Network Working Group   Y. Nir
Internet-Draft   Check Point
Intended status: Standards Track   March 31, 2014
Expires: October 2, 2014

ChaCha20 and Poly1305 and their use in IPsec
draft-nir-ipsecme-chacha20-poly1305-02

Abstract

This document describes the use of the ChaCha20 stream cipher along with the Poly1305 authenticator, combined into an AEAD algorithm for IPsec.

Table of Contents

1. Introduction

The Advanced Encryption Standard (AES - [FIPS-197]) has become the gold standard in encryption. Its efficient design, wide implementation, and hardware support allow for high performance in many areas, including IPsec VPNs. On most modern platforms, AES is anywhere from 4x to 10x as fast as the previous most-used cipher, 3-key Data Encryption Standard (3DES - [FIPS-46]), which makes it not only the best choice, but the only choice.

The problem is that if future advances in cryptanalysis reveal a weakness in AES, VPN users will be in an unenviable position. With the only other widely supported cipher being the much slower 3DES, it is not feasible to re-configure IPsec installations to use 3DES. [standby-cipher] describes this issue and the need for a standby cipher in greater detail.

This document proposes the ChaCha20 stream cipher as such a standby cipher in an AEAD construction with the Poly1305 authenticator for use with the Encapsulated Security Protocol (ESP - [RFC4303]). We call this ESP_ChaCha20-Poly1305. These algorithms are described in a separate document ([chacha_poly]). This document only describes the IPsec-specific things.

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. ESP_ChaCha20-Poly1305 for ESP

ESP_ChaCha20-Poly1305 is a combined mode algorithm, or AEAD. The construction follows the AEAD construction in section 2.7 of [chacha_poly]:

o The IV is 64-bit, and is used as part of the nonce.
o A 32-bit sender ID is prepended to the 64-bit IV to form the 96-bit nonce. For regular IPsec, this is set to all zeros. IPsec extensions that allow multiple senders, such as GDOI ([RFC6407]) or [RFC6054] may set this to different values.
o The encryption key is 256-bit.
o The Internet Key Exchange protocol (IKE - [RFC5996]) generates a bitstring called KEYMAT that is generated from a PRF. That KEYMAT is divided into keys for encryption, message authentication and whatever else is needed. For the ChaCha20 algorithm, 256 bits are used for the key. TBD: do we want an extra 32 bits as salt for the nonce like in GCM?

o The ChaCha20 encryption algorithm requires the following parameters: a 256-bit key, a 96-bit nonce, and a 32-bit initial block counter. For ESP we set these as follows:
  * The key is set to the key mentioned above.
  * The 96-bit nonce is formed from a concatenation of the 32-bit sender ID and the 64-bit IV, as described above.
  * The Initial Block Counter is set to one (1). The reason that one is used for the initial counter rather than zero is that zero is reserved for generating the one-time Poly1305 key (see below)
o As ChaCha20 is not a block cipher, no padding should be necessary. However, in keeping with the specification in RFC 4303, the ESP does have padding, so as to align the buffer to an integral multiple of 4 octets.
o The same key and nonce, along with a block counter of zero are passed to the ChaCha20 block function, and the top 256 bits of the result are used as the Poly1305 key. The nonce passed to the block function here is the same nonce that is used in ChaCha20, including the 32-bit Sender ID bits, and the key passed is the same as the encryption key.
o Finally, the Poly1305 function is run on the data to be authenticated, which is, as specified in section 2.7 of [chacha_poly] a concatenation of the following in the below order:
  * The Authenticated Additional Data (AAD) - see Section 2.1.
  * The AAD length in bytes as a 32-bit network order quantity.
  * The ciphertext
  * The length of the ciphertext as a 32-bit network order quantity.
o The 128-bit output of Poly1305 is used as the tag. All 16 bytes

are included in the packet.

The encryption algorithm transform ID for negotiating this algorithm in IKE is TBA by IANA.

2.1. AAD Construction

The construction of the Additional Authenticated Data (AAD) is similar to the one in [RFC4106]. For security associations (SAs) with 32-bit sequence numbers the AAD is 8 bytes: 4-byte SPI followed by 4-byte sequence number ordered exactly as it is in the packet. For SAs with ESN the AAD is 12 bytes: 4-byte SPI followed by an 8-byte sequence number as a 64-bit network order integer.

3. Security Considerations

The ChaCha20 cipher is designed to provide 256-bit security.

The Poly1305 authenticator is designed to ensure that forged messages are rejected with a probability of 1-(n/(2^102)) for a 16n-byte message, even after sending 2^64 legitimate messages, so it is SUF-CMA in the terminology of [AE].

The most important security consideration in implementing this draft is the uniqueness of the nonce used in ChaCha20. The nonce should be selected uniquely for a particular key, but unpredictability of the nonce is not required. counters and LFSRs are both acceptable ways of generating unique nonces, as is encrypting a counter using a 64-bit cipher such as DES. Note that it is not acceptable to use a truncation of a counter encrypted with a 128-bit or 256-bit cipher, because such a truncation may repeat after a short time.

Another issue with implementing these algorithms is avoiding side channels. This is trivial for ChaCha20, but requires some care for Poly1305. Considerations for implementations of these algorithms are in the [chacha_poly] document.

4. IANA Considerations

IANA is requested to assign one value from the IKEv2 "Transform Type 1 - Encryption Algorithm Transform IDs" registry, with name ESP_ChaCha20-Poly1305, and this document as reference.

5. Acknowledgements

All of the algorithms in this document were designed by D. J. Bernstein. The AEAD construction was designed by Adam Langley. The author would also like to thank Adam for helpful comments, as well as Yaron Sheffer for telling me to write the algorithms draft.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

[RFC6054] McGrew, D. and B. Weis, "Using Counter Modes with Encapsulating Security Payload (ESP) and Authentication Header (AH) to Protect Group Traffic", RFC 6054, November 2010.

[chacha_poly] Langley, A. and Y. Nir, "ChaCha20 and Poly1305 for IETF protocols", draft-nir-cfrg-chacha20-poly1305-01 (work in progress), January 2014.

6.2. Informative References

[AE] Bellare, M. and C. Namprempre, "Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm",

<http://cseweb.ucsd.edu/~mihir/papers/oem.html>.

[FIPS-197]
National Institute of Standards and Technology, "Advanced Encryption Standard (AES)", FIPS PUB 197, November 2001, <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.

[FIPS-46] National Institute of Standards and Technology, "Data Encryption Standard", FIPS PUB 46-2, December 1993, <http://www.itl.nist.gov/fipspubs/fip46-2.htm>.

[RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", RFC 4106, June 2005.

[RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, October 2011.

[standby-cipher]
McGrew, D., Grieco, A., and Y. Sheffer, "Selection of Future Cryptographic Standards", draft-mcgrew-standby-cipher (work in progress).

Author's Address

Yoav Nir
Check Point Software Technologies Ltd.
5 Hasolelim st.
Tel Aviv 6789735
Israel

Email: ynir.ietf@gmail.com