Network Working Group
Internet-Draft
Updates: 5246, 6347 (if approved)
Intended status: Standards Track
Expires: October 5, 2015

A. Langley
W. Chang
Google Inc
N. Mavrogiannopoulos
Red Hat
J. Strombergson
Secworks Sweden AB
S. Josefsson
SJD AB
April 3, 2015

The ChaCha Stream Cipher for Transport Layer Security
draft-mavrogiannopoulos-chacha-tls-05

Abstract

This document describes the use of the ChaCha stream cipher with Poly1305 in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) protocols.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes the use of the ChaCha stream cipher in the Transport Layer Security (TLS) version 1.2 [RFC5246] protocol, as well as in the Datagram Transport Layer Security (DTLS) version 1.2 [RFC6347], or any later versions.

ChaCha [CHACHA] is a stream cipher that has been designed for high performance in software implementations. The cipher has compact implementation and uses few resources and inexpensive operations that makes it suitable for implementation on a wide range of architectures. It has been designed to prevent leakage of information through side channel analysis, has a simple and fast key setup and provides good overall performance. It is a variant of Salsa20 [SALSA20SPEC] which is one of the selected ciphers in the eSTREAM portfolio [ESTREAM].

Recent attacks [CBC-ATTACK] have indicated problems with CBC-mode cipher suites in TLS and DTLS as well as issues with the only supported stream cipher (RC4) [RC4-ATTACK]. While the existing AEAD (AES-GCM) ciphersuites address some of these issues, concerns about the performance and ease of software implementation are sometimes raised.

Therefore, a new stream cipher to replace RC4 and address all the previous issues is needed. It is the purpose of this document to describe a secure stream cipher for both TLS and DTLS that is comparable to RC4 in speed on a wide range of platforms and can be

implemented easily without being vulnerable to software side-channel attacks.

2. The ChaCha Cipher

ChaCha [CHACHA] is a stream cipher developed by D. J. Bernstein in 2008. It is a refinement of Salsa20 and was used as the core of the SHA-3 finalist, BLAKE.

The variant of ChaCha used in this document is ChaCha with 20 rounds, a 96-bit nonce and a 256 bit key, which will be referred to as ChaCha20 in the rest of this document. This is the conservative variant (with respect to security) of the ChaCha family and is described in [I-D.irtf-cfrg-chacha20-poly1305].

3. The Poly1305 Authenticator

Poly1305 [POLY1305] is a Wegman-Carter, one-time authenticator designed by D. J. Bernstein. Poly1305 takes a 32-byte, one-time key and a message and produces a 16-byte tag that authenticates the message such that an attacker has a negligible chance of producing a valid tag for an inauthentic message. It is described in [I-D.irtf-cfrg-chacha20-poly1305].

4. ChaCha20 Cipher Suites

In the next sections different ciphersuites are defined that utilize the ChaCha20 cipher combined with various message authentication methods.

In all cases, the ChaCha20 cipher, as in [I-D.irtf-cfrg-chacha20-poly1305], uses a 96-bit nonce. That nonce is updated on the encryption of every TLS record, and is formed as follows.

```
struct {
    opaque salt[4];
    opaque record_counter[8];
} ChaChaNonce;
```

The salt is generated as part of the handshake process. It is either the client_write_IV (when the client is sending) or the server_write_IV (when the server is sending). The salt length (SecurityParameters.fixed_iv_length) is 4 bytes. The record_counter is the 64-bit TLS record sequence number. In case of DTLS the record_counter is formed as the concatenation of the 16-bit epoch with the 48-bit sequence number.

In both TLS and DTLS the ChaChaNonce is implicit and not sent as part of the packet.

The pseudorandom function (PRF) for TLS 1.2 is the TLS PRF with SHA-256 as the hash function.

The RSA, DHE_RSA, ECDHE_RSA, ECDHE_ECDSA, PSK, DHE_PSK, RSA_PSK, ECDHE_PSK key exchanges are performed as defined in [RFC5246], [RFC4492], and [RFC5489].

4.1. ChaCha20 Cipher Suites with Poly1305

The ChaCha20 and Poly1305 primitives are built into an AEAD algorithm [RFC5116], AEAD_CHACHA20_POLY1305, described in [I-D.irtf-cfrg-chacha20-poly1305]. It takes as input a 256-bit key and a 96-bit nonce.

When used in TLS, the "record_iv_length" is zero and the nonce is set to be the ChaChaNonce. The additional data is seq_num + TLSCompressed.type + TLSCompressed.version + TLSCompressed.length, where "+" denotes concatenation.

The following CipherSuites are defined.

```
TLS_RSA_WITH_CHACHA20_POLY1305         = {0xTBD, 0xTBD} {0xCC, 0xA0}
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305   = {0xTBD, 0xTBD} {0xCC, 0xA1}
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305 = {0xTBD, 0xTBD} {0xCC, 0xA2}

TLS_DHE_RSA_WITH_CHACHA20_POLY1305     = {0xTBD, 0xTBD} {0xCC, 0xA3}
TLS_DHE_PSK_WITH_CHACHA20_POLY1305     = {0xTBD, 0xTBD} {0xCC, 0xA4}

TLS_PSK_WITH_CHACHA20_POLY1305         = {0xTBD, 0xTBD} {0xCC, 0xA5}
TLS_ECDHE_PSK_WITH_CHACHA20_POLY1305   = {0xTBD, 0xTBD} {0xCC, 0xA6}
TLS_RSA_PSK_WITH_CHACHA20_POLY1305     = {0xTBD, 0xTBD} {0xCC, 0xA7}
```

5. Acknowledgements

The authors would like to thank Zooko Wilcox-OHearn and Samuel Neves.

6. IANA Considerations

IANA is requested to assign the following Cipher Suites in the TLS Cipher Suite Registry:

```
TLS_RSA_WITH_CHACHA20_POLY1305         = {0xTBD, 0xTBD} {0xCC, 0xA0}
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305   = {0xTBD, 0xTBD} {0xCC, 0xA1}
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305 = {0xTBD, 0xTBD} {0xCC, 0xA2}

TLS_DHE_RSA_WITH_CHACHA20_POLY1305     = {0xTBD, 0xTBD} {0xCC, 0xA3}
TLS_DHE_PSK_WITH_CHACHA20_POLY1305     = {0xTBD, 0xTBD} {0xCC, 0xA4}

TLS_PSK_WITH_CHACHA20_POLY1305         = {0xTBD, 0xTBD} {0xCC, 0xA5}
TLS_ECDHE_PSK_WITH_CHACHA20_POLY1305   = {0xTBD, 0xTBD} {0xCC, 0xA6}
TLS_RSA_PSK_WITH_CHACHA20_POLY1305     = {0xTBD, 0xTBD} {0xCC, 0xA7}
```

The ciphersuite numbers listed on the last column are numbers used for ciphersuite interoperability testing, and are the suggested to IANA to assign.

7. Security Considerations

ChaCha20 follows the same basic principle as Salsa20, a cipher with significant security review [SALSA20-SECURITY][ESTREAM]. At the time of writing this document, there are no known significant security problems with either cipher, and ChaCha20 is shown to be more resistant in certain attacks than Salsa20 [SALSA20-ATTACK]. Furthermore ChaCha20 was used as the core of the BLAKE hash function, a SHA3 finalist, that had received considerable cryptanalytic attention [NIST-SHA3].

Poly1305 is designed to ensure that forged messages are rejected with a probability of 1-(n/2^102) for a 16*n byte message, even after sending 2^64 legitimate messages.

The cipher suites described in this document require that a nonce is never repeated under the same key. The design presented ensures that by using the TLS sequence number which is unique and does not wrap [RFC5246].

This document should not introduce any other security considerations than those that directly follow from the use of the stream cipher ChaCha20, the AEAD_CHACHA20_POLY1305 construction, (see also the Security Considerations section of [I-D.irtf-cfrg-chacha20-poly1305]).

8. References

8.1. Normative References

[RFC4492] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, May 2006.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5489] Badra, M. and I. Hajjeh, "ECDHE_PSK Cipher Suites for Transport Layer Security (TLS)", RFC 5489, March 2009.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

[I-D.irtf-cfrg-chacha20-poly1305] Nir, Y. and A. Langley, "ChaCha20 and Poly1305 for IETF protocols", draft-irtf-cfrg-chacha20-poly1305-10 (work in progress), February 2015.

8.2. Informative References

[CHACHA] Bernstein, D., "ChaCha, a variant of Salsa20", January 2008, <http://cr.yp.to/chacha/chacha-20080128.pdf>.

[POLY1305] Bernstein, D., "The Poly1305-AES message-authentication code.", March 2005, <http://cr.yp.to/mac/poly1305-20050329.pdf>.

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", RFC 5116, January 2008.

[SALSA20SPEC] Bernstein, D., "Salsa20 specification", April 2005, <http://cr.yp.to/snuffle/spec.pdf>.

[SALSA20-SECURITY] Bernstein, D., "Salsa20 security", April 2005, <http://cr.yp.to/snuffle/security.pdf>.

[ESTREAM] Babbage, S., DeCanniere, C., Cantenaut, A., Cid, C., Gilbert, H., Johansson, T., Parker, M., Preneel, B., Rijmen, V., and M. Robshaw, "The eSTREAM Portfolio (rev. 1)", September 2008, <http://www.ecrypt.eu.org/stream/finallist.html>.

[CBC-ATTACK] AlFardan, N. and K. Paterson, "Lucky Thirteen: Breaking the TLS and DTLS Record Protocols", IEEE Symposium on Security and Privacy , 2013.

[RC4-ATTACK]
Isobe, T., Ohigashi, T., Watanabe, Y., and M. Morii, "Full Plaintext Recovery Attack on Broadcast RC4", International Workshop on Fast Software Encryption , 2013.

[SALSA20-ATTACK]
Aumasson, J-P., Fischer, S., Khazaei, S., Meier, W., and C. Rechberger, "New Features of Latin Dances: Analysis of Salsa, ChaCha, and Rumba", 2007, <http://eprint.iacr.org/2007/472.pdf>.

[NIST-SHA3]
Chang, S., Burr, W., Kelsey, J., Paul, S., and L. Bassham, "Third-Round Report of the SHA-3 Cryptographic Hash Algorithm Competition", 2012, <http://dx.doi.org/10.6028/NIST.IR.7896>.

Authors' Addresses

Adam Langley
Google Inc

Email: agl@google.com

Wan-Teh Chang
Google Inc

Email: wtc@google.com

Nikos Mavrogiannopoulos
Red Hat

Email: nmav@redhat.com

Joachim Strombergson
Secworks Sweden AB

Email: joachim@secworks.se
URI: http://secworks.se/

Simon Josefsson
SJD AB

Email: simon@josefsson.org
URI: http://josefsson.org/