Network Working Group V. Smyslov
Internet-Draft ELVIS-PLUS
Intended status: Informational September 7, 2018
Expires: March 11, 2019

# Clarifications and Implementation Guidelines for using TCP Encapsulation in IKEv2

draft-smyslov-ipsecme-tcp-guidelines-00

## Abstract

The Internet Key Exchange Protocol version 2 (IKEv2) defined in [RFC7296] uses UDP transport for its messages. [RFC8229] specifies a way to encapsulate IKEv2 and ESP (Encapsulating Security Payload) messages in TCP, thus making possible to use them in network environments that block UDP traffic. However, some nuances of using TCP in IKEv2 are not covered by that specification. This document provides clarifications and implementation guidelines for [RFC8229].

## Status of This Memo



## Copyright Notice

Table of Contents



## 1. Introduction

The Internet Key Exchange version 2 (IKEv2) as it is defined in [RFC7296] uses UDP as a transport protocol. As time passed the network environment has been evolved and sometimes this evolution has resulted in situations when UDP messages are dropped by network infrastructure. This may happen either by incapability of network devices to properly handle them (e.g. non-initial fragments of UDP messages) of by deliberate configuration of network devices that blocks UDP traffic.

Several standard solutions have been developed to deal with such situations. In particular, [RFC7383] defines a way to avoid IP fragmentation of large IKE messages and [RFC8229] specifies a way to transfer IKEv2 and ESP (Encapsulated Security Payload) messages over a stream protocol like TCP. This document focuses on the latter specification and its goal is to give implementers guidelines how to properly use reliable connection-oriented stream transport in IKEv2.

Since originally IKEv2 relied on unreliable transport, it was designed to deal with this unreliability. IKEv2 has its own retransmission timers, replay detection logic etc. Using reliable transport makes many of such things unnecessary. On the other hand, connection-oriented transport require IKEv2 to keep the connection alive and to restore it in case it is broken, the tasks that were not needed before. [RFC8229] gives recommendations how peers must behave in different situations to keep the connection. However, implementation experience has revealed that not all situations are covered in [RFC8229], that may lead to interoperability problems or

to suboptimal performance. This memo gives implementers more guidelines how to use reliable stream tranport in IKEv2 in situations, which are not covered in [RFC8229].

## 2. Terminology and Notation

This document shares the terinology with [RFC8229]. In particular, it uses terms "TCP Originator" and "TCP Responder" to refer to the parties that initiate or responded to the TCP connection created for the initial IKE SA (in a possible series of successive rekeys). More details are given in Section 1.2 of [RFC8229].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 3. Retransmissions

Section 2.1 of [RFC7296] describes how IKEv2 deals with unreliability of UDP protocol. In brief, exchange initiator is responsible for retransmissions and must retransmit requests message until response message is received. If no reply is received after several retransmissions, the SA is deleted. The responder never retransmits but must resend the response message in case it receives retransmitted request.

When IKEv2 uses reliable transport protocol, most of these rules become unnecessary. Since [RFC8229] doesn't provide clear guidance on using retransmissions in case of TCP encapsulation, this memo gives the following rules.

- the exchange initiator SHOULD NOT retransmit request message; if no response is received within some reasonable period of time, the IKE SA is deleted

- if TCP connection is broken and then restored while the exchange initiator is waiting for the response, the initiator MUST retrasmit the request and continue to fait for the response

- the exchange responder acts as described in Section 2.1 of [RFC7296], i.e. using TCP encapsulation doesn't change the responder's behavior

## 4. Using Cookies and Puzzles

IKEv2 provides a DoS attack protection mechanism called Cookie, which is described in Section 2.6 of [RFC7296]. [RFC8019] extends this mechanism for protection against DDoS attacks by means of Client Puzzles. Both mechanisms allow the responder to keep no state until the initiator proves its IP address is real (and solves puzzle in the latter case).

[RFC8229] gives no guidance on how these mechanisms should be used in case of TCP encapsulation. However, the connection-oriented nature of TCP brings additional considerations for using these mechanisms. In general, Cookie provides less value in case of TCP encapsulation, because when the responder receives the IKE_SA_INIT request the TCP session has already been established, so the initiator's IP address has been verified. Moreover, TCP Responder creates state as far as the SYN packer is received (unless SYN Cookies described in [RFC4987] are employed), that distorts the stateless nature of IKEv2 Cookies. So, it makes little sense to send Cookie request in this situation, unless the responder in concerned with the possibility of TCP Sequence Number attacks (see [RFC6528] for details). On the other hand, Puzzles still remain useful and their use requires using Cookies.

The following considerations are applicable for using Cookie and Puzzle mechanisms in case of TCP encapsulation.

- o the exchange responder SHOULD NOT request Cookie unless the responder has good reason to do it (like a concern of the possibility of TCP Sequence Number attacks or Puzzle request is sent in the same message)

- o if the responder chooses to send Cookie request (possibly along with Puzzle request), then the TCP connection that the IKE_SA_INIT request message was received over SHOULD be closed, so that the responder remains stateless at least until the Cookie (or Puzzle Solution) is returned

    - * note, that if this TCP connection is closed, then the responder MUST NOT include the initiator's TCP port into the Cookie calculation (*), since the Cookie will be returned over a new TCP connection with a different port

- o the exchange initiator acts as described in Section 2.6 of [RFC7296] and Section 7 of [RFC8019], i.e. using TCP encapsulation doesn't change the initiator's behavior

(*) Examples of Cookie calculation methods are given in Section 2.6 of [RFC7296] and in Section 7.1.1.3 of [RFC8019] and they don't include transport protocol ports. However these examples are given for illustrative purposes, since Cookie generation algorithm is a local matter and some implementations might include port numbers, that won't work with TCP encapsulation.

## 5. Error Handling in the IKE_SA_INIT

Section 2.21.1 of [RFC7296] describes how error notifications should be handled in the IKE_SA_INIT exchange. In particular, it is advised that the initiator should not act immediately after receiving error notification and should instead wait some time for valid response, since the IKE_SA_INIT messages are completely unauthenticated. This advise has little sense in case of TCP encapsulation. If the initiator receives the response message over TCP, then either this message is genuine and was sent by the peer, or the TCP session was hijacked and the message is forged, but in this case no genuine messages from the responder will be received.

So, in case of TCP encapsulation the initiator SHOULD NOT wait for additional messages in case it receives error notification from the responder in the IKE_SA_INIT exchange.

## 6. Interaction with MOBIKE Protocol

[RFC4555] defines MOBIKE protocol, that allows IKEv2 SA to migrate between IP addresses. Section 8 of [RFC8229] describes how interaction between MOBIKE and TCP encapsulation. This memo provides clarifications and additional recommendations for using MOBIKE in case of TCP encapsulation.

[RFC8229] recommends, that in case of IP address change, the initiator first try UDP initiate the INFORMATIONAL exchange containing UPDATE_SA_ADDRESSES notification using UDP transport and if no response is received then send this notification over TCP transport. This recommendation lacks some details.

- o when switching from UDP to TCP the Message ID of the exchange MUST NOT be changed

- o on the other hand, when switching from UDP to TCP the content of the NAT_DETECTION_SOURCE_IP notification included in the request MUST be recalculated (because TCP source port will most probably be different from UDP source port)

Section 3.7 of [RFC4555] describes an additional optional step in the process of changing IP addresses called Return Routability Check. It

is performed by the responder in order to be sure that the new initiator's address is in fact routable. In case of TCP encapsulation this check has little value, since TCP handshake proves rotability of the TCP Originator's address. So, in case of TCP encapsulation the Return Routability Check SHOULD NOT be performed.

## 7. Using TCP Encapsulation with High Availability Cluster

[RFC6311] defines a support for High Availability in IKEv2. The core idea is that in case of cluster failover a new active node immediately initiates the special INFORMATION exchange containing the IKEV2_MESSAGE_ID_SYNC motification, which instructs the client to skip some number of Message IDs that might not be synchronized yet between nodes at the time of failover.

The problem is that TCP states are much harder to synchronize than IKE states - it requires access to TCP/IP stack internals, which is not always avaivable for IKE/IPsec implementations. If a cluster implementation doesn't synchronize TCP states between nodes, then after failover event the new active node will not have any TCP connection with the client, so the node cannot initiate the INFORMATIONAL exchange as required by [RFC6311]. Since the cluster usually acts as TCP Responder, the new active node cannot re-establish TCP connection, since only the TCP Originator can do it. And for the client the situation of cluster failover may remain unknown for long time if it has no IKE or ESP traffic to send. Once the client sends any ESP or IKEv2 packet, the cluster node will reply with TCP RST and the client (as TCP Originator) will restore the TCP connection so that the node will be able to initiate the INFORMATIONAL exchange informing the client about the cluster failover.

This memo makes the following recommendation: if support for High Availability in IKEv2 is negotiated and TCP transport is used and a client is TCP Originator, then the client SHOULD periodically send IKEv2 messages (e.g. by initiating liveness check exchange) whenever there is no any IKEv2 or ESP traffic. This differs from the recommendations given in Section 2.4 of [RFC7296] in the following: the liveness check should be periodically performed even if the client has nothing to send over ESP. The frequency of sending such messages should be high enough to allow quick detection and restoring of broken TCP connection.

## 8. Security Considerations

Security considerations concerning using TCP encapsulation in IKEv2 and ESP are given in [RFC6311]. This memo doesn't provide additional security considerations.

# 9. References

## 9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, DOI 10.17487/RFC4555, June 2006, <https://www.rfc-editor.org/info/rfc4555>.

[RFC6311] Singh, R., Ed., Kalyani, G., Nir, Y., Sheffer, Y., and D. Zhang, "Protocol Support for High Availability of IKEv2/IPsec", RFC 6311, DOI 10.17487/RFC6311, July 2011, <https://www.rfc-editor.org/info/rfc6311>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/info/rfc7296>.

[RFC8019] Nir, Y. and V. Smyslov, "Protecting Internet Key Exchange Protocol Version 2 (IKEv2) Implementations from Distributed Denial-of-Service Attacks", RFC 8019, DOI 10.17487/RFC8019, November 2016, <https://www.rfc-editor.org/info/rfc8019>.

[RFC8229] Pauly, T., Touati, S., and R. Mantha, "TCP Encapsulation of IKE and IPsec Packets", RFC 8229, DOI 10.17487/RFC8229, August 2017, <https://www.rfc-editor.org/info/rfc8229>.

## 9.2. Informative References

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", RFC 4987, DOI 10.17487/RFC4987, August 2007, <https://www.rfc-editor.org/info/rfc4987>.

[RFC6528] Gont, F. and S. Bellovin, "Defending against Sequence Number Attacks", RFC 6528, DOI 10.17487/RFC6528, February 2012, <https://www.rfc-editor.org/info/rfc6528>.

[RFC7383] Smyslov, V., "Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation", RFC 7383, DOI 10.17487/RFC7383, November 2014, <https://www.rfc-editor.org/info/rfc7383>.

## Author's Address

Valery Smyslov
ELVIS-PLUS
PO Box 81
Moscow (Zelenograd)  124460
RU

Phone: +7 495 276 0211
Email: svan@elvis.ru