Working Group
Internet-Draft
Intended status: Informational
Expires: August 9, 2014

U. Chunduri
A. Tian
Ericsson Inc.
J. Touch
USC/ISI

February 5, 2014

A framework for RPs to use IKEv2 KMP
draft-chunduri-karp-using-ikev2-with-tcp-ao-06

Abstract

This document describes a mechanism to enable using IKEv2 with TCP-AO, which may also be of more general use to other pairwise Routing Protocols.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 9, 2014.

Table of Contents

1. Introduction

This document analyzes the pairwise Routing Protocol (RP) requirements needed to integrate the IKEv2[RFC5996] KMP and provides a framework to achieve this.

The KARP design guide [RFC6518] suggests various requirements and options for obtaining keys to protect the routing protocols and recommends using a Key Management Protocol (KMP) to automate key establishment, as well as rekeying to continuously protect the routing protocols. However, there are few gaps which need to be addressed for serene integration of IKEv2 KMP and any pairwise routing protocol either securing messages by RP itself or through a security protocol like TCP-AO [RFC5925]. For example, there are differences in both established protocols like IKEv2 and TCP-AO on how the Security Associations (SAs) to be maintained or there is a need for common framework in general on how the pairwise RPs can further offload SA management. This memo addresses these gaps by providing a common framework to interact pairwise RPs and IKEv2 KMP. The choice of IKEv2 KMP is based on the WG consensus.

A major portion of pairwise RPs analyzed in this document use TCP at transport layer and may use TCP-AO[RFC5925] to protect the RP

messages. There are other RPs, which use pairwise unicast signaling between the routing peers (for e.g., BFD [RFC5880]) and don't use TCP at transport layer. This memo also describes the interface for these RPs to integrate with IKEv2 KMP.

This document introduces a new Gatekeeper (GK) module, which provides a common interface and minimizes the changes for all pairwise routing protocols to be integrated with KMP. The Gatekeeper module does the SA management and interaction with KMP as well as TCP-AO protocol or the RP itself (for the RPs which don't use TCP-AO). The purpose of the Gatekeeper is to act as a shim between IKEv2 and RP/TCP-AO, so that RP/TCP-AO and the Gatekeeper together act like IPsec to IKEv2 (since IKEv2 is designed to tightly interact with IPsec). This document defines this common interface between pairwise RPs with Gatekeeper and IKEv2 [RFC5996]. The common interface defined here also serves the pairwise RPs with manual keying and this is further described in Section 2.1.

Currently IKEv2 can establish only Security Association (SA) for IPsec. A few extensions are needed for IKEv2 to establish SA for pairwise RPs which either protect protocol packets by themselves or use TCP-AO for protection. [mahesh-karp-rkmp] discusses the summary of extensions required for IKEv2 protocol for key establishment, traffic selectors negotiation and SA establishment to support the keying and parameters needed by RP or TCP-AO.

1.1. Requirements Language

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 RFC 2119 [RFC2119].

1.2. Acronyms

BGP - Border Gateway Protocol

GKR - Gatekeeper Record

IKEv2 - Internet Key Exchange Protocol Version 2

IPsec - Security Architecture for the Internet Protocol

KDF - Key Derivation Function as defined in TCP-AO

KMP - Key Management Protocol (auto key management)

LDP - Label Distribution Protocol

MKM - Manual Key management Protocols

MKT - Master Key Tuples as defined in TCP-AO

MSDP - Multicast Source Discovery Protocol

PAD - Peer Authorization Database

PCEP - Path Computation Element Communication Protocol

RP - Routing Protocol

SA - Security Association

TCP-AO - TCP Authentication Option

2. Motivation and Overview

The motivation of this document is to offload Security Association (SA) management and to provide a generic and common interface for all pairwise RPs to integrate with KMPs in general and specifically with IKEv2 KMP.

IKEv2 assumes IPsec triggers new SA requests, manages SA timers and rekeys SAs as needed to protect the actual traffic. For e.g., for TCP-based RPs, TCP-AO assumes an external key manager, which could support functions like Master key triggering, SA timers, and rekey triggering to get the parameters required including Master key to protect the TCP session. To bridge the gap between IKEv2 and TCP-AO or to simplify pairwise RPs which don't use TCP-AO, this document defines a Gatekeeper module as described in Section 3.

The following diagram depicts how, the Gatekeeper module interfaces with all protocols involved i.e., Pairwise RPs which do security by themselves, TCP-based RPs which use TCP-AO for providing security, IKEv2 KMP, and TCP-AO itself. This also shows the interaction with various databases viz., Peer Authorization Database (PAD) and Crypto Key Tables with the Gatekeeper.

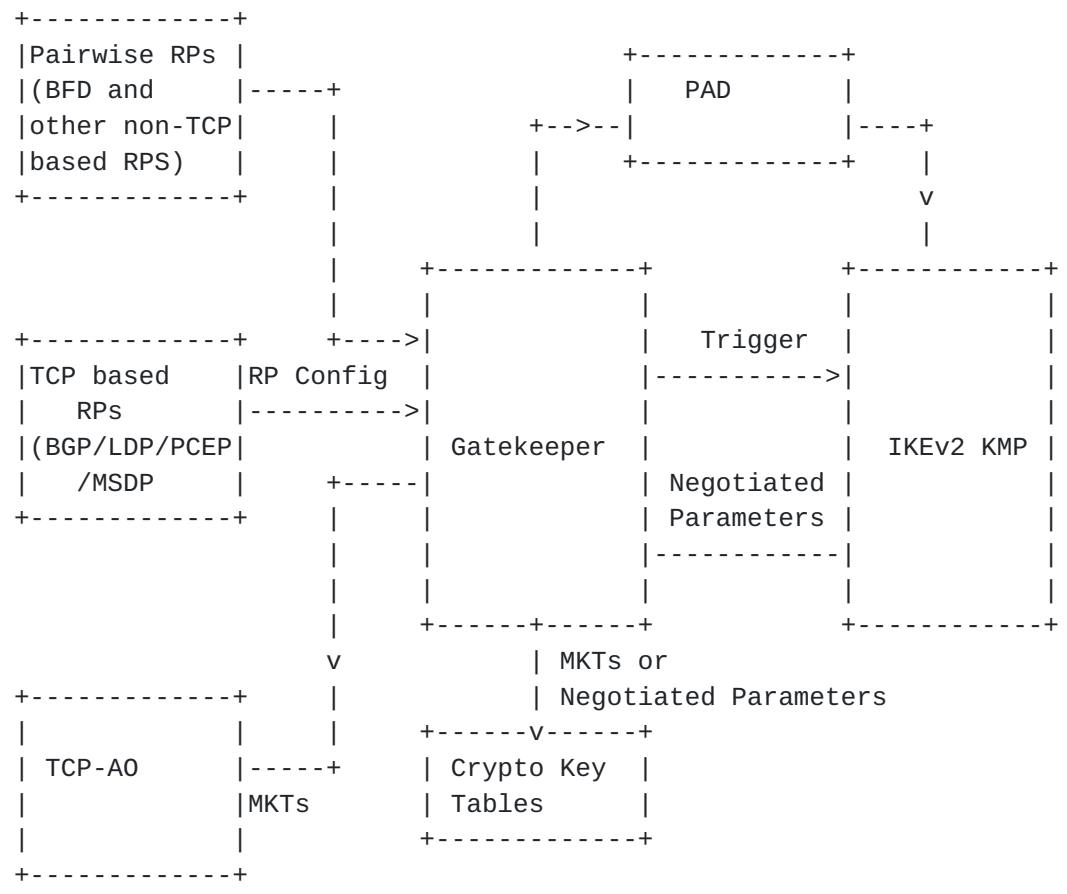

Figure 1: KARP KMP: Using IKEv2 with Pairwise RPs

In Figure 1, before initiating the RP messaging to the peer, non-TCP-based RPs communicate the provisioned configuration to Gatekeeper module. Similarly, before initiating the TCP connection, all TCP-based RPs communicate the provisioned configuration to Gatekeeper module. A entry in the KMP peer authentication/authorization is provisioned in PAD as defined in Section 4.4.3 of [RFC4301] and pointer to this entry SHOULD be part of the RP configuration. This facilitates Gatekeeper to issue a corresponding request, with all the proposed alternatives at the RP to the IKEv2 KMP. This enables the IKEv2 to negotiate the needed security policy parameters and derive Keying material to be used by RPs. When the local peer is acting as a responder, security policy information populated at the Gatekeeper can be referenced through PAD by IKEv2 KMP to create the CHILD_SAs ([RFC5996]). Either way, the negotiated SA's are kept in the crypto key table database as specified in [ietf-karp-crypto-key-table] and this information is the basis for provisioning MKTs in case of TCP-AO or applying security by BFD [RFC5880] and other non-TCP based RPs themselves.

The Gatekeeper can be viewed as a module, which maintains the KMP negotiated SAs as per the provisioning information at RPs and initiates rekey triggers as needed. For TCP-AO, the rekey triggers helps provision new MKTs for the long-lived TCP sessions protected by TCP-AO. The Gatekeeper also installs these new keys in TCP-AO consistent with TCP-AO's support for key changes. For non-TCP-based RPs as shown in the above diagram, the Gatekeeper populates the new keys in crypto key tables to be referenced for securing the protocol messages.

Section 3 describes in detail the role of Gatekeeper and it's interfaces to all the protocols and the databases it interacts with. Section 3.3.2, Section 3.3.1 describes the static databases used and the interaction with the Gatekeeper in detail.

2.1. Manual Keying with the Gatekeeper

Though the Gatekeeper defined offloads the SA management KMP databases interaction, the framework defined in this memo is consistent and can also be used purely for manual keying at pairwise RPs. The following diagram depicts the Gatekeeper module interfaces with all protocols involved i.e., Pairwise RPs which do security by themselves, TCP-based RPs which use TCP-AO, TCP-AO itself and the Crypto Key Tables database.

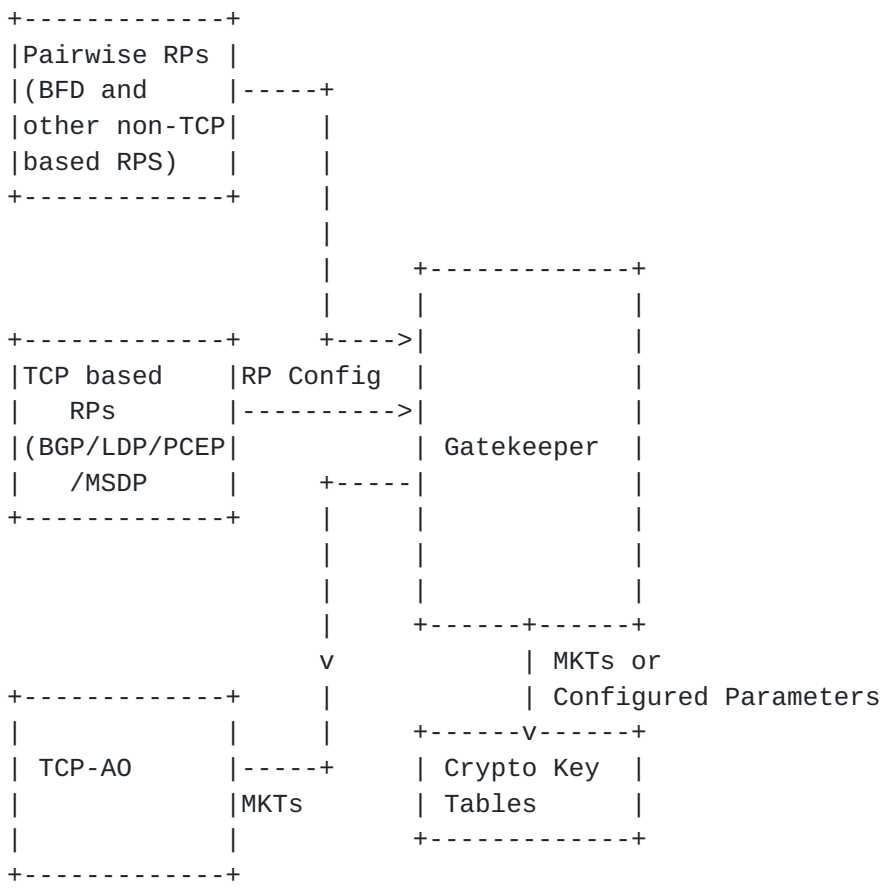

Figure 2: KARP: Using Manual Keying with Pairwise RPs

As represented in Figure 2 above; here the Gatekeeper creates the static entries as per provisioned credentials including the Keys to protect RP messages either in the crypto key table database as specified in [ietf-karp-crypto-key-table]; or provisioning MKTs in the TCP-AO for TCP-based RPs.

3. The Gatekeeper

The Gatekeeper primarily enables IKEv2 to support key and parameter negotiation, which are eventually used either by TCP-AO or by other pairwise RPs directly to protect the protocol messages. TCP-AO has a different model of security associations and key management than IPsec. IKEv2 is designed to support IPsec's model.

The Gatekeeper maintains a Gatekeeper record (GKR) to keep track of either TCP-AO MKTs or negotiated parameters used by other pairwise RPs. For long-lived TCP connections MKTs can be rolled over by rekeying, hence creating new MKTs and installing them in TCP-AO. The GKR for TCP-based RPs, can be viewed as a superset of MKT i.e., it

maintains and tracks the lifetime of the provisioned MKT, and includes other per-connection parameters needed by TCP-AO, such as algorithm, key length, etc. [RFC5926]. It also maintains the reference to PAD and Crypto Key Table entries to facilitate RP security parameters negotiation with IKEv2 KMP.

The following sections define the Gatekeeper module interface between TCP-based RPs, TCP-AO, other pairwise RPs seeking to use IKEv2 KMP, interface to IKEv2 KMP itself and other key databases.

3.1. TCP-based RP interface to the Gatekeeper

When a TCP-based routing protocol is configured to use TCP-AO with KMP (by not specifying the keys or through some other means), TCP connection identifiers, all configured Message Authentication Code (MAC) algorithms, all configured Key Derivation Function (KDF) parameters, rekey lifetime and the TCP option flag (i.e., all additional parameters specified in [RFC5926]) are populated in the Gatekeeper record. This information includes the reference to PAD, which has all the information to authorize and authenticate IKEv2 peer. Having this information at a central place is essential and enables the node to respond to the requests received from other IKEv2 peers in the network. In the case of manual keying, as there is no policy negotiation with the peer, the Gatekeeper record is populated with all the provisioned information at RP including the master keys.

If the same routing protocol needs to differentiate transport sessions by securing separate TCP connections between the same endpoints then the TCP connection identifiers need to be provisioned appropriately in the Gatekeeper. The TCP connection identifiers could be either full socket pair i.e., local IP address, remote IP address, local TCP port, and remote TCP port or partial socket pair, indicated with wildcards as required. GKRs SHOULD thus support full or partial socket pair specification and this forms the basis for traffic selector negotiation with IKEv2 KMP [RFC5926].

In general, a full socket pair is not needed for negotiating the TCP-AO MKT with KMP. As specified in Section 3.1 of TCP-AO [RFC5925], socket pair values can be partially specified using ranges, masks, wildcards, or any other suitable indication. These provisioned socket pair parameters are supplied to KMP as context in which to negotiate traffic selectors for which the MKT or Master key should be used in TCP-AO.

For more details on cases where a full socket pair is needed before opening the connection, please refer Section 7.1. Provisioning of the Gatekeeper record SHOULD be done before opening the TCP connection. From the RP interface, the record created in Gatekeeper

contains only the RP's connection information, and this information is given to KMP (IKEv2) to obtain the negotiated parameters to protect the underlying TCP session by [RFC5925].

3.1.1. TCP-AO interface to Gatekeeper

TCP-AO expects an external entity to provision its MKTs in order to protect TCP sessions. The Gatekeeper module provides this function so that all TCP-based RPs can benefit from this common interface.

The following are the details of the interface between TCP-AO and the GK:

1. After getting the negotiated parameters and mutually authenticated Master key from the KMP, the Gatekeeper inserts a corresponding MKT and parameters into TCP-AO. The session-specific parameters include negotiated Connection identifiers, MAC algorithms, KDFs, KeyIDs, the TCP option flag and the Master Key given by the KMP.

2. MKT IDs (as specified in Section 3.1 of TCP-AO [RFC5925]) require a SendID and a RecvID for each MKT, which are mutually agreed by the connection endpoints. These 1-byte quantities need to be part of the MKT when the KMP key(s) are populated in MKT.

3. For long-lived TCP sessions, the Gatekeeper removes the old MKTs from TCP-AO after rekeying the corresponding new MKTs, to continuously protect the underlying TCP sessions.

4. In general, restarted TCP sessions can use existing MKT in TCP-AO i.e., IKEv2 need not be retriggered, since new key and parameter negotiation is not needed due to the protection already provided by TCP-AO (refer Section 5.3.1 of TCP-AO [RFC5925]). However, if GKR and hence TCP-AO MKT is created with full socket pair (in other words without using ranges, masks, wildcards for socket pair values, for the cases as specified in Section 7.1), then IKEv2 needs to be retriggered to get the new master key for the corresponding restarted TCP session.

3.2. Other pairwise RPs interface to the Gatekeeper

When a non-TCP-based RP is configured to use the KMP, before initiating connection with peer; connection identifiers, all configured Message Authentication Code (MAC) algorithms, all configured Key Derivation Function (KDF) parameters, rekey lifetime and reference to the PAD are populated in the Gatekeeper record. The RP connection identifiers at the Gatekeeper could be either full socket pair i.e., local IP address, remote IP address, local, remote

transport ports and protocol or partial socket pair, indicated with wildcards as required.

For non-TCP-based RPs all negotiated parameters from KMP are populated in Crypto Key table database [ietf-karp-crypto-key-table]. The entries in this database as specified in [ietf-karp-crypto-key-table] SHOULD directly be used by non-TCP-based RPs for securing the protocol messages.

3.3. KMP interaction with the Gatekeeper

As an initiator, IKEv2 expects an external trigger that contains the information required to negotiate security associations. There needs to be a way to trigger the KMP to initiate negotiation with all the provisioned parameters of a Gatekeeper record by any pairwise RP. A similar trigger is also required to rekey, to maintain the negotiated SAs for long-lived connections. As a responder to the peer IKEv2 requests and CHILD_SA creation; Gatekeeper record is consulted through the reference in PAD as described in Section 3.3.2 .

The purpose of this section is to define a common interface between the Gatekeeper and the IKEv2 KMP and also to list all the negotiated parameters to form an entry in the Crypto Key Tables as described in Section 3.3.1 .

The following are the details:

1. At the time of a new connection, a trigger to the KMP occurs to negotiate the session-specific parameters with the needed information on MAC algorithm, Traffic Selectors, and additionally for the TCP-based RPs KDF parameter, the TCP option flag from the Gatekeeper record are given as input parameters. The Gatekeeper at the peer is expected to have similar provisioning in place for responding to the received KMP request.

2. A KMP session identifier, provided by a successful key negotiation by the KMP, needs to be stored and should be used when the Gatekeeper make decision based on the lifetime to rekey the existing session.

3. For TCP-based RPs, MKT IDs (as specified in Section 3.1 of TCP-AO [RFC5925]) require a SendID and a RecvID for each MKT, mutually agreed by the connection endpoints. These 1-byte quantities need to be negotiated by the KMP with the peer to populate in the MKT. These fields are populated as "LocalKeyName" and "PeerKeyName" in the Crypto Key Table entry.

4. Crypto Key Table "Peers" field SHOULD be populated with the peer IP address.

5. For TCP-based RPs, KMP-negotiated KDF parameters for each session used to generate traffic keys from master keys to be populated in MKT. The same is referred as "KDF" in a corresponding Crypto Key Table entry.

6. A KMP-negotiated MAC algorithm, MKT connection identifiers (negotiated traffic selectors) and optionally life time for traffic keys for each session, need to be populated in MKT. The same is referred as "AlgID" in corresponding Crypto Key Table entry.

7. The "Key" field defined in Crypto Key Table contains a long-lived symmetric cryptographic key or Master Key in the format of a lower-case hexadecimal string. The size of the Key depends on the KDF and the AlgID.

8. IKEv2 does not negotiate rekey lifetime and rekeying is based on local operator policy. The Gatekeeper MUST add this capability for tracking the key lifetime provisioned at RPs and explicitly triggering the KMP to rekey when indicated. This rekey trigger then creates a new MKT for the underlying TCP connection. Implementations can proactively negotiate a new MKT Master Key before the lifetime of the current Master key expires.

The two essential databases being interacted by the Gatekeeper are explained below.

3.3.1. Interaction with KARP Crypto Key Table

KMP negotiated parameters are kept in the crypto key table database as specified in [ietf-karp-crypto-key-table]. In case of Manual keying, all the provisioned information including master key at RP is populated in the crypto key table database through the Gatekeeper to keep a common interface. The database is characterized as a table, where each row represents a single long-lived symmetric cryptographic key or Master key. The Gatekeeper record SHOULD have a reference to the Crypto Key Table Entry. One of the reasons to separate the negotiated parameters in a different table is to alleviate the population manually or through an external source. Non-TCP-based RPs can eventually use crypto key table entries to secure the protocol messages as specified in [ietf-karp-crypto-key-table].

3.3.2. Interface to the PAD

The Peer Authorization Database (PAD) for IPsec is described in Section 4.4.3 of [RFC4301]. This section describes the embodiments of the same in the context of RP security associations and security policies provisioned at the routing protocols. This is still the link between policies provisioned at the routing protocol and the SAs created by IKEv2 KMP. Instead of the Security Policy Database (SPD), Gatekeeper record holds the data for traffic selectors for child SA creation.

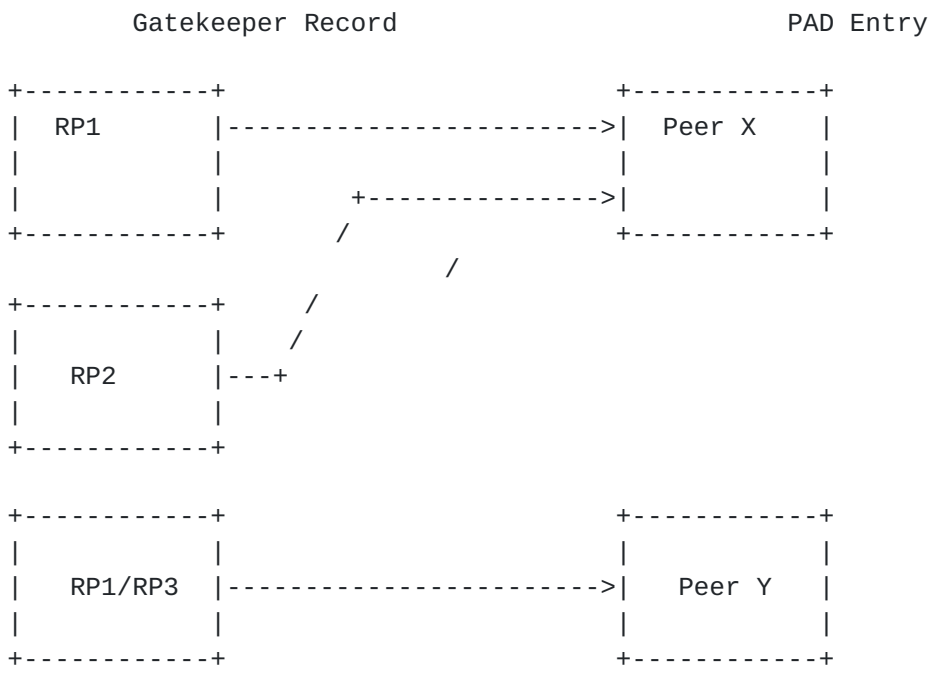

Figure 3: KARP KMP: Gatekeeper interface to the PAD

As shown in Figure 3, multiple RPs can point to the same peer and in this case, a PAD entry holds the reference to both the corresponding Gatekeeper records. The PAD entry for the IKEv2 peer is used to constrain the creation of child SAs; specifically, the PAD entry specifies how the Gatekeeper record is searched using a traffic selector proposal from a peer. For CHILD_SA creation, peer IP addresses asserted in traffic selector payloads SHOULD be used for Gatekeeper record lookups based on the remote IP address field portion of a Gatekeeper Record entry.

3.4. Impact of Policy changes

Once the routing session is secured either by TCP-AO or non-TCP-based RP itself, any security policy changes initiated by the operator at RP MUST cause a tear down of the existing session and MUST be replaced with a new CHILD_SA at IKEV2 KMP and corresponding new MKT at TCP-AO. Similarly, any changes in the peer Authentication data at PAD MUST cause re-authentication of the peer at IKEv2 KMP with changed credentials and also due to this change, all CHILD_SAs/MKTs need to re-negotiated.

4. IANA Considerations

This document defines no new namespaces.

5. Security Considerations

This document does not introduce any new security threats for IKEv2 [RFC5996] or TCP-AO [RFC5925]. For more detailed security considerations please refer the Security Considerations section of the KARP Design Guide [RFC6518] document as well as KARP threat document [I-D.ietf-karp-threats-reqs].

6. Acknowledgements

The authors would like to thank Joel Halpern for his initial discussions and providing feedback on the document. The authors also thank Tero Kivinin and Dan Harkins for reviewing the document and Ron Bonica for his initial requirement discussions. Thanks to Sam Hartman for his KARP working group discussions on this topic. The Gatekeeper module is originally proposed by Joe Touch.

7. Appendix A

7.1. BGP Multi Session and transport level differentiation

[ietf-idr-bgp-multisession] describes MP-BGP, which uses multiple TCP sessions between a pair of BGP speakers. Each TCP session is used to exchange routes related by some session-based attribute, such as AFI/SAFI. The reason transport level distinction is required could be because of operator policy. Though it is less likely to see different MAC/KDF parameters for each of these sessions, it is possible rekey lifetimes or TCP option flags for TCP-AO can be different for each of these AFI/SAFI based sessions.

If transport level separation is required for all sessions between a pair of BGP speakers, a unique and full socket pair (i.e., a local IP address, a remote IP address, a local TCP port, and a remote TCP

port) MUST be known before establishing a TCP connection. The full socket pair is required for both unique MKT creation in TCP-AO, as well as for the KMP to negotiate unique Master keys for each connection.

The use of different IP addresses to differentiate connections in multi session BGP is discouraged in [ietf-idr-bgp-multisession] and the destination port is always BGP. As a result, the only option for transport level differentiation is by knowing the source port of the connection being initiated. This is required to negotiate unique KMP SAs by the Gatekeeper, as well as to configure unique TCP-AO MKTs for each TCP connection. How source port lock-down is done is beyond the scope of this document (this is an implementation issue) and this can be achieved in many different ways before making the TCP connection.

The Gatekeeper interface, defined in Section 3, is oblivious to this issue and can well accommodate this requirement.

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.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, June 2010.

[RFC5926] Lebovitz, G. and E. Rescorla, "Cryptographic Algorithms for the TCP Authentication Option (TCP-AO)", RFC 5926, June 2010.

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

[RFC5998] Eronen, P., Tschofenig, H., and Y. Sheffer, "An Extension for EAP-Only Authentication in IKEv2", RFC 5998, September 2010.

8.2. Informative References

[I-D.ietf-idr-bgp-multisession] Scudder, J., Appanna, C., and I. Varlashkin, "Multisession BGP", draft-ietf-idr-bgp-multisession-07 (work in progress), September 2012.

[I-D.ietf-karp-crypto-key-table]
Housley, R., Polk, T., Hartman, S., and D. Zhang, "Database of Long-Lived Symmetric Cryptographic Keys", draft-ietf-karp-crypto-key-table-10 (work in progress), December 2013.

[I-D.ietf-karp-threats-reqs]
Lebovitz, G., Bhatia, M., and B. Weis, "Keying and Authentication for Routing Protocols (KARP) Overview, Threats, and Requirements", draft-ietf-karp-threats-reqs-07 (work in progress), December 2012.

[I-D.mahesh-karp-rkmp]
Jethanandani, M., Weis, B., Patel, K., Zhang, D., Hartman, S., Chunduri, U., Tian, A., and J. Touch, "Negotiation for Keying Pairwise Routing Protocols in IKEv2", draft-mahesh-karp-rkmp-05 (work in progress), November 2013.

[RFC3618] Fenner, B. and D. Meyer, "Multicast Source Discovery Protocol (MSDP)", RFC 3618, October 2003.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, June 2005.

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4746] Clancy, T. and W. Arbaugh, "Extensible Authentication Protocol (EAP) Password Authenticated Exchange", RFC 4746, November 2006.

[RFC4754] Fu, D. and J. Solinas, "IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 4754, January 2007.

[RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007.

[RFC5440] Vasseur, JP. and JL. Le Roux, "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, March 2009.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, June 2010.

[RFC5931] Harkins, D. and G. Zorn, "Extensible Authentication Protocol (EAP) Authentication Using Only a Password", RFC 5931, August 2010.

[RFC6124] Sheffer, Y., Zorn, G., Tschofenig, H., and S. Fluhrer, "An EAP Authentication Method Based on the Encrypted Key Exchange (EKE) Protocol", RFC 6124, February 2011.

[RFC6518] Lebovitz, G. and M. Bhatia, "Keying and Authentication for Routing Protocols (KARP) Design Guidelines", RFC 6518, February 2012.

Authors' Addresses

Uma Chunduri
Ericsson Inc.
300 Holger Way
San Jose, California 95134
USA

Phone: +1 (408) 750-5678
Email: uma.chunduri@ericsson.com

Albert Tian
Ericsson Inc.
300 Holger Way
San Jose, California 95134
USA

Phone: +1 (408) 750-5210
Email: albert.tian@ericsson.com

Joe Touch
USC/ISI
4676 Admiralty Way,
Marina del Rey, California 90292-6695
USA

Phone: +1 (310) 448-9151
Email: touch@isi.edu