Network Working Group   Y. Nir
Internet-Draft   Check Point
Intended status: Standards Track   April 25, 2015
Expires: October 27, 2015

              ChaCha20, Poly1305 and their use in IKE & IPsec
                 draft-ietf-ipsecme-chacha20-poly1305-03

Abstract

   This document describes the use of the ChaCha20 stream cipher along
   with the Poly1305 authenticator, combined into an AEAD algorithm for
   the Internet Key Exchange protocol (IKEv2) and for IPsec.

Status of This Memo

Copyright Notice

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 - [SP800-67]). 3DES also has a
64-bit block, which means that the amount of data that can be
encrypted before rekeying is required is not great. These reasons
make AES 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 Authenticated Encryption with Associated Data (AEAD)
construction with the Poly1305 authenticator for use with the
Encapsulated Security Protocol (ESP - [RFC4303]) and the Internet Key
Exchange Protocol (IKEv2 - [RFC7296]). The 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. ChaCha20 & Poly1305 for ESP

   AEAD_CHACHA20_POLY1305 is a combined mode algorithm, or AEAD. The
   construction follows the AEAD construction in section 2.8 of
   [chacha_poly]:

   o The Initialization Vector (IV) is 64-bit, and is used as part of
      the nonce. The IV MUST be unique for each invocation for a
      particular SA but does not need to be unpredictable. The use of a
      counter or a linear feedback shift register (LFSR) is RECOMMENDED.
   o A 32-bit Salt is prepended to the 64-bit IV to form the 96-bit
      nonce. The salt is fixed per SA and it is not transmitted as part
      of the ESP packet..
   o The encryption key is 256-bit.
   o The Internet Key Exchange protocol generates a bitstring called
      KEYMAT using a pseudo-random function (PRF). That KEYMAT is
      divided into keys for encryption, message authentication and
      whatever else is needed. For the ChaCha20-poly1305 algorithm, 256
      bits are used for the key, and a subsequent 32 bits are used for
      the Salt.

   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:

   o The key is set as mentioned above.
   o The 96-bit nonce is formed from a concatenation of the 32-bit Salt
      and the 64-bit IV, as described above.
   o 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)

   As the ChaCha20 block function is not applied directly to the
   plaintext, 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.

   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
   Salt, and the key passed is the same as the encryption key.

   Finally, the Poly1305 function is run on the data to be
   authenticated, which is, as specified in section 2.8 of [chacha_poly]
   a concatenation of the following in the below order:

   o The Authenticated Additional Data (AAD) - see Section 2.1.

o Padding that rounds the length up to 16 bytes. This is 4 or 8
   bytes depending on whether extended sequence numbers (ESN) is set
   for the SA. The padding is all zeros.
o The ciphertext
o Padding that rounds the total length up to an integral multiple of
   16 bytes. This padding is also all zeros.
o The length of the additional authenticated data (AAD) in octets
   (as a 64-bit little-endian integer).
o The length of the ciphertext in octets (as a 64-bit little-endian
   integer).

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. Use in IKEv2

AEAD algorithms can be used in IKE, as described in [RFC5282]. More
specifically:

o The Encrypted Payload is as described in section 3 of that
   document.
o The IV is 64 bits, as described in Section 2.
o The AAD is as described in section 5.1 of RFC 5282, so it's 32
   bytes (28 for the IKEv2 header + 4 bytes for the encrypted payload
   header) assuming no unencrypted payloads.

4. Negotiation in IKEv2

When negotiating the ChaCha20-Poly1305 algorithm for use in IKE or
IPsec, the value xxx (TBA by IANA) should be used in the transform
substructure of the SA payload as the ENCR (type 1) transform ID. As
with other AEAD algorithms, INTEG (type 3) transform substructures
MUST NOT be specified or just one INTEG transform MAY be included
with value NONE (0).

5. 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.

   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.

6. IANA Considerations

   IANA is requested to assign one value from the IKEv2 "Transform Type
   1 - Encryption Algorithm Transform IDs" registry, with name
   ENCR_CHACHA20_POLY1305, and this document as reference for both ESP
   and IKEv2.

7. 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. Thanks
   also to Martin Willi for pointing out the discrepancy with the final
   version of the algorithm document, and to Valery Smyslov and Tero
   Kivinen for helpful comments on this draft.

8. References

8.1. 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.

   [RFC5282]   Black, D. and D. McGrew, "Using Authenticated Encryption
               Algorithms with the Encrypted Payload of the Internet Key
               Exchange version 2 (IKEv2) Protocol", RFC 5282, August
               2008.

   [RFC7296]   Kivinen, T., Kaufman, C., Hoffman, P., Nir, Y., and P.
               Eronen, "Internet Key Exchange Protocol Version 2
               (IKEv2)", RFC 7296, October 2014.

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

8.2. Informative References

   [AE]   Bellare, M. and C. Namprempre, "Authenticated Encryption:
               Relations among notions and analysis of the generic
               composition paradigm", 2000,
               <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>.

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

   [SP800-67]
               National Institute of Standards and Technology,
               "Recommendation for the Triple Data Encryption Algorithm
               (TDEA) Block Cipher", FIPS SP800-67, January 2012,
               <http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/
               SP-800-67-Rev1.pdf>.

   [standby-cipher]
               McGrew, D., Grieco, A., and Y. Sheffer, "Selection of
               Future Cryptographic Standards", draft-mcgrew-standby-
               cipher (work in progress), January 2013.

Author's Address

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

Email: ynir.ietf@gmail.com