a **single-pass scheme**, but it could be enhanced with **commitment security**, **variable tag size**, larger **nonce size**, improved **multi-user security**, better **performance**,...
- Additionally, it is also possible to define a **multi-pass** ChaCha-based AEAD scheme that provides **misuse-resistance**, **RUP security**, **variable tag size**, **nonce hiding**, etc**.**

What Can We Change?

- Note that **ChaCha** is the only component **requiring empirical cryptanalysis**.
- On the other hand, the **polynomial hash** and AEAD **construction** can be easily changed without invalidating any of the cryptanalysis (and trust) on ChaCha as a streamcipher.

What Can We Change?

- Note that **ChaCha** is the only component **requiring empirical cryptanalysis**.
- On the other hand, the **polynomial hash** and AEAD **construction** can be easily changed without invalidating any of the cryptanalysis (and trust) on ChaCha as a streamcipher.

- For context, on the right we observe the improvement in **AEAD performance** when replacing Poly1305 with our improved polynomial hash design (Poly1163).
- Source of Benchmarks: https://tinyurl.com/2s4zfab6

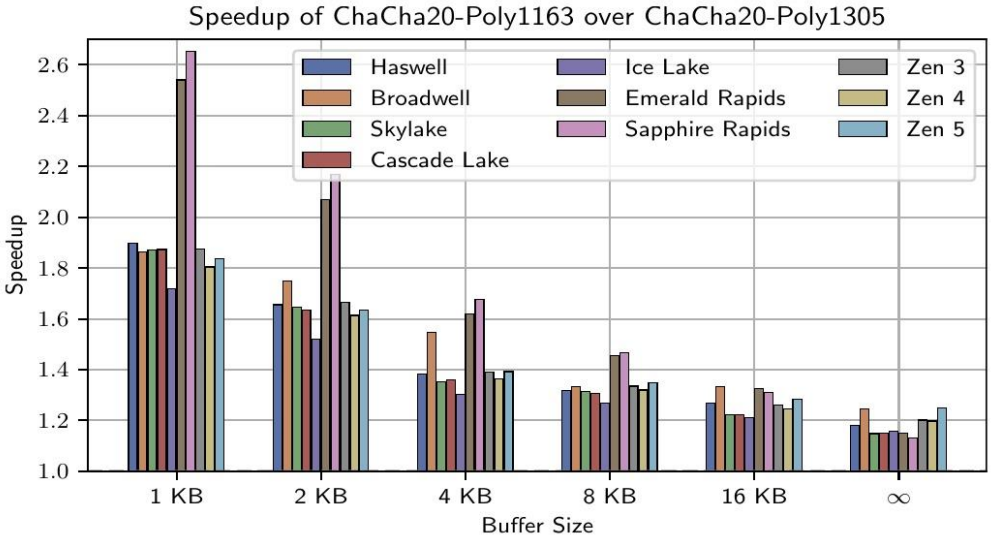

Figure 6.5: Speedup of our ChaCha20-Poly1163 implementations over OpenSSL's ChaCha20-Poly1305.

Discussion & Feedback

- Is there interest in working on this problem within CFRG?
- Which enhancements would your application/protocol benefit from?
- Is a multi-pass scheme viable for Internet applications?