A SECURITY FRAMEWORK FOR ROUTING PROTOCOLS

Nitin Prajapati

A thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science

Concordia University

Montréal, Québec, Canada

October 2014

Abstract

A Security Framework for Routing Protocols

Nitin Prajapati

With the rise in internet traffic surveillance and monitoring activities, the routing infrastructure has become an obvious target of attack as compromised routers can be used to stage large scale attacks. Routing protocols are also subjected to various threats such as capture and replay of packets that disclose the network information, forged routing control messages that may compromise a connection by deception, disruption of an on-going connection causing DoS attacks and spreading of unauthentic routing information in the network. Presently, strong cryptographic suites and key management mechanisms (IPsec and IKE) are available to secure host-to-host data communication but none of them focus on securing routing protocols. Today's routing protocols use a shared secret to perform mutual authentication and authorization, and depend on manual keying methods. For message integrity, they either rely on some built-in or external security feature that uses the same shared secret.

The KARP working group of the IETF identified that the work is required to tighten the security of the routing protocols and demonstrated that automated key management solutions are needed for increasing security. Towards this goal we propose the RPsec framework. RPsec provides a common baseline for development of KMPs for the routing protocols, supports both automated and manual key management, and overcomes the weakness of existing manual key methods.

Acknowledgments

I would like to thank the Almighty God for his divine guidance and continued blessings, “Om Namah Shivaya”.

I would like to gratefully and sincerely thank my advisor, Dr. J. William Atwood, for his guidance, support and encouragement throughout my master study at Concordia University. I learned a lot from him about life, research, thinking out of the box, and developing a proper approach to solve technical problems. Many thanks to him for his guidance that accomplishing the work done in this thesis was a joyful experience.

I would like to thank my parents, Mr. Suresh Babu and Mrs. Gayatri Devi. Their love and continuous moral support helped me to accomplish this thesis. I would like to thank my sisters, Alka and Ruby for their motivation and encouragement. Lots of love to my baby niece, Svara. A special thanks to Rashmi for her encouragement during my research. Finally, I wish to thank all my friends in Canada and in India who supported me throughout my life.

Contents

List of Figures

List of Tables

List of Acronyms

AD	Administrative Domain
AH	Authentication Header
AKM	Automated Key Management
AS	Autonomous System
CKT	Crypto-Key-Table
DR	Designated Router
ESP	Encapsulating Security Payload
FIPS	Federal Information Processing Standards
GCKS	Group Controller/Key Server
GDOI	Group Domain of Interpretation
GM	Group Member
GSA	Group Security Association
GSAKMP	Group Security Association Key Management Protocol
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
ISAKMP	Internet Security Association Key Management Protocol
IPsec	Security Architecture for Internet Protocol
KARP	Keying and Authentication of Routing Protocols
KMP	Key Management Protocol

MRKMP	Multicast Router Key Management Protocol
NIST	National Institute of Standards and Technology
PAD	Peer Authorization Database
PBS	Perfect Backward Security
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PFS	Perfect Forward Security
PIM	Protocol Independent Multicast
RAPD	Routing Authentication Policy Database
RPAD	Routing Peer Authorization Database
RSPD	Routing Security Policy Database
SA	Security Association
SAD	Security Association Database
SIDR	Secure Inter Domain Routing
SPD	Security Policy Database
SPI	Security Parameter Index
TEK	Traffic Encryption Key

Chapter 1

Introduction

The Internet has become an important part of our everyday life. The number of services offered on it have benefited our society in various fields such as communication, education, medicine, business, banking, media and more. At the same time, it has made us dependent on its availability. It would be hard to imagine a world without the Internet.

The Internet is a global network. It is composed of many small sub-networks, inter-connected with each other to offer global connectivity. The services offered on the Internet can be accessed from any part of the world. However, to use an online service we need a way to reach to it.

Just like commuting in the real world for going from one place to the other, accessing services on the Internet also needs commuting. It is not us who commute but the requests that are sent to the online services that we want to access from our network devices (mobile handsets, laptops, PCs, etc.) that are connected to the Internet. In this case, our device becomes the source and the online service is the destination somewhere on the Internet. Finding a path from the device to the service on the network is the task of the routers. A router runs a routing protocol to identify the *adjacent routers* that are connected to it. Then, it determines how to inter-connect the separate networks to form the Internet.

A router finds a suitable path to a requested (online) service by using the "forwarding table" that has been built by the routing protocol(s) running on it.

Assuming that the service you want to access is available online, a router will always be able to find a suitable path to it, of course, with the help of the routing protocols. The routers route your request to the online service via this path. Similarly, the service sends a response to your request, but in this case, the destination is your device and the server is the source address. This is how we get access to the online services on the Internet.

We refer to an online service by name, e.g., www.example.com, but the routers do not refer to the services by this name. They need to know where the service resides on the Internet. With the help of the other support services, the service name is resolved into an Internet Protocol (IP) address. An IP address is the location identifier on the Internet, just like a regular real world address. However, these other support services are reachable only because some routing protocol has provided their location information and the router has suitable paths to these services. Clearly, the routing protocols play a key role in providing the online services.

There are various threats to the routing protocols that can disturb the routing on the network. For example, a rogue router can alter the path (misdirect the packets) or make a copy of a packet and re-use it later in a malicious way. Directly or indirectly, these threats affect the availability of the online services. Depending on the type of service, its unavailability can affect one or more organizations and their users at various levels (for example, economically, if a banking/business service).

Therefore, it is important to ensure that all the routers on a path are *legitimate*. It is also important to ensure that the *contents* of the routing protocol exchanges are valid (i.e., that the sending router has the right to share the information). There are security mechanisms available to secure the routing protocol from the kinds of threats noted above. These mechanisms range from as simple as using a password to strong cryptographic algorithms to secure the routing protocol exchanges. Once configured, these mechanisms protect the routing protocol exchanges.

Unfortunately, although security mechanisms are defined for most routing protocols, the configuring of the routers with the information that defines the parameters for the secure exchanges is today a manual process. The operators must manually configure the security mechanism for each routing protocol running on the routers in their network, in accordance with their organization's security policy.

This manual method of configuring security presents some security and deployment issues. Firstly, because of the need for manual intervention, once a security mechanism is configured, it is hardly ever changed. It is known that many organizations have been using the same configuration for over 5 years [28]. The prolonged use of a single security mechanism presents significant security risk. There are various ways to gain access to the security information configured on the router and utilize it to disrupt the normal functioning of the network [32].

Secondly, the lack of staff makes it hard for the operators to change the security mechanisms on all the routers in their network at the same time [27]. Therefore, it has become an operational practice to use the same security for a routing protocol across all the routers in the network. Thus, the lack of staff for carrying out such a large change in the network has become a major deployment issue. One way to overcome this problem is to use a solution that can address the configuration/distribution aspects of the security mechanisms for the routing protocols. However, there is no such solution available to address this issue for routing protocols. Lastly, since each router is manually installed and configured with the security information, mere possession of the security information is assumed to imply that the router is a valid participant in the network. There is at this time no effective mechanism that can be used to (independently) validate the router in a network. It is necessary to validate the router's participation in the network to assure that a potential attacker has not connected to the network without authentication. An attacker can stage large scale attacks if he is able to connect to the network without authentication.

The Keying and Authentication for Routing Protocols (KARP) Working Group [1] of the Internet Engineering Task Force (IETF) was established to identify solutions to these problems. The Working Group has developed guidance for those working on security for routing protocols, and has proposed solutions for automatically generating the keys to be used. However, they have not addressed the issue of validating the adjacent routers in the network, nor have they proposed any solution for the issue of managing the credentials that are needed to validate a particular router. In this thesis, we present a solution that addresses these problems. We call this solution "Routing Protocol Security" (RPsec).

The use of RPsec may benefit an organization economically by reducing the need for a large security staff. The use of RPsec solves the deployment problem currently faced by the operators, by enabling them to perform regular revision of keys or other security parameters. The use of RPsec will reduce the exposure of the routing protocols to the existing security risks. Given its ease of deployment, it provides an incremental approach towards a more secure Internet.

The rest of the thesis is organized in 10 chapters as explained below. Chapter 2 covers the background information of routing protocols and the associated security mechanisms. Chapters 3 and 4 cover the IETF standard key management mechanisms and the existing work that is going on in developing the key management protocols, an alternative to present manual methods, for the routing protocols, respectively. Chapter 5 explains the standard protocols that will be used for the proposed configuration and distribution mechanism. Chapter 6 discuss the motivation behind this work and explains the actual problem. Chapter 7 sheds more light on the solution proposed in this thesis. Chapters 8 and 9 contain in depth explanation of the RPsec framework. Chapter 10 explains the proposed architecture for the configuration/distribution of security mechanisms and some examples of using RPsec. Chapter 11 is the concluding chapter of our thesis.

Apart from these chapters, there are 6 appendices. Appendices A, B, C and D describe the YANG modules that we have developed for the RPsec. Rest of the appendices provide supplementary material to this work.

Chapter 2

Background

The agenda of this chapter is to discuss about the existing IETF standard security protocols in Section 2.2 and categorize routing protocols from a security perspective in Section 2.4. Then we discuss the generic threats to routing protocols in Section 2.5 and present security solutions used by routing protocols in Section 2.6.

2.1 Common Terms and Definitions

Hash function– A function that maps an arbitrary length of data to a fixed length string. In cryptography, the hash functions that are one-way and collision resistant, are used to calculate the *message digest*. One-way means it is impossible to find a message given its fixed length string representation. Collision resistant means its computationally infeasible to find any two distinct messages that produce the same output string.

Message Digest (MD)– It is the output of a hash function that accepts a variable length message and processes it (iteratively, using some defined algorithm) to generate a condensed representation of that message is called a message digest. The message digest protects the integrity of the message. Any change to the message will also change its message digest value [4].

Message Authentication Code (MAC)– The output of a hash-function that uses a secret key and a fixed-size block of data to produce a cryptographic code called message authentication code. This code is appended to the end of the original message before the transmission. This cryptographic checksum is calculated such that any change to the message after its calculation can be detected. This is possible because both the sender and the receiver of the message use the same secret key to calculate the MAC. The purpose of a MAC is to authenticate both the source of a message and its integrity [3].

Key Management Protocol (KMP)– Also known automated key management protocol (AKMP), "is useful for allowing simple, automated updates of the traffic keys used in a base security protocol. KMPs replace the need for manual management of keys and allow for periodically updating the key's on running systems. It also removes the need for a chain of manual keys to be chosen or configured on such systems. When configured properly, a KMP will enforce the key freshness policy among peers by keeping track of the keys lifetime and negotiating a new key at the defined interval" [27].

2.2 IETF Standard Security Protocols

2.2.1 TCP-MD5 and TCP-AO

TCP-MD5 This option was primarily used for protecting the communication of BGP routing protocol over the network using the MD5 message-digest algorithm. The MD is calculated over the TCP segment and a shared secret known only to communicating ends. The receiving end recalculates the message digest of the TCP segment including the shared secret and compares it with received signature. Any discrepancy will result in the rejection of the segment without sending a response. This provides protection against false connection resets and against segment spoofing or replay attacks by including the shared secret in calculation of message digest and associating sequence numbers with each segment. The use of this option is not negotiated rather, its use is dictated by the security policy for the communicating peers. The TCP-MD5 specification discusses nothing about the management of shared keys for BGP peers [33].

TCP-AO It was introduced to replace the TCP-MD5 [39]. It supports the use of stronger HMACs, provides increased protection against replay attacks (even for long-lived connections as it has increased size for the sequence number field) and coordinates key rollover between end points within a connection, i.e., without dropping the active connection. It is based on specification of master key tuple (MKT) and calculation of traffic keys for one or more TCP connections.

A MKT stores security parameters to be used to generate traffic keys and dictates the use of these keys for a connection. TCP-AO is used to secure BGP and LDP routing protocols. More information on TCP-AO can be found in [39].

2.2.2 IPsec

The Security Architecture for IP, a.k.a. IPsec, is designed to protect traffic at layer 3 (OSI) or Internet layer (TCP/IP). It provides interoperable, high quality, cryptographically based security for IPv4 and IPv6. It provides services such as access control, connectionless integrity, data-origin authentication, and detection and rejection of replays of the packets using two protocols, Authentication Header (AH) and Encapsulating Security Payload (ESP) [26].

The IPsec supports two modes of operation—transport mode and tunnel mode.

Both of these modes are actually the operational modes supported by the AH and ESP protocols. In transport mode, the security is provided to the next layer protocols, e.g., TCP and UDP. In tunnel mode, the whole IP packet is encapsulated in a new IP packet that is protected by AH or ESP.

The IPsec defines three databases—Security Association Database (SAD), Securtiy Policy Database (SPD) and Peer Authorization Database (PAD)—for effective specification of the security services it offers. It also has an optional automated key management mechanism—Internet Key Exchange (IKE) (Section 3.1.2), for dynamics Security Association (SA) establishment [25]. Below we discuss the role of these databases in IPsec:

SPD All the security policies for the incoming and outgoing traffic are dictated in the SPD. Each policy entry is identified by a Security Parameter Index (SPI) assigned to it at the time of configuration. The SPD entry contains the parameters used to identify the traffic (traffic selectors), the desired security protocols (AH or ESP) that should be used in order to protect that traffic and the operational mode of the IPsec.

SAD For effective protection, IPsec mandates establishment of an SA between the communicating peers. An SA in IPsec can be established manually or dynamically. Irrespective of the method used, all the SAs are stored as entires in the SAD. Effectively, it contains the final security parameters (as specified in the SPD corresponding to the SPI) that are used for securing the IP traffic. The applicable SA for the protected traffic is identified using the SPI. Each SA is pointed to by a corresponding SPI in SPD cache that is used to establish it. The SPI itself is enough to identify an appropriate SA for unicast communications. However for the multicast communication, it can be used in conjunction with the source and destination addresses of the packet to identify the most appropriate SA.

PAD The PAD provides the link between the SPD and key management protocols (KMP). It contains all the necessary parameters that are used for identification, authentication and authorization of each IKE peer or group that communicates with this IPsec entity.

In summary, the IPsec creates a boundary between the unprotected and the protected side of a host or a network. The traffic crossing this boundary is subjected to various checks to find the appropriate behavior (protect, bypass or discard) as per the security policies in place. Once the behavior is identified, the information from the SAD is used to protect the traffic as desired.

2.2.3 Multicast IPsec

Multicast IPsec is a little different from unicast IPsec. In multicast communication, a packet is sent to multiple destinations (one-to-many). This group of peers is identified using a group identifier in GPAD (Group PAD). The role of GPAD is similar to IPsec PAD but is specifically used for group authorization. For that the GPAD requires an explicit specification of a group identifier (GroupID) that uniquely identifies a multicast group. The GPAD is used by only a group key management protocol (GKMP, Section 3.2) to provide authorization services for the multicast group corresponding to the GroupID [42].

A multicast IPsec entity needs to identify the incoming and outgoing multicast traffic separately. This is because the multicast IP address can never appear in the source address field of an IP protocol packet. Since each sender may also be a potential receiver of the multicast traffic, the “directionality” is explicitly specified in group SPD (GSPD) entry for each traffic. This information was not required for the unicast IPsec because the source and destination addresses are swapped to represent the direction of the traffic. The directionality field allows to specify the traffic as sender-only, receiver-only or symmetric.

The GSPD can be used to support both unicast and multicast security policies [42].

One shortcoming to multicast IPsec is that the SA cannot be negotiated. It can only be assigned for a group by some central controller or some group key management protocol. Multicast IPsec is used for protocols such as PIM-SM [6].

2.3 Common Routing Protocol Functions

All the routing protocols share the following functionality to achieve routing and network reachability in a network–

Neighbor and Adjacency Management A router provides a way to reach to its directly connected networks. However, the networks that are not directly connected to it, can only be reached via some other router(s) that connects to those networks. Identification of such routers or neighbors is the first step to determine the network layout. The routing protocols need to know adjacent routers and then establish and maintain the neighbor relationships with them. Each routing protocol defines a set of parameters that should be met in order for a peer router to become its neighbor. The set of these parameters is called the peer eligibility criterion. Each routing protocol has implicit mechanisms to check for this criterion. The routing protocols thus establish and maintain neighbor relationship(s) with routers that qualify according to the eligibility criterion. The messages used for neighbor discovery and maintenance are called routing protocol's control packets.

Different routing protocols use different approaches to find and maintain neighbors, usually they fall under one of the following:

- Simply broadcast/multicast the update packets. These packets are sent to routing protocol's broadcast/multicast address, where the potential neighbors must be running the same routing protocol to process these update packets. Other routers/devices simply drop these packets.

- Search/maintain neighbor adjacency using “hello” packets. Hello packets also serve as keepalive messages that are sent at regular intervals to insure that the neighbor is up and running. If keepalives are not received for a certain period of time, the neighbor is declared unreachable and the routing protocol send updates to other peers to spread this information.

- Manual configuration of neighbors. For example, BGP cannot dynamically search for its neighbors. The manual configuration of neighbors is used by the routing protocols as soon as their instances are up and running.

Once the neighbors are discovered, routing protocols share the network reachability information.

Network Reachability and Routing Information Maintenance Routers run routing protocols to exchange the information about the directly connected networks as well as to process the network information they receive from the neighbors. Each router uses this information to create a forwarding table. The router uses the forwarding table to route all the traffic passing through it onto the network. The messages carrying routing information from peer routers are called routing protocol’s data packets.

Routing Transports The routing protocols perform all the above functions by transmitting messages to the router’s neighbors (or potential neighbors) using some underlying transport protocol [8]. For example BGP uses TCP, RIP uses UDP, and OSPF and PIM use IP as transport protocol. Another important thing to note is that each routing protocol defines its own message format. These messages are then encapsulated in the transport protocol before being transmitted on the network. Usually these messages are link-local, i.e., they are not forwarded by a router.

Security Aspects Since the routing protocols play an important role in connecting isolated/partitioned networks, it is necessary to protect the message exchanges between them. Present day routing protocols use a shared key (or a shared secret) to ensure that the peers are authenticated and authorized to talk to them, and to protect the integrity of the messages exchanged between them. For message integrity, it either relies on some built-in or external security feature.

It should be noted that confidentiality is not desired for routing protocol messages because encryption may increase the time it takes to create and maintain the routing table. Also encryption may obliterate the concept of priority/critical routing protocol messages that must be processed before other messages.

2.4 Classification of Routing Protocols

As discussed previously, a routing protocol needs authentication, authorization and message integrity for securing its communication with the peer(s).

Authentication insures that the peer is “who it says it is”. Authorization means the peer is allowed to communicate with this routing protocol entity. Message integrity insures that the message sent by the peer did not change in transit.

Presently, routing protocols use shared-secret among devices to perform mutual authentication and authorization. The shared key is also included in message digest calculation to verify the source of the message. However, the methods of authentication and authorization for any routing protocol are dependent upon the following factors:

Communication Model The routing protocols can send messages to its peers using following communication models:

One-to-One It is also known as unicast communication. In this type of communication the source address and destination address in the packet belong to the devices at either end of the communication. In unicast communication, both the participants can be the sender and the receiver of the messages. For example, routing protocols such as BGP and OSPF on NBMA networks.

One-to-Many Multicast communication follows this type of communication model. It is different from the unicast communication in that the source address belongs to a sender of the packet but the desitnation address belongs to a group of hosts. This group of hosts is actively listening for any packet with the destination address set to the multicast address that uniquely identifies this group. In multicast, a host may be sender-only, receiver-only or both. Sender only means the host is the transmitter of information. Receiver only means the host only receives the information but never transmits it. In the third type, the host is the sender as well as the receiver of the information transmitted by its peers belonging to the same group. For example, routing protocols such as OSPF on broadcast networks and PIMs use this communication model.

Key Scope From a routing protocol's point of view, the key scope for a key defines the limit on the use of that key in the network. The key scope from a KMP's point of view describes the way that key should be configured, negotiated or distributed for use on the devices in the network. In general, there are two types of key scope as follows:

Peer Keying In peer key scope, the key is unique between two routers running unicast routing protocols. From a KMP's perspective, the key in this scope needs to be configured on or negotiated between two peers.

Group Keying In group keying, the key is shared and used by multiple routers simultaneously. Group keying exists for routing protocols using multicast and broadcast communications [27]. From a KMPs perspective, the key in this scope needs to be configured or distributed among the members of the multicast group.

An in-depth discussion on key scopes and its effect on key management can be found in [38].

2.5 Threats to Routing Protocols

Routing protocols are subject to various threats as follows [8]:

- Threats at the routing transport level.
- Attacks on the messages that carry control information (adjacency and peering information).
- Attacks on the messages carrying network information.

At the routing transport level, an attacker may attack the routing transport subsystem to disrupt the routing protocol message exchanges. An attacker may capture and replay the routing protocol control packets that are used for searching neighbors and adjacency management. An attacker may infuse incorrect information in the routing protocol data packets that contain the network reachability information. Such attacks are intended to deceive the routing protocols, i.e., spread bogus routing information in the network. This could also lead to the disclosure of routing information and disruption in network services. An IETF working group, Secure Inter Domain Routing

(SIDR), is working on ensuring authenticity of information in the routing protocol data messages. These attacks may have following consequence on the overall network as identified in [8]:

- Disclosure: An attacker will gain more understanding of the network infrastructure if he is able to successfully capture the routing protocol packets. Such information may help in mounting a large scale attack against an organization's network.

- Deception: If a legitimate router is said to be deceived if it fails to check the authenticity of the forged messages and believes it to be authentic. A forged information can infuse incorrect or insecure routes in the network.

- Disruption: An attacker can disrupt the normal functioning of the routers in the network. This can be done by inserting, corrupting, replaying, delaying or dropping routing messages or by breaking routing sessions between legitimate routers.

- Usurpation: It happens when an attacker is able take control of a legitimate router and services running on it. Such a compromise can be used to various advantages and to harm a user or organization on a large scale.

An attacker can mount such attacks on the network from a close proximity or by taking control over the router from some remote location. Therefore, it is extremely important to authenticate the router participation in the network as well as reviewing the security policies at regular intervals.

2.6 Security Mechanisms for Routing Protocols

Most of the security solutions for routing protocols are based on calculating message digest or HMACs over the routing protocol packet and a shared secret that is used to identify the source of the message. This is because the routing protocols require only peer authentication and assurance that the message sent by the peer has not changed in the transit. RIPv2, OSPFv2 and OSPFv3 have a built-in mechanism called authentication trailer. BGP and OSPFv3 depend upon the transport subsystem TCP and IP, respectively, for securing message exchanges.

Table 1 below, classifies the routing protocols based on their security type. We

Routing protocol	Key scope	Communication type	Security feature	Standard
BGP	Peer keying	Unicast	OoB	TCP-AO
RIPv2	Group keying	Multicast	Built-in	AT
OSPFv2	Group keying	Both	Built-in	AT
OSPFv3	Group keying	Both	Built-in	AT
OSPFv3	Group keying	Both	OoB	IPsec
PIM-SM	Group keying	Multicast	OoB	IPsec

Table 1: Classification of routing protocols

Legend- AT:Authentication Trailer; OoB:Out-of-Band; Both:Unicast and Multicast

discuss more about the available security solutions in the following sections.

2.6.1 Built-in security

In this section we discuss about the most commonly available built-in security mechanism—Authentication Trailer (AT).

AT is the message digest or HMAC calculated over the routing protocol message including the shared secret. This is attached at the end of the routing packet before transmitting over the network. The receiver of the message detaches the AT from the message and recalculates the digest over the rest of the message. If the calculated digest matches the AT, then message is believed to be received from an authenticated source unchanged.

RIPv2 The original RIP specification [7] suggests to use password authentication or keyed-MD5 security. The RFC4822 [5] specification updates the RIPv2 cryptographic authentication by adding support for the SHA family hash algorithms in place of keyed-MD5 for increased security.

The processing of the packet depends on the type of hash-algorithm (SHA or keyed-MD5) chosen for security that is identified by the key identifier in the packet itself. The resultant MAC is attached at the end of the routing protocol packet.

RIPv2 security specification supports key rollover by allowing configuration of multiple SAs on each peer [7]. The key management for RIPv2, however, relies upon manual configuration or some private (vendor specific) key management method.

OSPFv2 The cryptographic authentication for OSPFv2 dictates the use of a shared secret key to verify a message digest that is tagged at the end of the OSPF packet before transmission [33]. It is the same as calculating message digest over the routing protocol packet and shared key as discussed in RIPv2. OSPF also supports SHA family of hash functions for increased security compared with MD5 [11].

The shared secret is never sent over the network in clear text thus provides security against passive monitoring. An OSPF SA has a set lifetime. Thus multiple SAs can be configured with overlapping lifetimes to facilitate key rollover.

It should be noted, however, that neither specification does indicates any standard key management method. It is safe to say that the security association management is based on either manual configuration or some type of vendor specific method.

OSPFv3 RFC6505 offers OSPFv2-like authentication trailer security for OSPFv3 protocol [12]. This specification extends the OSPFv3 packet formats to include an AT option bit and authentication data, to afford the authentication trailer.

The SHA-family of hash functions are supported by this specification. Like other AT solutions, OSPFv3 also uses a shared secret among its members for verification and authentication of data. This specification also offers configuration of more than one SA with overlapping lifetime parameters for smooth transition from an old SA to a new one.

OSPFv3 also has a security specification using IPsec which is discussed in the next section [18].

As in others, this solution also lacks use of any automated mechanism for key management thus relying on some vendor-specific or manual key management solution.

2.6.2 Out-of-Band security

The out-of-band security mechanisms for routing protocols are provided by the routing transport protocol. BGP depends on TCP for security while both OSPFv3 and PIM depend upon IP for security. As discussed in Section 2.2, TCP provides two options TCP-MD5 and TCP-AO for securing BGP. The IPsec protocol is used for securing OSPFv3 and PIMs.

BGP The BGP uses TCP as its transport protocol. TCP is a reliable transport layer protocol (OSI layer 4) that provides end-to-end data delivery.

The TCP-MD5 SA simply specifies the use of a shared secret when calculating the message digest for TCP-segment and provides an incrementing sequence number for each BGP packet sent over the network. This sequence number is used to provide protection against replay attacks. TCP-AO on the other hand consists of one or more MKTs as SA between BGP peers. TCP-AO offers more security features using MKTs as compared to TCP-MD5. In that, it specifies use of strong MACs, increases the length of sequence numbers for protection against replays even for long lived connections and facilitates key-rollover within an ongoing connection [39].

TCP-AO manadates generation of four unique keys for a BGP connection out of which only three are used at either end depending on who initiated the TCP connection. These traffic keys are used to calculate the MAC of individual TCP segments. TCP-AO supports use of static MKTs and explicitly recommends use of external key management mechanims [39].

OSPFv3 OSPFv3 uses IPv6 as its routing transport and IPsec is used to secure the OSPFv3 messages [18]. As OSPF uses both one-to-one and one-to-many communication model (Table 1), using a key management protocol poses a challenge for OSPF protocol. For key management, the specification RFC4552 mandates using manual keying and only symmetric SAs are configured among OSPF peers for securing group communication [18]. It also suggests use of group key management protocols if/when available.

PIM PIM uses IP as its transport mechanism. RFC4601 and RFC5796 specify IPsec for securing PIM's link-local messages. Both AH and ESP can be used to protect PIM's link-local messages [17], [6].

PIM is protected using multicast IPsec with manual keying. However, use of automated key management is suggested if/when available.

2.7 OSPF Security Overview

Table 2 shows the IETF standard security protocols for both–the OSPFv2 and the OSPFv3 routing protocols.

OSPF Version	Security Protocols	Standard
OSPFv2	Keyed-MD5	RFC 2328
OSPFv2	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	RFC 5709
OSPFv3	IPsec	RFC 4552
OSPFv3	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	RFC 7166

Table 2: IETF standards for OSPF security

OSPFv2 and OSPFv3 use IPv4 and IPv6 as their transport subsystem, respectively. RFC2328 is the specification of OSPFv2 that suggests the use of Keyed-MD5 for its security. Later, RFC5709 introduced use of HMACs for its security. HMACs are considered more secure compared to MD5. The steps for computing the HMACs are similar to the steps for computing MD5, only the security algorithms have changed.

RFC2740 is the original specification of OSPFv3 that suggests use of IPsec for security. RFC4552 standardized the use of IPsec for OSPFv3 security. RFC5340 replaced RFC2740 as a new specification for OSPFv3 routing protocol. However, among the various changes, the security considerations of OSPFv3 are not changed. As discussed in Section 2.6.1, OSPFv3 also have an AT-like security option initially suggested in RFC6506. RFC7166 replaced RFC6506 and is the current standard for an AT-like security for OSPFv3. Among all the changes, the steps for calculating the AT remains the same as in RFC2328. Let us discuss OSPFv2 security. OSPFv2 has 3 choices for the security built into its framework [33]—null authentication, simple password authentication and cryptographic authentication. Null authentication means no authentication. The OSPF packet is sent as it is on the network. The use of this option is not recommended. In case of simple password authentication, each OSPF packet contains a password in clear text. The same password is used for all the packets. Clearly, not a secured choice. In cryptographic authentication, one can choose between Keyed-MD5 (RFC2328) and HMACs (RFC5709). Presently, the operators manually configure the choice of the security protocol. It could be any one of the above depending upon the security requirements.

Chapter 3

Existing KMP Standards

In this chapter we give a brief introduction to key and security association management. Then we discuss existing IETF key management standards. Lastly, we discuss the present key/SA management method for the routing protocols and the need of the KMPs for the routing protocols.

The term key and SA management refers to the establishment of a set of keying parameters that is required to use some cryptographic algorithms. These cryptographic algorithms are used to provide security services such as authentication, confidentiality and integrity to important information shared in the unsecured network environment [10]. The protocol that performs key and SA management is called a key management protocol (KMP). KMP is also referred to as automated key management (AKM) protocol.

The way a KMP performs the key management tasks depends upon the type of communication model (as discussed in Section 2.4) it serves. There are two types of KMPs—unicast KMP and group KMP. The unicast KMPs establish SAs for unicast communication and the group KMPs (GKMP) for group communications. We discuss existing unicast KMPs in Section 3.1 and group KMPs in Section 3.2.

3.1 Unicast KMP

Unicast KMPs follow a simple one-to-one communication model. In that, the SAs are negotiated and established between two devices only. Separate SAs may be established for more than one unicast sessions between two peers. Similarly, multiple security associations may be established for each adjacent peer. In the following sections we discuss the IETF standard unicast KMP solutions.

3.1.1 ISAKMP/IKEv1

The Internet Security Association and Key Management Protocol (ISAKMP) defines the procedures for authenticating a peer, creation and management of Security Associations, key generation techniques, and security from denial of service and replay attacks [31].

ISAKMP provided a clear separation of SA management from the details of key exchange. A separate specification describes key exchange mechanism called Internet Key Exchange (IKE) for it [19]. This separation was to allow many different key exchange approaches to ISAKMP. It provided a common set of attributes required for establishing SA for AH/ESP security protocol between two peers.

It works in two phases. The first phase provides peer authentication and authorization, and establishes an SA to protect phase 2 communication. The phase 2 communication is used to establish SAs for other data security protocols. ISAKMP describes various payloads that are used in these phases to establish and maintain SAs. The security domain of interpretation (DOI) for ISAKMP described in [35] identifies the situations when a particular ISAKMP payload and associated attributes will be used and how it will be interpreted for securing IP traffic. The key exchange mechanism for ISAKMP is described in the Internet Key Exchange [19].

For consistency—the security domain of interpretation for ISAKMP [35], ISAKMP

[31] and IKE [19] together are called *Internet Key Exchange version 1* (IKEv1). Therefore we will refer to ISAKMP and associated support references as IKEv1 throughout the rest of this document.

3.1.2 IKEv2

The original Internet Key Exchange is specified in IKEv1 as discussed above.

IKEv2 included major changes that were learned from operational problems of IKEv1 [25].

Like IKEv1, it establishes SAs for AH and ESP security protocols between two peers in two phases—IKE_SA_INIT and IKE_AUTH exchanges. In IKE_SA_INIT exchange, the IKEv2 peers negotiate security parameters for the IKE SA, and exchange the values required for key derivation. In IKE_AUTH, the IKEv2 peers use a derived secret to prove their authentication and set up an SA for AH or ESP security protocol.

IKEv2 is simple, more reliable and is not compatible with its predecessor. Most of the key management protocols described in this thesis use the services of either IKEv1 or IKEv2 for authentication and authorization purposes. From here onwards, any reference to IKE should be treated as IKEv2 unless specified otherwise.

3.2 Group KMP

The GKMPs provide group key and SA management for peers communicating in a group. In that, the GKMP provides secret keys for each group for which the device is a member. A device thus maintains secret keys for each group it is participating in. Unlike unicast KMPs, keys and SAs cannot be negotiated in group communication. Instead, the SAs are assigned externally and then shared among the group members.

The GKMP-Specification [22] and GKMP-Architecture [21] together specify an architecture for GKMPs. Other specifications, MSEC-GKMP-Architecture [9] and AAKM-RP [38] provide multicast security architectures for group key management. All these architectures describe common roles and responsibilities for the devices participating in GKMP operations as follows:

Group Controller (GC): A group controller has the authority to generate, distribute and rekey (an act of replacing an old key with a new key) cryptographic keys for a multicast group [22]. It is the responsibility of GC to ensure that each participant is valid and posses correct cryptographic key that is being used for group communication. A group controller should also be a valid participant in order to perform this role.

Group Member (GM): A valid participant who is not acting as a GC is a group member [22]. The group members assist the GC in creation of keys, request and accept group keys and perform mutual authentication to validate each other. A GC is also a GM.

Group Controller Key Server (GCKS): A GCKS is a network device that generates and distributes group keys and group security policies for one or more multicast groups. Its functions are similar to that of a GC. The GCKS is an important component in a multicast security framework architecture described in [9]. The terms GCKS and GC are used to refer to the same thing. We will use the term GCKS for consistency.

Policy Server (PS): A policy server stores the security policies for various multicast groups in a network environment.

In the next sections we describe the IETF GKMP standards.

3.2.1 GDOI

Group Domain of Interpretation (GDOI) is a group key and SA management protocol that adheres to the mulitcast security architecture described in [9]. It is based on the ISAKMP DOI and extends ISAKMP payloads for group key management [31]. GDOI has two phases of negotiation. The first phase uses the ISAKMP phase 1 procedure, which provides authentication, authorization and an SA that is used to protect the phase 2 exchange. It defines two new protocols—GROUPKEY_PULL and GROUPKEY_PUSH—that provide phase 2 exchange for group key management. The GM uses GROUPKEY_PULL to retrieve group SA (GSA) from a GCKS. The second protocol is used by the GCKS for rekeying the existing SAs.

3.2.2 GSAKMP

Group Secure Association Key Management Protocol (GSAKMP) provides a security framework for managing cryptographic keys for group communication [20]. It is used to protect multicast application data.

GSAKMP assigns the same roles and responsibilities to the devices as discussed in Section 3.2. However, it also introduces the notion of a Group Owner (GO). The group owner is responsible for creating the security policy rules for a group. The GCKS's role is to perform key management operations by adhering to the security policy received from GO. GCKS also enforces the security policy on all the GMs as dictated in the policy. For effective security both the GCKS and the GM mutually authenticate each other before any key management related exchanges take place. The GCKS validates the GM's request by making sure that the GM is listed as a participant in the group it requested. The GM on other hand validates that the GCKS is authorized to represent the group it requested to join.

GSAKMP also provides an optional distributed key management architecture using a subordinate GCKS. The subordinate GCKS's role is to provide scalability and administer the distribution of the security policies in a large network.

3.3 Present Key Management and Need of KMPs

At present, the key/SA management for the routing protocols is done manually. In that, the operators configure the key/SA on device-by-device basis. The operators often face the following problems with manual key/SA management—

- Dependency on manual keying methods only. No automated key management protocol is available for the routing protocols.
- Manually accessing and configuring SAs on routers implies the authenticity of the device in the network.
- Manual method of changing the key/SA cause a router to drop the active connections with its peers.
- Change of SA in one router means changing SA in all the neighboring routers.
- As a drawback of manual method, the operators have to configure one SA per routing protocol. The same SA is configured on all the neighboring routers configured to run that routing protocol.
- Manually accessing each router and changing the keys/SA is a labor intensive task.
- Also, the lack of staff makes it difficult to implement such changes across the network.

These problems demonstrate the need of KMPs for routing protocols. In previous sections, we studied the standard unicast and group KMPs. Accordingly, a KMP automates the key/SA negotiation and establishment. It authenticates a peer before starting the key/SA negotiation. A KMP also provides automatic key/SA rollover, rekeying and renegotiation of the SAs. But, all these standards lack one thing–support for the cryptographic protocols that are used to secure the routing protocols. It is because these standards were not built for the routing protocols, but for IPsec.

Some of the existing security solutions for routing protocols either specify the need of an external key management mechanism or simply leave it to the implementers to use manual or private key management solutions. This calls for a management scheme that can allow the operators to perform the key/SA management for all the routers without accessing each router manually.

Chapter 4

Existing Work

In Chapter 3, a need for KMPs for routing protocols is identified. The agenda for this chapter is to discuss the existing/on-going work in the KARP working group of the IETF to fulfill that need.

4.1 KMPs For Routing Protocols

The existing work on developing KMPs for routing protocols relies upon the KMP solutions discussed in Section 3.1 and Section 3.2. The sections below provide a brief introduction to existing work towards developing KMPs for routing protocols.

4.1.1 RKMP

Negotiation for Keying Pairwise Routing Protocols (RKMP) [24] describes a mechanism to secure the unicast routing protocols using IKE. The focus is exclusively on unicast routing protocols such as BGP, which uses TCP as its primary routing transport. The premise for this work is based on the fact that the key management for the unicast routing protocols today is limited to static configurations.

Towards this goal, RKMP provides modified IKE payloads to add support for

TCP-AO. As with IKE, the RKMP works in two phases. The first phase is the same as the IKE phase 1 exchange. The second phase—RP_AUTH—is similar to IKE's IKE_AUTH exchange but uses the extended IKE payloads.

RKMP is effectively an extended IKEv2 that supports authentication of routing protocols and also cooperates with a key management database such as CKT (discussed in Section 4.2) for storing the negotiated SAs for routing protocol.

4.1.2 G-IKEv2

Group IKEv2 (G-IKEv2) proposes a group key management approach between a GCKS and a GM using IKEv2. It adheres to the multicast security architecture described in [9].

G-IKEv2 has a minimum of two exchanges IKE_SA_INIT and GSA_AUTH. The IKE_SA_INIT exchange is the same as IKEv2 phase 1 exchange. The GSA_AUTH exchange however has more features than the IKE_AUTH in that it also provides options for group member registration and group authorization. One key difference between GSA_AUTH and IKE_AUTH is that group SAs are not negotiated but are downloaded from the GCKS to the GM.

G-IKEv2 is an easier, reliable and a robust protocol, which is intended to replace GDOI. It is indeed a GDOI version 2 which is better in performance than its predecessor because it requires fewer message exchange(s) to establish a group security association.

It should be noted that G-IKEv2 does not provides an authentication list of peers for adjacency management because it is not made for securing routing protocols. It also assumes that the identity of an existing GCKS in the group will be provided by some external mechanism. Section 4.1.4 explains such a process that is used to dynamically elect a GCKS in a group.

4.1.3 G-IKEv2-MRKM

The G-IKEv2-MRKM is known as *G-IKEv2 for the Mulitcast Router Key Management*. It proposes the use of G-IKEv2 to protect routing protocols such as OSPF and PIMs, between a group of network devices [40].

More specifically it proposes extensions to G-IKEv2 payloads that carry group security policies. It advocates to include the support for routing protocols specific security protocols and tranforms in G-IKEv2 policy payloads. The MRKM supports use of both static GCKS configuration and dynamic election of GCKS as suggested in the next section.

4.1.4 MaRK

MaRK stands for *Multicast Router Key Management*. It proposes a key management method for multicast routing protocols. It defines a group protocol for establishing and managing symmetric keys for multicast routing protocols. It also provides an election protocol for GSA management protocols that may also be used for protocols such as G-IKEv2 and MRKM.

The election process uses a GCKS priority number defined for each GM. A high number means higher chances of becoming a GCKS. The elected GCKS listens for the group-join requests from the GMs for the groups it manages. This protocol needs at least two phases to download a GSA from GCKS to GM. The first phase uses the IKE_SA_INIT and IKE_AUTH to establish mutual authentication between a GCKS and a GM. In the second phase, the GCKS processes group join requests from GMs and performs group key management functions.

It has features similar to G-IKEv2 and is intended to be used for security of routing protocols involved in a group communication.

4.2 Cryptographic Key Table

The cryptographic key table database, also known as *crypto-key-table* (CKT), is a standard conceptual database of long-lived cryptographic keys intended to be used by many different routing protocols for securing routing information exchange [23].

It is designed to support both manual and automated key management methods. The standard aims to provide a simplified specification of keys, security protocols and associated key derivation functions (KDF) for routing protcols while allowing different implementation approaches to use this database. There is a similarity between SAD and CKT. The SAD database is built to store security associations for AH and ESP security protocols that are used to protect IP traffic. Similarly, the CKT focuses on storing SAs for all the security protocols that are used to protect routing protocols. The CKT has achieved it by its generic design, which allows storage of all the important information required to use any security protocol available for routing protocols.

The CKT also specifies the conventions for the representation of keys and identifiers such that all the implementations represent the information in the same way. The keys stored in CKT are called long-lived keys. The implementations are required to use this key to generate traffic keys specific to the communication wherever required. However, it does not constrain the direct use of the long-lived keys if required.

4.3 Routing Authentication Policy Database

Routing Authentication Policy Database (RAPD) [44] provides a concept of a database that provisions policies for KMPs for routing protocols and uses CKT for storing negotiated keys and security associations. The RAPD also provides for authorization policies that contains information to validate peers. We extend the RAPD's concept in this thesis.

The RAPD provides peer/group authorization information, specifies KMP/manual method and cryptographic protocols for establishing SAs for routing protocols. It provisions security policies that a KMP can use to negotiate a most preferred SA for a routing protocol. The overall use of RAPD is similar to IPsec's SPD and PAD databases. A KMP mechanism for routing protocols may utilize the security policies and authorization information provided by the RAPD to secure the routing protocol communication.

Several organizations of RAPD are possible. It can be implemented with two different databases for incoming and outgoing policies or, as a single database that contains all the (symmetric) security policies and authorization information. The third type of organization is to divide it into two databases similar to SPD and PAD in IPsec.

The third type of RAPD organization is illustrated in this thesis. Two databases are specified—Routing Peer Authorization Database (RPAD) and Routing Security Policy Database (RSPD)—that together with the CKT reproduce an IPsec like security framework for routing protocols. Chapter 8 explains in detail about these two databases and the security framework for routing protocols.

Chapter 5

NETCONF and Yang

The agenda for this chapter is to present a brief introduction to NETCONF, YANG model and PYANG, a YANG model validator.

The *Network Configuration Protocol* (NETCONF) [16] and YANG [13] are IETF standards for network configuration management. The former is a configuration management protocol and the latter is used to model the data managed by NETCONF [41].

5.1 NETCONF

NETCONF is an IETF standard for network configuration management. It provides a basic set of operations for installing, manipulating and deleting the configuration of network devices [16]. It follows the principle of the client-server architecture and provides simple remote procedural call (RPC) based operations such as get, get-config, edit-config, copy-config, delete-config, etc., to configure, edit and delete the configuration data of network devices. It uses *Extensible Markup Language* (XML) based data encoding for configuration data and protocol messages. It provides multiple logical data-stores such as writable-running, startup and candidate data store. These data stores represent the state of the configuration data in a device [16]. Each of these data-stores can be configured independently, locked and unlocked to ensure safe manipulation and consistency of the configuration data.

It has all the features that are required for remote configuration management, clearly a protocol made for configuring network devices.

However, it had lacked a specific data modeling language that can be used to model the configuration and the state data, the RPC calls and notifications. This forced the vendors to develop private solutions (mostly XML based) that can closely meet the NETCONF operation requirements. The closest match to the NETCONF protocol type functioning can be achieved using XACML.

5.2 XACML

XACML stands for *extensible access control markup language.* The XACML is an OASIS standard to define a core schema and corresponding namespace for the expression of authorization policies in XML against objects that are themselves defined in XML. XACML defines two things that are important from NETCONF's perspective. It defines an XML schema for defining a network information, syntax of requests, access control policies and responses [36]. This can be used to implement the NETCONF RPCs and notification functions. Secondly, it specifies how a request from the policy requester should be processed and replied to, by a policy distribution system [36].

XACML allows defining new nodes (data or configuration) in the existing schema. Thus network data types such as IP address, port number, device identity and other management relevant parameters can be defined when required.

It is based on XML, which can be used to express a wide range of policies. While such scope is desired, it presents a problem of reaching a general consensus [30] Also, it was found that different vendors had developed a different data modeling approach in XML to meet NETCONF requirements. Thus, the solution using XML lacked uniformity across the network community [30].

The NETCONF like functionality can be achieved in XACML using XML schema definition (XSD). "While it was a viable solution it had some limitations: The XSD is used for modeling of arbitrary XML documents while NETCONF needed modeling management commands, notification, management information and related behavior. Then defining XSD is complex and is not readable enough for general population" [30].

Thus the IETF came up with a more appropriate easy to read but more effective data model for NETCONF—YANG data model.

5.3 Data Modeling Language

Data modeling languages are used to develop and specify configuration data for the managed systems. The data models are used by the policy developers and the network operators for managing network services.

It is imperative to choose an appropriate data model for the security policies that provides an easy and concise format to express policies, and benefits both the implementor and the reader or the network operator managing those policies.

5.3.1 XML

Section 5.2 explains why using XML as a data modeling language for NETCONF is a less appropriate solution. The aim is to use the data model that can be easily understood by the creator, reader and implementor of the security policies. Towards this goal, the IETF developed Yang data modeling language for NETCONF as explained in the next section.

5.3.2 YANG

YANG is a heirarchical and modular data modeling language for NETCONF. It is also used to specify NETCONF-based operations (RPCs, e.g., get, get-config, edit-config, copy-config, delete-config, etc.), configurations, state data and notifications. It models heirarchical organization of the data as a tree, where each node has a mandatory description. Further, data can be structured into modules and submodules. The modules are reuseable, extensible and importable in other modules (similar to including external libraries in programming languages). It provides a set of built-in types and allows for defining the new types/derived types from the built-in types. The new types can be further used to derive other dervied types. It allows for constraining the form of data such as appearance, value and condition based processing of data. Its beauty lies in the fact that it is not mandatory to use the YANG representation itself as it can be translated into an equivalent YIN (YANG Independent Notation) notation which can be parsed using any standard XML parser. YANG also supports versioning of modules to indicate that changes are made to the module. It organizes its data into nodes, leaf nodes, leaf-list nodes, container nodes and list nodes. It allows for easy distinction between state data and configuration data [13].

5.4 PYANG

PYANG is an extensible YANG validator, transformater and code generator written in Python. It is used to validate YANG modules and used to convert YANG modules into equivalent YIN, YANG, DSDL and XSD formats. It can be integrated in other applications to generate equivalent codes corresponding to the module [2]. PYANG is free open source tool available for the YANG model validation at [2]. A similar tool written in Java—jYang is also available now which provides the same features as PYANG [2]. We choose PYANG to validate the YANG modules that are used to represent the security policies suggested in this work.

Chapter 6

Problem Statement

In this chapter we will understand the existing layers of security management for the routing protocols. We will also identify the deficiencies in these layers that have motivated the work done in this thesis.

6.1 Security Management Framework

Figure 1 shows a possible security management framework for the routing protocols. Let us discuss the layers in the figure in detail.

Layer 1 is the routing protocol layer. All the routing protocols discussed in Chapter 2 exist on this layer. The routers run routing protocols among themselves to collect and distribute topological information for the network. The routing protocols distribute the network information by "exchanging messages" with the peer routers (neighbors). Each router processes all the information received from the routing protocol peers to create and maintain the forwarding table. This forwarding table is used to decide where to forward a particular packet when it arrives.

Layer 2 represents the security mechanisms (Section 2.6) available for a routing protocol. All the relevant security mechanisms have been discussed in Chapter 2. A

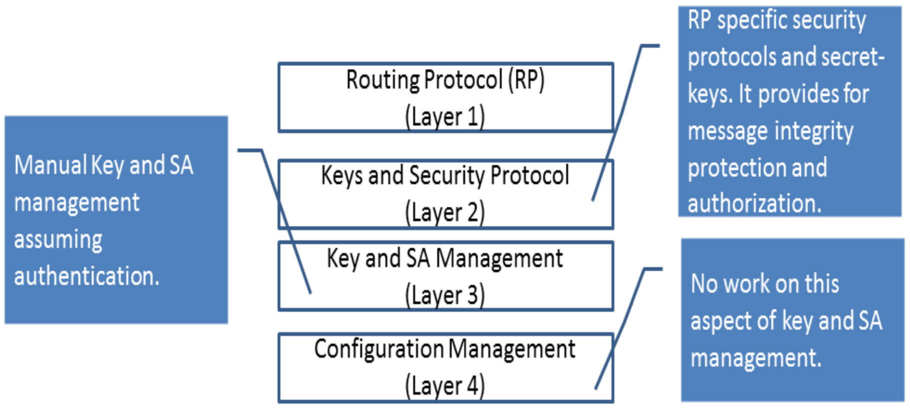
Routing Protocol (RP) (Layer 1)
Keys and Security Protocol (Layer 2)
Key and SA Management (Layer 3)
Configuration Management (Layer 4)
Manual Key and SA management assuming authentication.
RP specific security protocols and secret-keys. It provides for message integrity protection and authorization.
No work on this aspect of key and SA management.

Figure 1: The present security management framework for routing protocols

routing protocol needs to be assured of two things about the messages that it receives from its peer routers:

1. that the peer is legitimate, and
2. that the message from that peer has not been altered in transit.

The most common approach today is for a routing protocol to use a pre-shared key for authorizing its neighbors as well as for validating the message integrity. As discussed in Chapter 2, the security mechanisms calculate the HMAC/MD using this key for each routing protocol message that is sent and received by the router. Any modification to the message will change its digest and thus, the routing protocol/router can detect and discard the corrupt message. In effect, all the neighbors (running the same routing protocol) that possess this key are authorized to communicate with each other.

The configuration of keys/SA, the choice of keys and the security mechanism used for a routing protocol depend on the key management methods at Layer 3. As discussed in Section 3.3, the network operators use the manual key/SA management method, which is the only solution that is available at this time for routing protocols. For this reason, the operators face various deployment problems. The operators configure keys/SA for a routing protocol by manually accessing each router in the network. This is a highly labor-intensive task when there is a large number of routers in a network. There is no flexibility for the operators to specify more than one SA per routing protocol. Changing the SA of a routing protocol requires change in all the routers that run that routing protocol. Also, a change in the SA drops the active connection. To avoid this, operators must change the SA simultaneously across all the routers in the network which is impossible to achieve manually.

These limitations demotivate the operators from performing regular key/SA management for the routing protocols. According to the report in [28], the operators are using the same key/SA throughout the network for over 5 years. A capable attacker can successfully perform cryptanalysis to obtain keys, security protocol and other relevant information in this duration. It is also known that passwords to access the routers are sold in the underground economy [32]. If an attacker gains access to this information, he will be able to compromise the security of the routing infrastructure. As we learned in Chapter 2, any router that possesses the keys/SA is allowed to participate in the network operations. Therefore, an attacker can use a rogue router to participate in the network operations undetected. At this point, an attacker may also stage attacks such as spoofing, man-in-the-middle attack and distribute false routing information among routers. Therefore, the operators should perform regular SA management to avoid information leakage due to cryptanalysis attacks and from other sources who exposed the sensitive security information to the world. KMPs can be used to mitigate these threats, as discussed in Section 3.3. There is work in progress to develop KMPs for the routing protocols (discussed in Chapter 4), but no concrete solution has been standardized yet. Clearly, the routing protocols are exposed to security risks at Layer 3.

Layer 4 focuses on the configuration and the distribution aspects of keys/SAs for the routing protocols. Presently, configuration of routers is done on a device-by-device basis, by physically or remotely accessing the device. There is no easy-to-use solution that controls all the aspects of the key/SA management for the routing protocols in a network. In fact, there is no information base that allows specification of all the keys/SAs for the routing protocols at one place. Since the information base that needs to be managed is missing at Layer 3, there is no way to control the management aspects of keys/SAs for the routing protocols. Therefore, there is no mechanism for the remote configuration and distribution of the keys/SAs across the routing infrastructure. Practically, there is no 4th layer in existing security framework.

6.2 Problem Statement

Below we identify the exact requirements that are needed to enhance the security of the routing infrastructure.

1. The need for KMPs for the routing protocols.
2. The need for structured management information that can be used for the development of the KMPs.
3. The need for a management scheme for the configuration/distribution of keys/SAs for the routing protocols.

In 2012, the KARP working group of the IETF was established to address these issues for the routing protocls. The working group analyzed the current security practices for selected routing protocols and identified the need for stronger cryptographic mechanisms (Layer 2), KMPs for routing protocols (alternative to manual method at Layer

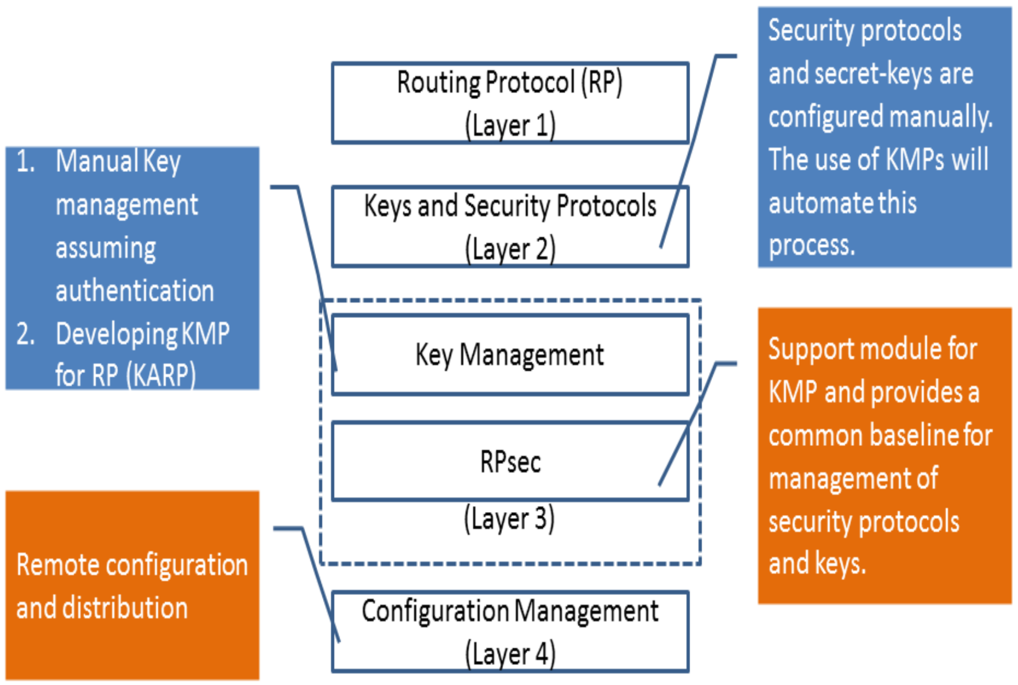
Routing Protocol (RP) (Layer 1)
Keys and Security Protocols (Layer 2)
Key Management
RPsec
(Layer 3)
Configuration Management (Layer 4)
1. Manual Key management assuming authentication
2. Developing KMP for RP (KARP)
Remote configuration and distribution
Security protocols and secret-keys are configured manually. The use of KMPs will automate this process.
Support module for KMP and provides a common baseline for management of security protocols and keys.

Figure 2: RPsec and security management framework

3) and a reusable architecture framework that can be used to address the issues related to manual methods. Some work has been completed to strengthen the cryptographic mechanisms and some work is in progress for developing the KMPs, but no work has been done in other areas.

The first requirement noted above is being addressed by various working groups within the IETF, so no further discussion will be given on this point.

The second requirement specifies the "identification and structuring of management information of the keys/SAs for the routing protocols". The idea is to identify the common information for the KMPs functionality, routing protocol security and structuring of that information for the management of the keys/SAs for the routing protocols.

We propose a *"Routing Protocol Security"* (RPsec) framework towards this requirement. The RPsec will provide the administrators with the ability to specify validation data for router authentication and multiple keys/SAs for the routing protocols. The administrators can also specify the lifetime of each of these parameters so as to avoid prolonged use of a SA in the network. The RPsec provides a way to provision all the important information required for the management of keys/SAs for the routing protocols. The KMPs are expected to use RPsec for their functionality. This will modify the Layer 3 of the current security framework, as shown in Figure 2.

The third requirement states the need for a key/SA management scheme that the operators can use to manage all the above information (authentication data, keys and SA, and associated lifetimes) for the routing protocols without expending too much efforts and big staff for it. We propose a solution for the configuration/distribution of the RPsec at Layer 4. In that, we provide yang modules that describe the common structure and types of the information managed in RPsec, and an architecture for configuration/distribution of RPsec. Overall, we have tried to cover the deficiencies at layers 3 and 4, respectively.

6.3 Management Information

In this section we list the most important security parameters that are required for a KMP's functionality and the security of the routing protocols.

6.3.1 Mandatory requirements for KMP functionality

The KMPs perform following functions—peer validation, negotiation of security protocols, and deriving/establishing traffic keys for secure communication between peers.

The RPsec provisions following security parameters that are required to support the above functionality of KMPs:

1. Information for peer validation-

- Peer identity

- Peer credentials

2. Information required for security protocol negotiation-

   -List of security protocols supported by routing protocols.

   -List of cryptographic algorithms (transforms) associated to these security protocols.

3. Information required for deriving traffic keys for secure communications-

   -Master key, which is used to derive the traffic keys.

   -Key derivation functions (KDF) used for deriving traffic keys from the master key.

6.3.2 Security parameters for routing protocols

After careful study of routing protocol security mechanisms (Section 2.6), we have identified the following security requirements in common:

1. Authentication data to validate routing peers.

2. Security protocol and associated transforms to integrity protect message exchange.

3. Keys that are used with the transforms to generate message authentication data.

4. A key derivation function (KDF) to derive traffic keys complying to the security protocol's requirement, if any.

5. A master key that may be used to derive keys or can be used directly as a traffic key.

6. Lifetime of traffic keys.

This common information is collected after a careful study of the SA requirements of the routing protocols specified in Appendix E.

Chapter 7

Proposed Framework

In previous chapters, we identified deficiencies at layers 3 and 4 in the present security framework for the routing protocols.

To overcome those problems, we proposed the RPsec framework. The RPsec is designed to overcome the deficiencies noted in the present security framework for routing protocols.

The RPsec provisions the management information (Section 6.3) in its component databases. The RPAD, the RSPD and the CKT are the component databases of the RPsec framework as shown in the Figure 3. The arrangement of RPsec databases is similar to the arrangement of PAD, SPD and SAD databases in IPsec framework as shown in Figure 3. This type of modular design supports the development and use of the KMPs.

The RSPD provisions the cryptographic security protocols for the routing protocols. It also allows the operators to configure multiple security options for the routing protocols. A KMP solution must negotiate the security protocols specified in the RSPD.

The CKT provides the master key, KDFs and other parameters that are used for securing the routing protocols. Each entry in the RSPD has a corresponding entry in

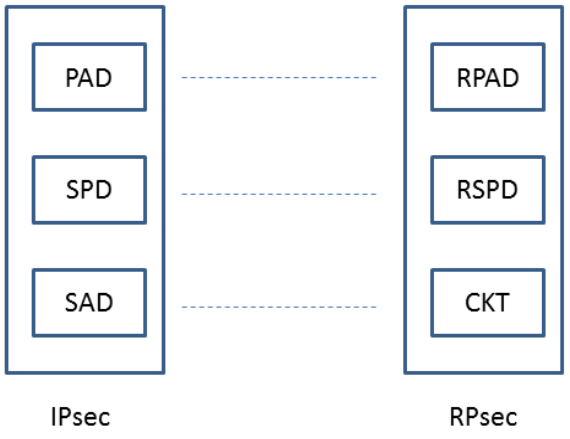

Figure 3: Similarity between IPsec and RPsec

the CKT database.

Finally, the RPAD provides information required for peer authentication and links the RSPD and a KMP solution. The use of a KMP facilitates the separation of the authentication and the authorization. In that, the router's participation in the network is authenticated using information in the RPAD while the CKT provides the session keys for authorization. Together these databases are known as RPsec databases. In addition, each database allows specifying the lifetime for data stored in it. Thus, administrators can precisely control the use of SAs among the routers.

The RPsec modifies the present security framework for the routing protocols. Figure 4 shows the way RPsec can be implemented as a management module for the key/SA management of the routing protocols at Layer 3. The layer 4 shows a yang module corresponding to each RPsec database. The yang modules are used to structurally represent the data stored in RPsec databases. The NetConf protocol can be used to remotely configure and distribute the RPsec databases to each router in the network. Then we propose a configuration/distribution architecture for RPsec in

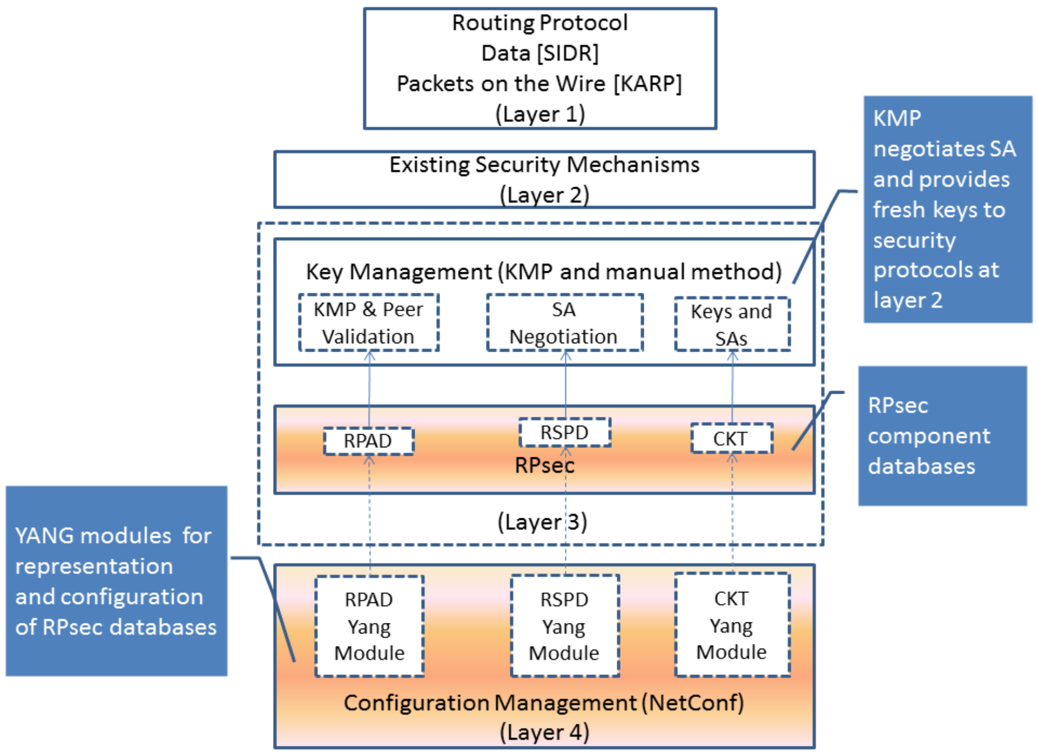
Routing Protocol
Data [SIDR]
Packets on the Wire [KARP]
(Layer 1)
Existing Security Mechanisms
(Layer 2)
KMP negotiates SA and provides fresh keys to security protocols at layer 2
Key Management (KMP and manual method)
KMP & Peer Validation
SA Negotiation
Keys and SAs
RPAD
RSPD
CKT
RPsec
RPsec component databases
(Layer 3)
YANG modules for representation and configuration of RPsec databases
RPAD Yang Module
RSPD Yang Module
CKT Yang Module
Configuration Management (NetConf)
(Layer 4)

Figure 4: Integration of routing security management framework and RPsec

Chapter 10. We also describe two cases that exemplify the use of RPsec.

7.1 Goals/Benefits

The goal of the RPsec is to provide a common base line for the management of the keys/SAs for the security of the routing protocols. Its modular design leaves the scope of adding more featured elements when required. However, such additions will be described as an update to the basic specification. It contains the list of all the essential elements that are required for the proper functioning of the KMPs and the security of the routing protocols at one place. Such an standardization of information has following advantages -

- The administrators can specify the SAs for all the routing protocols in the RPsec databases.

- It defines a standard information base for each KMP developed for a routing protocol. Each KMP will use the information available in these databases.

- It allows development of more specific solutions as per the requirement of a routing protocol. The additional requirements can be added as a separate specification to this standard.

- The yang module of each database can be modified, extended and reused. It standardizes the representation of information in the RPsec. The modules can be also be added with new features incrementally.

- Its modular design provides increased support for development and use of KMPs for the routing protocols.

- Lastly, the yang modules provide the opportunity to develop multiple configuration/distribution mechanisms using the NetConf protocol. A vendor can still develop their unique solutions using the architecture suggested for remote configuration/distribution of the RPsec databases. Such a mechanism will resolve the problems faced due to manual methods (discussed in Section 3.3).

7.2 Novelty

RPsec is a security framework only for routing protocols. It represents an effort to increase the security standards for the routing protocols. The novelty of our work is that we have provided a framework that addresses multiple problems that are faced by the present security practices of routing protocols. We provide a KMP support module at Layer 3. However, it is designed such that it can also be used without the KMPs (i.e., the RPsec parameters can be installed manually). This is the only solution that allows operators to specify the SAs for all the routing protocols in the network at one place. Then we have provided a standard representation for each RPsec database. These modules can be used with the most popular NetConf protocol for the remote configuration and distribution of SAs for routing protocols. The RPsec clearly addresses the problems faced at layers 3 and 4 as discussed in the previous sections.

Chapter 8

RPsec

In this chapter we discuss the RPsec design objectives, overview, scope of usage and the way RPsec can be integrated with the current security framework of the routing protocols.

8.1 Design Objectives

Below we present the design objectives for the RPsec.

- The RPsec is designed to separate protocol-specific aspects from both manual and automated key management. The RPsec aims to support the cryptographic protocols that are used to secure the routing protocols.
- It should be easy to specify multiple security options for a routing protocol and adaption of a new, more secure protocols should require minimum changes to the existing implementation. The idea is to provide rich security options for the routing protocols, when available. The implementors will need to develop an interface for the interaction between the security mechanisms and the RPsec such that the routing protocols can use the security mechanisms as per the

policy set in the RPsec.

- Multiple routing protocols should be able to consult these databases for their security needs via some interface. However, implementors will need to develop interfaces so that routing protocols can consult the RPsec for security. The router (or some process in the router) will provide such interface for each routing protocol that intends to use the RPsec.
- The framework should be implemented such that multiple KMPs can use it.

8.2 Caveats and Assumptions

RPsec is a framework that supports the use of the KMPs for the routing protocols. To achieve its objectives, we assume that the KMPs and the routing protocols will use this framework for security via “some interface”. Through out this text we have/will use the term interface among routing protocols, associated security mechanisms, KMPs and RPsec. The implementors will need to develop such interfaces to interact with the RPsec. We assume that such interfaces exists or will exist in future. As per [27] and [28], it is understood that the use of RPsec like solutions will require some change to the implementations of existing routing protocols. It is assumed that any new routing protocols that will be developed will use the RPsec by default. The effectiveness of the RPsec will depend on the way KMPs utilize its databases. However, in cases where KMPs are not desired, RPsec’s manual method shall be used.

The security of routing protocols depends on various factors. No single solution can provide absolute security to routing protocols. The work in [34] describes a multi-fence defense framework for routing protocols and RPsec fits in the “Cryptographic Protection Schemes” of that framework. According to it, the RPsec will provide for secure neighbor-to-neighbor communication, authentication and authorization.

In the next section, we point out how RPsec is not a KMP and the desired level of diversity that RPsec should offer in the choice of security protocols for the routing protocols.

8.2.1 Difference between RPsec and KMPs

The function of a KMP is to negotiate and establish desired SA between the routing peers. Each KMP defines its own message format and procedures to support this functionality. RPsec provides data that the KMPs will carry in their messages. For example, for initial authentication of peers the KMP will refer to the RPAD for authentication data. After the authentication, the KMP will refer to the RSPD to negotiate which security protocols to use. The successful negotiation will finally result in an SA that is stored in the CKT. The routing protocols will use this SA for security. Upon reaching the expiry of the keys/SA, some process will trigger the KMP's rekey mechanism. For example, in IKE the initiator (the local/remote IKE peer) requests IKE for rekey of existing keys/SA. The concerns regarding re-keying, renegotiation are responsibility of the KMP in use. Towards this functionality, each entry in the RPsec has a set lifetime. The lifetime information should be used to trigger the re-keying/renegotiation.

When a GKMP is used with the RPsec, a GCKS will be responsible for rekeying and renegotiation of the group keys/SAs as explained in Section 3.2. In case of using the RPsec without a KMP, the interface between the routing protocols and the RPsec and/or between the security protocol and the RPsec will keep track of the lifetimes of the keys/SAs. The interfaces will bear the responsibility of rolling over to the use of new keys/security protocols.

Therefore, RPsec provides a standard baseline for the development of KMPs for routing protocols. All the KMPs (developed specifically for the routing protocols) must use the RPsec framework for consistency.

8.2.2 Support for diverse security protocols

Figure 5 shows the diversity in the security mechanisms of the routing protocols as discussed in the Section 2.6. From the figure, we can see that the built-in security

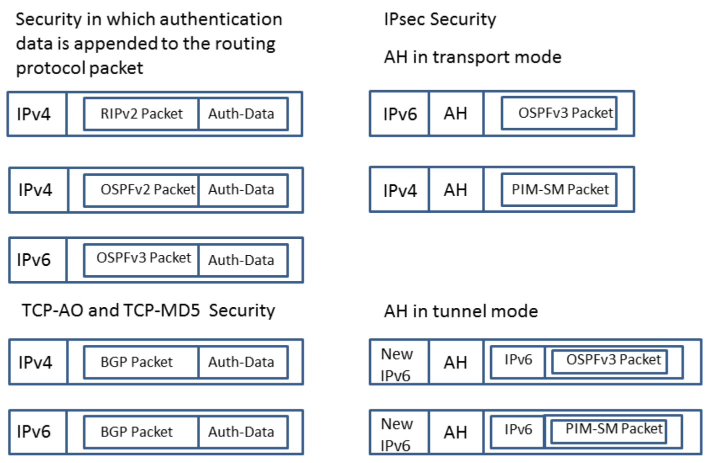
Security in which authentication data is appended to the routing protocol packet
IPv4 | RIPv2 Packet | Auth-Data
IPv4 | OSPFv2 Packet | Auth-Data
IPv6 | OSPFv3 Packet | Auth-Data
TCP-AO and TCP-MD5 Security
IPv4 | BGP Packet | Auth-Data
IPv6 | BGP Packet | Auth-Data
IPsec Security
AH in transport mode
IPv6 | AH | OSPFv3 Packet
IPv4 | AH | PIM-SM Packet
AH in tunnel mode
New IPv6 | AH | IPv6 | OSPFv3 Packet
New IPv6 | AH | IPv6 | PIM-SM Packet

Figure 5: Security protocols of RPsec

mechanisms and the TCP-AO/TCP-MD5 for the routing protocols append the authentication data to the routing protocol packets before transmission on the network. In case of IPsec security, we have two modes of operation–transport mode and tunnel mode. In transport mode the IP packet is modified to use the AH protocol. The AH protocol provides protection to both the IP header and the routing protocol packet. In the tunnel mode, the original IP packet carrying the routing protocol is encapsulated in a new IP packet. We have shown only AH protocol which is used to integrity protect the routing protocols. The same rules apply when the ESP will be used.

The RPsec is designed to support such diversity in the choice of the security protocols. It does not mandate use of any specific security protocol (whereas IPsec uses AH and ESP only). This diversity is desired because the security mechanisms vary for the routing protocols.

8.3 RPsec Overview

This section provides a high level description of the RPsec's component databases and how they will fit into the routing protocol security requirements. RPsec has three component databases—RPAD, RSPD and CKT—that provide peer authentication information, security protocol choices and key related parameters respectively, for the security of the routing protocols. The RSPD contains the security protocol options for the routing protocols. Each entry in the RPsec will point to an entry in the CKT that provides the key related parameters to use with the security protocol. The RPAD will contain the authentication data for the peer routers and also specifies an appropriate KMP for negotiation/establishment of a SA with the peer.

The RPsec will be implemented in the routing devices as a support module (in layer 3) for the key management of the routing protocols. In that, the KMPs will use RPsec for establishing an SA. In absence of a KMP, the routing protocol will consult the RPsec databases directly for the security. In this case, no negotiation would take place. In either case, the actual security protocol (as discussed in existing security mechanisms in Section 2.6) will be provided the master key and/or the derived key using an appropriate KDF (if any), via some interface. These mechanisms will then integrity protect the routing protocol packets using the keys as usual.

The RPsec also guides the processing behavior for the routing protocol traffic. To be specific, the entries in the RPsec will allow the router to decide whether to PROTECT, BYPASS or DISCARD that traffic.

The support offered by RPsec to the KMPs, the routing protocols and the security mechanisms will depend upon the quality of the design and the implementation of the interfaces for each of them as discussed in Section 8.2.

The RPsec is designed such that it could be used with/without the KMPs. We highly support the development of the KMPs that can be applied for key and SA management of the routing protocols. However, manual methods are also supported because not all deployments may require an automated key management solution.

8.4 Scope of Usage

The use of this framework is limited to the routing protocols and the associated security mechanisms only. It can also be used to provide the keys for existing security mechanisms with each having its own interface to the RPsec. For example, it may be used to populate the master key tuple used by TCP-AO to secure the BGP communication [15] via some appropriate interface. The new routing protocols may be designed to inherently use this framework for securing their messages.

8.4.1 Interface between a Routing Protocol and the RPsec

The details of how to interface the RPsec framework and a routing protocol is an implementation issue. However, the implementors should consider following things when designing such an interface.

The implementations of the routing protocols should be able to consult RPsec for security requirements. In that, the implementation should provide the routing protocol traffic description, (optionally) an associated interface to search an appropriate entry in the RSPD database.

The traffic should be dropped if the entry specifies DISCARD behavior for it. Allow the traffic to pass through if the entry specifies BYPASS. This case is equivalent to NULL authentication (as discussed in Section 2.7, OSPFv2 has a NULL authentication option), in which the traffic is simply allowed to pass through the router.

If PROTECT is specified for the traffic, the implementation should find the associated CKT entry (as discussed in previous section) for keys and related parameters. The key and the security protocol is used to protect the traffic as described in that CKT entry.

If there is no CKT entry look in the RPAD. If a KMP is specified do the negotiation. Otherwise, drop the packet.

8.4.2 Interface between RPsec and a KMP

In absence of a CKT entry, the implementation should use the peer identity, security protocol (specified in the RSPD) and (optionally) an associated interface to find an appropriate KMP in the RPAD database. If no such entry is found in the RPAD, the traffic should be discarded.

The function of the KMP is to authenticate the peer, negotiate/establish SA between peers, rekey, and to derive and establish traffic keys for secure communication between the routing protocol peers. The implementors should consider following things when designing such an interface between the RPsec and a KMP.

To authenticate the peer, the interface should be designed such that a KMP could fetch authentication data for the local and the peer router from the RPAD database. The KMP should use the authentication data to validate both the participants before starting the negotiation phase.

In negotiation phase, the KMP must be able to negotiate the most appropriate security protocol available in the RSPD for that routing protocol peer. If one or more entries are found, the KMP should be able to use the most recent entry based on its lifetime information.

Once the negotiation is successful, i.e., an appropriate CKT entry has been established, the security protocol must consult the CKT for the keys and related parameters to secure the routing protocol messages as described in Section 3 of the CKT specification [23].

The KMPs have mechanisms to track the lifetime of a traffic key and to rekey the existing key before it expires. The references in Chapter 3 and the IKE specification [25] explain how KMPs can perform all these functions in detail.

8.5 Security Policy and the RPsec

In this section, we would like to explain the relation between the term security policy and the RPsec. Each organization has one or more network security policies. As per the security policy, the administrators can specify multiple entries in RPsec databases for each routing protocol running in their network. For example, an administrator may wish to put multiple entries in the RSPD with overlapping lifetimes for key/SA rollover. One may specify multiple security protocols in the RSPD in decreasing order of preference. This may be required if they have some routers supporting old security protocols only and some supporting both—the old and the new security protocols. A KMP will negotiate an appropriate security protocol in each context as described in previous section. In case of the manual method, the administrator may put old security protocols as a first preference for the security. Therefore, when the routing protocol consults the RPsec, it uses the first available entry for security. Since each RPsec entry points to an entry in the CKT database, the administrator will have to configure a CKT entry for each security protocol likely to be used by the routing protocol. So the security policy defines the number of security options an administrator may configure for a routing protocol.

Similarly, a router may have multiple identities in the network. The administrator will provide information for each potential identity that the routing peers may use to communicate with each other in the RPAD database.

Therefore, the number of the entries in the databases, the choice of security protocols, the authentication information, keys, etc., for the routing peers will be decided by the security policy of the organization. However, an organization's security policy must specify a default DISCARD entry in the RSPD for all the routing protocols that should not be bypassed/protected. No entries must exist in RPAD and CKT databases for a routing protocol traffic that should be bypassed or discarded. All the three databases must contain appropriate entries for each routing protocol traffic that should be protected. In case of manual methods, only the RSPD and/or the CKT will contain appropriate entries. When we mention the availability of a security policy or just a security policy for a routing protocol traffic, we mean that there is at least one entry (including the default entry) that specifies the processing behavior for that routing protocol traffic.

8.6 A Formal Introduction to RPsec databases

In this section, we formally introduce the RPsec databases.

8.6.1 RSPD

The objective of the RSPD is to provide security options (choice of security protocol) for a routing protocol's security. Each entry (a choice) specify the security parameters required to establish a SA between the peers. An authorized device may communicate with many routing protocol peers. To do so, it must agree on the security requirements of the routing protocol peer for successful communication. The peers must agree on security protocols, transforms, mode of communication along with the key required to integrity protect messages exchanged between them. This database aims to provide such information. The RSPD contains the traffic descriptors for identifying each routing protocol traffic that needs to be protected, bypassed or discard. The RSPD, thus, is a database to specify the traffic descriptors for the routing protocol traffic, security protocols, lifetime and related parameters for securing the communication between the two devices or among a group in case of the multicast communication. This database provides partial information towards security requirements of the routing protocols. The rest of the information is provided by the CKT. Chapter 9 explains the RSPD database in detail.

8.6.2 CKT

The CKT, as discussed in Section 4.2, is an important database that provisions key material and associated cryptographic algorithms to protect the routing protocol messages. In RPsec, the CKT performs the role similar to the SAD in IPsec. It stores the negotiated (or manually configured) SAs for the routing protocols. In that, each RSPD entry points to an appropriate entry in the CKT. Each RSPD entry that protects the routing protocol traffic, provides a (security) protocol id and a peer id (traffic descriptor) that identify an entry in this database. The form of the protocol id and the peer id is specified in [23]. The RSPD together with CKT ensure that the key is provided to a security protocol that is used for securing the routing protocol.

8.6.3 RPAD

The RPAD's objective is to provide authentication information and a KMP for the routing peers. It provides authentication information necessary to assert a local device's identity and to validate the identity asserted by the peer devices. A KMP uses the information in the RPAD and the RSPD for authentication and SA negotiation, respectively. Authentication is required to ensure that the devices participating in the network infrastructure are legitimate. A legitimate device should present its identity, identity of remote peer(s) or group it wishes to communicate with, and an organization-wide acceptable credentials. If the device successfully passes the peer device's scrutiny, it is authenticated to communicate with the requested peer(s) or a group in the network. The communication between the two devices must stop if the KMP fails to authenticate the peers using the information available in the RPAD database. A KMP negotiates SA only after the authentication is successful. Chapter 9 explains the RPAD database and its contents.

8.7 Using the RPsec

We identified the way RPsec fits in the security framework for routing protocols in Chapter 7. Now let us understand how RPsec can be used with/without a KMP.

8.7.1 Using RPsec with a KMP

Figure 6 shows the similarity between the IPsec-IKE interaction and the RPsec-KMP interaction. In that, a KMP uses the RPAD database for the authentication parameters required for peer validation. Next, the RSPD provides the list of security protocols for the negotiation/establishment of the desired SA. The negotiated SA points to an appropriate entry stored in the CKT. The routing protocol packet is then protected using the cryptographic protocol and the key specified in that CKT entry. As discussed in the previous sections, the routing protocol checks for the entries that match its packet description in the RSPD. If the database has no entry for it, the packet must be discarded as per the default discard entry in the RSPD. Allow the packet to pass through if an entry specifies the bypass behavior for it.

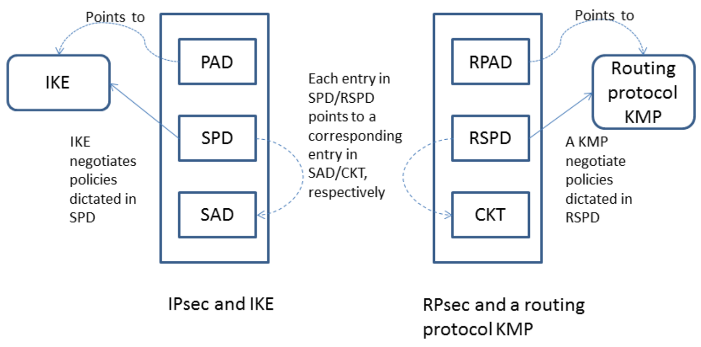
Points to
IKE
PAD
SPD
SAD
IKE negotiates policies dictated in SPD
Each entry in SPD/RSPD points to a corresponding entry in SAD/CKT, respectively
RPAD
RSPD
CKT
Points to
Routing protocol KMP
A KMP negotiate policies dictated in RSPD
IPsec and IKE
RPsec and a routing protocol KMP

Figure 6: IPsec and RPsec interacting with KMP

If a RSPD entry specifies to protect that traffic, the implementation must look for a corresponding entry in the CKT. If no entry is found, the routing protocol refers to the RPAD for a KMP invocation. If an entry is found it must specify the use of a KMP. The KMP performs mutual authentication using credentials from the RPAD and establishes a SA with the peer router using the security parameters from the RSPD entry. In case the KMP fails to negotiate a SA, the packet must be discarded. The packet must also be discarded if the RPAD does not contain any authentication information for the destination of the packet (a peer router). Finally, the negotiated SA points to an entry in the CKT database that is used to protect the routing protocol communication. The security mechanism is provided with the key and related parameters from this CKT entry via an interface. The work in [15] specifies how an interface from CKT to security mechanisms for routing protocols can be made. Figure 7 shows a flow chart that describes the process flow when a KMP is used.

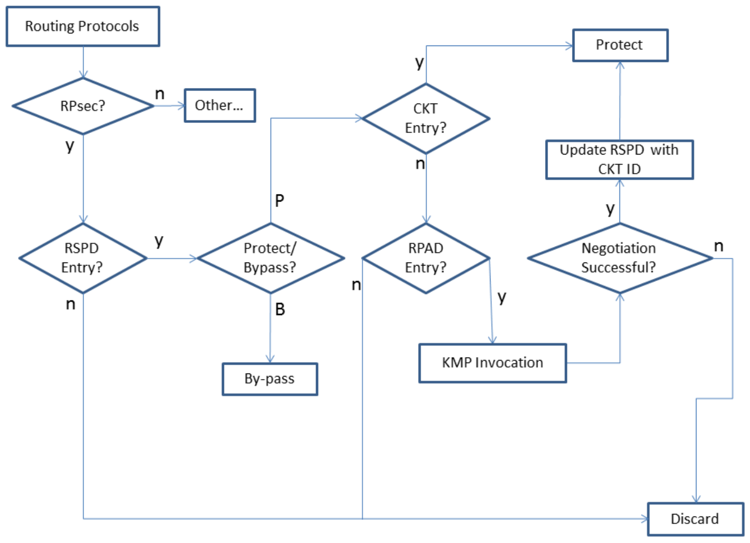

Figure 7: Using RPsec with a KMP mechanism

8.7.2 Using RPsec with Manual Method

The RPsec also supports manual method. The routing protocol implementation consults the RSPD for the security. There is no role of RPAD as SAs are not negotiated in manual method. Instead, the choice of a security protocol is defined by the administrator as per the security policy of the organization. The routing protocol uses the first available entry that matches its packet description. The routing protocol traffic is discarded if no entry exists for it as per the default discard entry. The traffic is allowed to pass through if a bypass entry exists for it. Finally, if a RSPD entry that specifies the protect behavior for it exists, the routing protocol uses the keys from the CKT entry pointed to by this RSPD entry. Thus, the packet is protected using the key and the cryptographic protocol specified in the CKT entry. The keying material to the security mechanisms is provided in the same way as discussed in the previous section. It should be noted that, in case of the manual method, the information in the RPsec databases should be consistent at the communicating peers. Figure 8 shows a flow chart that describes the process flow when the manual method is used. For

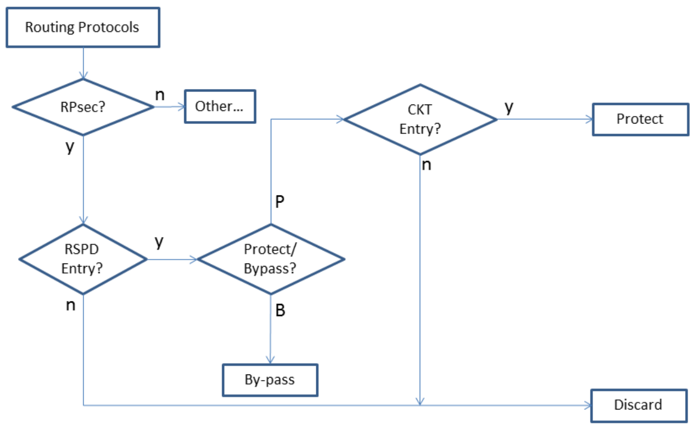

Figure 8: Using RPsec with manual method

key/SA change or rollover, some process in the router should be able to keep track of the lifetime of the key/SA. This process should trigger the key change before the current key/SA expires. Since, the security information will be consistent for a routing protocol in the network, all the routers will rollover/change the key/SA before it expires. Thus, the change occurs simultaneously in all the routers in the network. This is one of the major problem that is faced by the operators using present manual methods (as discussed in Section 3.3).

Chapter 9

RPsec in Detail

In this chapter, we discuss in detail the information that is included in the RPsec databases—RSPD and RPAD. These conceptual databases are designed to cooperate with the CKT to provide an overall security framework for the routing protocols.

Appendix E specifies the SA parameters for the existing routing protocol security mechanisms as discussed in the Section 2.6. This chapter updates and re-groups those SA parameters for developing a common framework.

9.1 RPsec Databases

As with the CKT, the RSPD and the RPAD databases are designed to separate protocol-specific aspects from both the manual and the automated key management methods. The information in these databases describe the security services for the routing protocols. The aim is to allow different implementation approaches to RPsec while simplifying the specification of the security services and the way those services can be deployed in the routing protocol environment. The implementations should conform to the characteristics of the RPsec that is communicated by these databases. The information included in the RPsec does not specify a system configuration, instead, it resides over the system configuration as the management of security information for the KMPs and the routing protocols. The way the RPsec is deployed will depend on an organization's security policy as discussed in the previous chapter.

We use the following conventions for the default-values in each field of the RPsec databases. Any RPsec database field may contain some valid data, ANY or OPAQUE. Each field defines the form and type of the valid data. However, the use of ANY or OPAQUE is as follows. ANY is used as a wildcard that matches any value in the corresponding field of the packet. Use of OPAQUE means, the value corresponding to that field does not exist or is not required for processing the packet. Such use is well tested in IPsec [26] and we will use these values in our RPsec databases as well, when appropriate.

9.1.1 RSPD-Routing Security Policy Database

The RSPD is an important database in the RPsec. It allows a security administrator to specify the desired security level for the routing protocols.

The RSPD is consulted at all the times for all the traffic generated by/received for the routing protocols. Routing protocol traffic should be discarded if a specific entry or a global entry specifies DISCARD processing behavior for it. The traffic should be allowed to BYPASS if an entry specifies so in the RSPD. The traffic is protected if an entry defines the PROTECT behavior and contains all the security parameters for it. The actual protection is provided by the security mechanisms as discussed in the previous chapter.

The RSPD allows the security administrators to specify multiple entries for routing protocol traffic in the order of most preferred entry first. An entry is selected based on the longest match that fits the packet description. The traffic must match at least one of these entries for the RPsec to protect it.

There are at least two entries for a unicast traffic protected by the RPsec, one for the outbound traffic to the peer and one for the inbound traffic from the peer. This is done by swapping the source and the destination addresses and the ports in the RSPD entry. But this cannot be done in case of a multicast communication as a multicast address can never appear in the source address field of an IP packet. Later in this chapter we will discuss the RPsec for multicast communications and explain how the directionality is specified for a multicast traffic. The role of the RSPD database–

- Identification of the routing protocol traffic of interest.
- Specify the processing behavior for the identified routing protocol traffic.
- Specify one or more available security options for the routing protocol in user-ordered preference.
- Allow existence of the multiple entries to promote regular key/SA rollover.

RSPD Contents

A RSPD entry contains traffic selectors, security parameters, lifetime, and one or more associated interfaces. In case of partial entries, one or more fields may contain "ANY" or "OPAQUE" value. The fields in the RSPD are as follows-

Traffic Selectors

A set of values used to define the routing protocol traffic that should be protected by the RPsec. A set of traffic selectors may identify a single or multiple routing protocols depending on the values of the parameters. Each parameter in the set may contain a specific value, a range of values, ANY or OPAQUE. The traffic that exactly matches or falls under the range specified by the traffic selectors is protected by the RPsec.

The ANY value literally matches any value in the corresponding field for the traffic that crosses the router. The OPAQUE is set to convey that, either the information corresponding to those fields does not exist or is not applicable to the traffic in the context. For example, the IPv4 and the IPv6 packet formats are different and to reflect that the database may contain OPAQUE values wherever necessary. Also, if a next layer protocol does not use the IANA port numbers then OPAQUE must be set against the corresponding fields in the database. The following traffic selectors are defined for identifying the traffic:

Source IP Address

This is an IP address assigned to the local router's interface.

Destination IP Address

It refers to an IP address assigned to the peer router's interface. The peer may lie on the same network as the source router.

Next Layer Protocol (NLP)

The value in this field corresponds to the IPv4 packet's "protocol" field or the "next header" field found in an IPv6 packet. This field may contain an individual protocol number or ANY. The NLP in an IPv6 packet is anything that comes after the extension headers.

The values in the next two fields depend upon the protocol specified in the NLP field. For example, if the protocol is TCP(6) or UDP(17), the next two fields will contain the port numbers required to identify the connection. If the NLP is for example, an OSPF protocol (89), then the source and destination port fields will contain ANY or OPAQUE.

Source Port

If the NLP refers to a protocol that uses local port number(s) then this field contains the supported port numbers or the range of port numbers corresponding to the NLP. The value OPAQUE specifies—this field is not required. The value ANY means any IANA defined number is allowed in this field.

Destination Port

If the NLP refers to a protocol that uses specific remote port number(s) then this field contains the supported port number or the range of port numbers corresponding to the NLP. The value OPAQUE specifies—this field is not required. The value ANY means any IANA defined number is allowed in this field.

Both the source and the destination ports are defined as a range. The administrator must define the lower and the upper values for these fields. For example, a range starting with 1025 (lower-value) to 5000 (upper-value). The administrators can define a trivial range for specifying a unique port. For example, lower-value=1024 and upper-value=1024.

Processing Behavior

The RSPD allows an administrator to define the processing behavior for each routing protocol traffic passing through the router. As discussed earlier, the value of this field determines whether to protect, bypass or discard the routing protocol traffic.

Processing behavior

This field can contain any one of these values—PROTECT, BYPASS or DISCARD. PROTECT specifies that the identified routing protocol traffic should be protected. The router can further refer to RSPD, RPAD and CKT for security parameters as discussed earlier. BYPASS allows the traffic to pass through (no further lookups in RPAD/CKT). The last choice specifies that the identified traffic is not allowed to pass through the router and must be dropped.

Security Parameters

The security parameters specify the security protocols for the routing protocol. It contains only those cryptographic protocols that are supported by the routing protocols identified by the traffic selectors. Multiple RSPD entries may be specified for a routing protocol in the order of preference.

Following security parameters are specified in the RSPD entry:

ProtocolID

It identifies a single security protocol that is supported by a routing protocol. If a routing protocol supports multiple security protocols, the administrator may configure a RSPD entry for each in the order of preference. A KMP solution will negotiate the security protocol in this order. The list of present supported security protocols is given in Section F.1, Appendix F.

Mode

This field identifies the mode of operation of the security protocol. For example, AH and ESP support two modes of operations—transport mode and tunnel mode.

TransformType

It specifies which feature of the security protocol (corresponding to the ProtocolID) to use, integrity(INTEG) or encryption(ENCR). It is unlikely to use the ENCR type security for the routing protocols, however, this field will be used to specify which feature of the security protocol to use for securing the routing protocol. For example, ESP protocol's INTEG feature may be used for protecting the OSPFv3 routing protocol.

TransformID

A security protocol may have one or more transforms that it uses to integrity protect or transform (encrypt and decrypt) the routing protocol messages. Section F.2, Appendix F contains the possible transforms ids that may be supported by RPsec.

Lifetime Parameters

The lifetime parameters define the time period for which the corresponding RSPD entry is valid for use. Just like security parameters, multiple RSPD entries may exist for a routing protocol with slightly overlapping time periods. Overlapping time periods allow for the key/SA rollover without dropping the ongoing communication between the routing protocols. Eventually, the old entry must expire for a new SA to take over completely. The format of the lifetime parameters in the RSPD is the same as specified in the CKT specification [23]. The negotiated SAs established will have the same age (or less) as the age of this RSPD entry.

SendLifeTimeStart

It specifies the earliest time in coordinated universal time (UTC) at which this entry should be considered for use when sending the traffic.

AcceptLifeTimeStart

It specifies the earliest date at which the traffic protected by the security protocol specified in this entry should be processed by the communicating entities.

SendLifeTimeEnd

It specifies the latest time at which this entry should be stopped for use.

AcceptLifeTimeEnd

It specifies the latest time at which further processing of the traffic protected by security protocols in this entry should stop.

Interfaces

One or more interfaces of the router can be specified on which to use the security protocol dictated by this RSPD entry. The question of whether the interface field is required to properly apply a security protocol is specified by the routing protocol itself. For example, the security of OSPF protocol is interface based, i.e., all the routers on the same network as the OSPF interface use same SA. It should be noted that the SAs are usually tied to the interfaces. The combination of the key identifier and the interface associated with the message uniquely identify the security protocol and associated keys to use as specified in [7] and [33].

InterfaceID

This field identifies the physical or virtual interfaces of the router on which to use the security protocol. The value in this field is set to "ALL" if the security protocol is to be used on all the interfaces of this router. Also, the security type for a routing protocol can be host based or interface based. In host based, the security protocols can be specified per host. In interface based, all the peers that connect to that interface share the same security protocol. The format of this field conforms to the CKT specification [23].

9.1.2 RPAD-Routing Peer Authorization Database

The RPAD provides a link between the RSPD and a KMP. It provides the identities of the peers or identity of a group of peers that will communicate with this router. It specifies a KMP and authentication data to be used for the peer authentication. The order of the RPAD entries is defined by the administrator. The entry contains only those identities that a peer can assert to this router for identification.

In the next section we will see the contents of the RPAD entry. The RPAD helps in performing following critical functions–

- Provides a link between the RSPD and a KMP solution.
- Identifies the peer or a group of peers that are authorized to communicate with this router.
- Specifies the KMP protocol and method used to authenticate each peer.
- Provisions authorization-credentials for the routing peers.

RPAD Contents

A RPAD entry contains the identity of a peer or a group of peers, specification of a KMP, associated credentials, lifetime of the entry and the associated interfaces as follows:

Identification Information

It contains the identity of each peer or a group of peers that may communicate with this router. Each entry has a local identity that represents this router and a remote identity of the peer router. There is no limitation on the number of identities that a router can possess in the network.

LocalID

This field contains the identity of the local router that will be used to communicate with the peer specified in the PeerID field. A router may possess multiple identities in which case the administrator will have to put multiple RPAD entries for each identity. This field can contain all the ID types that are supported by the IKE protocol [25].

PeerID

This field contains the identification information of the peer router. This field may contain only one value per entry. Since a peer may possess multiple identities, the administrator will have to list all the identities of the peer router that the local device is expected to communicate with. This field and LocalID field can contain all the ID types that are supported by the IKE protocol [25]. The Section F.4 in Appendix F lists the possible identities that may be supported by the RPsec.

Authentication Information

Authentication information indicates whether a KMP is used for authenticating the peer router. If a KMP is used, it further specify the type of the KMP, the identity of the KMP protocol and the authentication method. Authentication information consists of the following fields–

KMPType

It specifies the type of the KMP to use for peer authentication—UNICAST OR MULTICAST.

KMPID

This field specifies the identity of the KMP (of type KMPType) to be used for key and SA negotiation.

Auth-Type

This field indicate the type of authentication to be performed. Presently two types of authentication are available—pre-shared secret and X.509 certificate. The encoding of these types of authentication data is described in the Section F.5.

Credentials

It is a logical container that stores the authentication data and the related information in the RPAD depending upon the value of Auth-Type field.

AuthData

If the Auth-Type is a pre-shared key, this field contains a shared-secret. If the Auth-Type value is X.509 certificate, then this field contains the certificate provided by the administrator. The encoding of the AuthData depends on the IKE supported encoding types defined in Section F.5.

CertificateAuthority (CA)

If the certificate authentication is used, this field may contain an indicator of the trusted authorities, OCSP server or a policy server that is used to validate the AuthData. The value in this field may be OPAQUE if a pre-shared secret is used. The value in this field follows the rule specified in the IPsec's PAD database [26] and IKE [25].

Lifetime Information

The administrators may want to configure the lifetime for each entry in the RPAD database. The lifetime of the RPAD entry may be the same as the lifetime of the pre-shared key or the certificate that is used for authentication. The information can be simply be stated in terms of hours, minutes, seconds, days or months.

Interfaces

This field is same as the InterfaceID in the RSPD. It is used to identify the interfaces on which to use the KMP specified by the KMPID.

With all this information, the RPAD provides security parameters to protect against the spoofing attack because a KMP consults the RPAD database for peer validation. The KMP may re-authenticate the peer during rekey/renegotiation phase, if desired. The administrator must provision all the peer information in the RPAD for a secured communication.

9.1.3 Security association database for routing protocols

The CKT performs the role of the security association database in the RPsec framework. As discussed in the previous chapters, the CKT is used to store SAs for the routing protocols. If a KMP is used, the negotiated SA is put into the CKT. An administrator can also manually configure the SA at each peer involved in the communication. The CKT fits the requirements of a SAD for this framework because it is developed exclusively for key management of routing protocols. The CKT entries are effectively used for identifying the master key, list of peers that share this key, the cryptographic algorithm, associated KDFs and the lifetime information. A CKT entry is identified based on a (security) protocol identity and a peer identity provided by the routing protocol. It also standardizes the way this information is represented in the key table. The information of the fields in the CKT is specified in [23].

9.2 Multicast RPsec

Multicast RSPD

In the unicast communications, two RSPD entries can be specified. For example, one entry for the outbound traffic from “source address A:source port Ap” to a “destination address B:destination Port Bp”. The second entry for the inbound traffic from “source address B:source port Bp” to “destination address A:destination port Ap”. Thus, the directionality is represented by swapping the source-destination addresses and ports (if applicable).

This way of representing the directionality is not possible in multicast communications as the destination address (a multicast address) can never appear in the source address field of an IP packet. To represent the directionality for multicast communications a new field “Direction” is added to the RSPD. This field may contain following values—SENDER-ONLY, RECEIVER-ONLY or SYMMETRIC—as suggested in [42]. Thus, the RSPD extended by adding the direction field becomes the multicast RSPD (mRSPD). The mRSPD will support RSPD entries for both, the unicast and the multicast communications. The fields which are not required for the unicast communications will be set to OPAQUE.

Multicast RPAD

As discussed in the available GKMP standards (Section 3.2), each GM communicates with the GCKS of the group it wishes to join. The knowledge of the GCKS is important because it provides the group SA that will be used among the GMs for a secure communication. Thus, each GM must know the identity of the GCKS as well as the identity of the group it wishes to join. The former can be specified in the RPAD identity field. However, the latter introduces a new field—GroupID—in the RPAD. This field stores the identity of the multicast group. The GM should specify the identity of the multicast group it wishes to join. The GCKS then checks the validity of the GM's request and responds only if the request is validated. Thus, for a multicast communication, following data fields are required:

- GCKSID: It does not require a separate field. This information is stored in the peer-id field in the RPAD. Each multicast group may have its own GCKS. It is also possible that a GCKS manages more than one multicast group.
- GroupID: This adds a new field to the RPAD. It represents the identity of a multicast group that a GM wishes to join. The GroupID along with the GCKSID will be used to authenticate the GCKS. This information will also show a GM's intent to join a particular multicast group controlled by the GCKS. We will use the IANA assigned addresses to refer to the routing protocol multicast groups. For example, 224.0.0.5 refers to the ALL_OSPF_ROUTERS.

This extended RPAD is called the multicast RPAD (mRPAD). The mRPAD will support authentication data entries for both, the unicast and the multicast communications. The GroupID field which is not required for the unicast communication will be set to OPAQUE.

Chapter 10

Representation and Distribution of RPsec Policies

This chapter explains the YANG modules for each RPsec database. Section 9.5 describes a possible way of configuring RPsec databases in the network in compliance with the IETF's policy-based network management (PBMN) and distributed management architecture as described in [14]. Finally, we present two examples for using the RPsec in Section 10.6.

For management of the contents of the RPsec databases, we have organized and defined the data fields of the RPsec databases in following four modules–

- RPsec common types module: We have grouped all the common and reusable RPsec specific data elements and data types in this module for consistency. It acts like a data type library for the other three modules. The other three modules import RPsec common types module. The other standard modules that these modules import are the YANG common types (RFC6020) and the IETF internet types (RFC6991). The Yang model for it is described in Appendix A.

- RPAD module: It contains all the data elements described in Section 9.1.2.

Section 10.2 represents the tree diagram of the RPAD YANG model. The complete YANG data module for the RPAD is described in Appendix B.

- RSPD module: It contains all the data elements described in Section 9.1.1. Section 10.3 represents the tree diagram of RSPD data model. The complete YANG model for the RSPD is descibed in Appendix C.

- CKT module: It contains all the data elements described in RFC7210 [23]. Section 10.4 represents the tree diagram of CKT. The complete YANG model for the CKT is described in Appendix D.

We have represented only the tree diagram in this chapter. For actual modules, please refer to appendix corresponding to each module. The modules are self-explanatory in that, each data type, node, leaf, leaf-lists contains a description for themselves. However, the descriptions are the same as explained in Chapter 8.

The tree representation for RPsec common types module cannot be represented because it does not contain any real node.

The tree diagrams for YANG modules are generated using the PYANG tool. This tool is first used to compile the original YANG modules. Once the compilation is successful, one can generate the tree representation of each module. We have compiled and validated the above mentioned YANG modules for compliance with YANG syntax and semantics, IETF standard representation and proper referencing of imported modules. Then, we generated the XSD for each module. These XSDs are used to generate equivalent XML documents for the RPsec databases. The sample XML documents are shown with examples later in this chapter.

10.1 Terminology for Yang Tree Diagrams

In the next section, we discuss the tree diagram for each module. A YANG tree diagram represents the structural arrangement of the nodes in the YANG model. It provides a concise representation of the YANG module. It also represents the logical containers that contain actual nodes and leaf nodes. One can identify the data type of each leaf node in the tree.

The following conventions are used for representing tree diagrams. Each node is printed with following information–status, flags, name, opts, type and if-features.

- Status, where status is one of *+*, *x* or *o*. + is for current, x for deprecated and o for obsolete.
- Flags, where flag is one of *rw*, *ro*, *-x*, *-n*. rw for the configuration data, ro for the non-configuration data,-x for the RPCs and -n for the notifications.
- Name is name of the node. Name between () brackets means that the node is a choice node. Name between ":[name]" is a case node. If the node is augmented into the tree from another module, its name is printed as prefix-of-imported-module:name.
- Opts means options. ? for an optional leaf or choice. ! for a presence container, * for a leaf-list or list and [Keys] for a list's keys. List keys are used to uniquely identify list items.
- Type, where type is the name of the type for leafs and leaf-lists.
- if-features, where if-features is the list of features this node depends on, printed within curly brackets and a question mark "...?".

We have numbered each line in the tree diagram for the ease of referencing and explanation.

10.2 YANG Model for RPAD

The tree diagram below shows the logical arrangement of the contents of the RPAD database.

```
1. module: rpad
2.    +--rw rpsec-rpad
3.       +--rw rpad-entry* [local-id]
4.          +--rw local-id              rpsectype:device-id
5.          +--rw credential
6.          |  +--rw encoding    rpad:cert-encoding
7.          |  +--rw authdata    rpad:cert-data
8.          |  +--rw CA?         rpad:cert-authority
9.          +--rw auth-type             rpad:authtype
10.         +--rw rpad-peers* [peer-id]
11.         |  +--rw peer-id          rpsectype:device-id
12.         |  +--rw group-id?        union {mrpad}?
13.         |  +--rw credential
14.         |  |  +--rw encoding    rpad:cert-encoding
15.         |  |  +--rw authdata    rpad:cert-data
16.         |  |  +--rw CA?         rpad:cert-authority
17.         |  +--rw key-management
18.         |     +--rw KMPtype    KMP-type
19.         |     +--rw KMPID      yang:yang-identifier
20.         +--rw interface         rpsectype:interface-id
21.         +--rw sendlifetimestart      yang:date-and-time
22.         +--rw acceptlifetimestart    yang:date-and-time
23.         +--rw sendlifetimeend        yang:date-and-time
24.         +--rw acceptlifetimeend      yang:date-and-time
```

The line 1 represents the original module name. Lines 2, 5, 13 and 17 represent logical containers. The logical containers are used for logically grouping the data items. Line 2 shows a logical container that contains all the RPAD entries. Lines 5 and 13 logically group the credential information. Line 17 groups key-management information. Line 3 represents a list of RPAD entries. Line 10 represents a list of peers. All other lines are leaf nodes.

10.3 YANG Model for RSPD

This section explains the logical structuring of the contents of RSPD database.

```
1. module: rspd
2.    +--rw rpsec-rspd
3.       +--rw rpsec-entry* [src-ip-address dest-ip-address NLP]
4.          +--rw src-ip-address         inet:ipv4-address-no-zone
5.          +--rw dest-ip-address        inet:ipv4-address-no-zone
6.          +--rw src-ipv6-add?          inet:ipv6-address-no-zone {ipv6}?
7.          +--rw dest-ipv6-add?         inet:ipv6-address-no-zone {ipv6}?
8.          +--rw source-port
9.          |  +--rw lower-port    union
10.         |  +--rw upper-port    union
11.         +--rw destination-port
12.         |  +--rw lower-port    union
13.         |  +--rw upper-port    union
14.         +--rw next-layer-protocol    uint16
15.         +--rw processing-behavior    enumeration
16.         +--rw interface              rpsectype:interface-id
17.         +--rw direction?             rpsectype:direction {mrspd}?
18.         +--rw security-protocol* [protocol-id]
19.         |  +--rw protocol-id       yang:yang-identifier
20.         |  +--rw mode?             union
21.         |  +--rw transform-type    enumeration
22.         |  +--rw transform-id      yang:yang-identifier
23.         +--rw sendlifetimestart      yang:date-and-time
24.         +--rw acceptlifetimestart    yang:date-and-time
25.         +--rw sendlifetimeend        yang:date-and-time
26.         +--rw acceptlifetimeend      yang:date-and-time
```

Please note that the NLP is short for the next-layer-protocol (line 14). Line 1 represents the RSPD module. Line 2 represents a logical RSPD entry container. Line 3 is a list of RSPD entries and line 18 represents a list of the security protocols. Lines 6, 7, 17 and 20 are optional nodes. Lines 8 and 11 logically group the source and destination ports information. All other lines represent leaf nodes.

10.4 YANG Model for CKT

This section explains the logical arrangement of the contents of the CKT database.

```
1. module: ckt
2.    +--rw rpsec-ckt
3.       +--rw ckt-entry* [Protocol]
4.          +--rw adminkeyname?             yang:yang-identifier
5.          +--rw localkeyname             yang:yang-identifier
6.          +--rw peerkeyname              yang:hex-string
7.          +--rw peers* [peer-id]
8.          |  +--rw peer-id    inet:ip-address-no-zone
9.          +--rw Protocol                  yang:yang-identifier
10.         +--rw mode?                     string
11.         +--rw protocol-specific-info?   string
12.         +--rw interface                rpsectype:interface-id
13.         +--rw KDF?                      union
14.         +--rw ALGID                    yang:yang-identifier
15.         +--rw Key                      yang:hex-string
16.         +--rw direction                rpsectype:direction
17.         +--rw sendlifetimestart         yang:date-and-time
18.         +--rw acceptlifetimestart       yang:date-and-time
19.         +--rw sendlifetimeend           yang:date-and-time
20.         +--rw acceptlifetimeend         yang:date-and-time
```

Line 1 is the name of the CKT module. Line 2 represents a logical container for routing protocol SAs. Line 3 represents a list of CKT entries (SAs). Line 7 represents a list of peers. It contains peer-id, protocol (security protocol) to use, interface on which it lies with respect to the local router, master key to be used with the security protocol and, the lines 4,11 and 13 represent optional nodes. The rest all are the leaf nodes.

10.5 Distribution of Security Policy

This section explains a policy management architecture for distribution of RPsec contents. This architecture follows the IETF's standard for policy based networking defined in [43].

The two main architectural elements for policy control are a PEP (Policy Enforment Point), preferably a router, and a PDP (Policy Decision Point) which is likely to be a remote policy server (PS) [43]. The PEP is a RPsec policy aware network node and the PDP is a remote policy server that uses external repository or other servers for fetching the policy information for the PEP. The PDP serves the requests from the PEP and have the authority to force push/update the policies to it. Figure 9 shows the architectural elements for the RPsec policy distribution. The router and the policy server are NETCONF aware network nodes, i.e., the PS acts as a NETCONF server while the router is a NETCONF client.

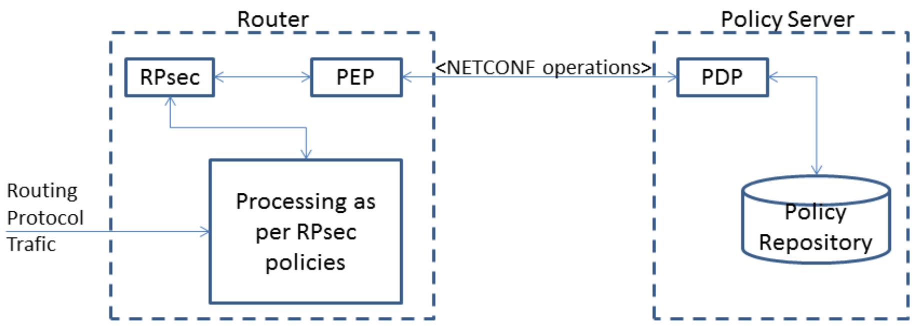

Figure 9: Policy distribution architecture for RPsec

The PEP and the PDP interact in the following manner- Since the router (a PEP) is RPsec aware, the router will send policy request(s) to policy server (PS), a PDP. By RPsec aware we mean that the device implements RPsec framework as explained in previous chapters. The requests from the router may include the identity information, the interface addresses and all the routing protocols that are configured to run on it. This information will be used by the PDP to find appropriate RPsec policies for each routing protocol mentioned in the request message. PS will only reply with the policies for the routing protocols running on the router. The PS may serve individual requests received from each router in a unicast communication. For group communications, the PS should be able to push the shared policies to all the member routers of that group. The decision of when to push the group policies is an implementation detail. For example, such a push will be initiated when one of the router in the group requested for the group security policies. This will enable the PDP to look for the list of group members from some external repository (like LDAP) and push the policies to each identified member. Another way could be to wait for the requests from each group member. Thus, each member of the group is provided with the shared policy on explicit requests only.

For a distributed policy management, this architecture can be enhanced to a distributed architecture by leveraging the work done in distributed policy-based network management with NETCONF [14]. In that, each node may act as both, the PEP and the PDP. The PEP module in router will receive notifications/configuration information for itself and the other routers it manages, from a PDP higher up in the heirarchy. The local PDP (LPDP) module will act as a distribution point for the routers lower in the heirarchy.

Figure 10 shows a distributed policy distribution architecture for RPsec policies. It depicts a distributed architecture where an intermediary node is both a PEP and a PDP. The PEP module of the router receives the policies for itself and the nodes it manages from the policy manager. The LPDP on the other hand deals with the policy requests from the managed routers. The PEP and the LPDP modules may be

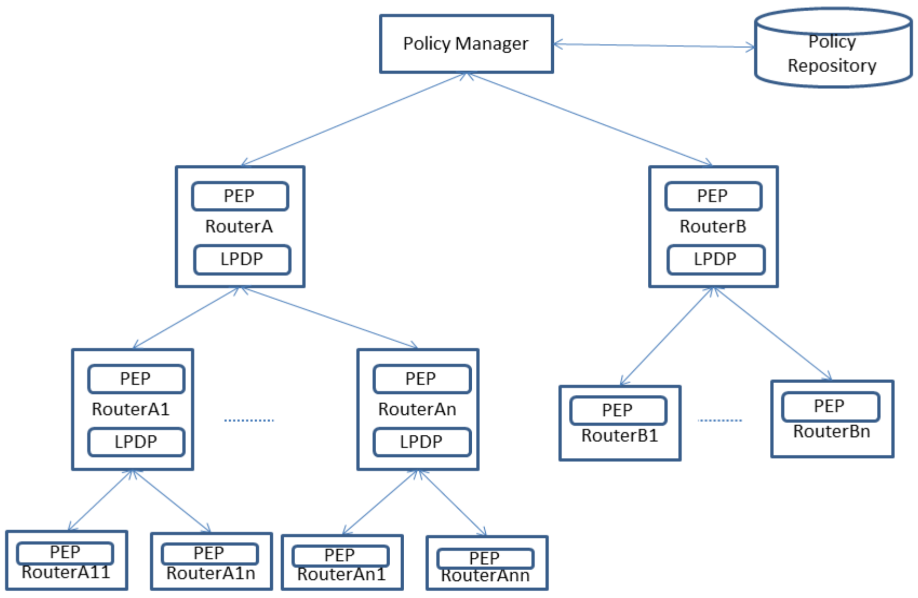

Figure 10: Distributed policy distribution architecture for RPsec

implemented following the guidelines in [14].

These architectures must be implemented following the IETF guidelines for building network management applications using NETCONF and YANG [37].

10.6 Examples

Following conventions are used for the examples below:

- Administrators should use appropriate KMPs (unicast or multicast) when available. For now, we assume that some KMP exists for each routing protocol mentioned in the examples.
- We have used security protocols, transforms, certificate encodings from Appendix F.

- In case of a unicast communication, we have shown only one-entry, i.e., from a sender to a receiver. For an incoming traffic, same entry exists but the source address is swapped with destination address and source ports are swapped with destination ports. For that reason, we have set the direction to "SYMMETRIC".

- For multicast communications, the GCKS (corresponding to the GCKS-ID in RPAD) provides key/SA information to the router. It is because SAs cannot be negotiated for the multicast communications.

- For outgoing multicast traffic (from a source to a multicast address), the direction is set to "SENDER-ONLY". For incoming multicast traffic the direction is set to "RECEIVER-ONLY". In absence of a peer information, the administrators may define a single entry for all the incoming traffic to a multicast address. It can be done by setting the source address to "ANY", the destination address to the multicast address of the routing protocol and by setting the direction to "RECEIVER-ONLY".

- We refer to a multicast group by its multicast address. For example, in case of the OSPFv2, ALL_OSPF_ROUTERS and ALL_DESIGNATED_ROUTERS groups are identified by 224.0.0.5 and 224.0.0.6, respectively.

- We set a field's value to "OPAQUE" if it is not required.

- Lastly, each example shows RPsec databases contents from a single router's perspective. Similar configurations would be required on its neighbors as well.

We have used the PYANG tool for generating the XSD corresponding to each YANG module presented in this work. The XML documents are generated using these XSDs. In Section 2.7, we learned that the OSPFv2 has null authentication, simple password authentication and cryptographic authentication options for security. However, the operators can configure only one option per OSPF interface. Usually, MD5 or HMAC is used.

For the null authentication, the RSPD entry will specify bypass behavior for the traffic. This way the packets will be allowed to pass through. Use of this option is not recommended. The simple password mechanism is not suggested for use, however, CKT's "key" field contains the password that the OSPF can use directly as a simple password. Lastly, the cryptographic mechanisms. The RSPD contains a list of desired security protocols for the OSPF. The KMPs should be able to use this list for negotiation/establishment of a SA. The KMPs use the keys, lifetime and other information from the CKT for a key change or a SA renegotiation.

Below we illustrate two examples that specify the RPsec entries for securing the OSPFv2 traffic:

- OSPFv2 is unicast on a Non Broadcast Multiaccess (NBMA) network such as frame-relay. In this case, the operators manually configure the neighbors for the OSPF routers.

- OSPFv2 is multicast on the broadcast networks such as Ethernet. In this case, the OSPF neighbors listen on two multicast addresses, 224.0.0.5 and 224.0.0.6. The first address is used to send updates to ALL_OSPF_ROUTERS and the second is used to send updates to ALL_DESIGNATED_ROUTERS.

All other routing protocol traffic should be discarded by default.

10.6.1 Example 1: OSPF on NBMA

In this case, the RPsec contains entries for the unicast OSPFv2 traffic.

Scenario For a OSPFv2 router 192.168.0.1, we have two neighbors—192.168.0.2 and 192.168.0.3—on the NBMA network. These neighbors lie on the interface “serial-0/1”. The organization has decided the following security policy for the OSPFv2– The Authentication type should be pre-shared secret. There are two choices for the security protocol–

- OSPFv2_HMAC with RP_AUTH_HMAC_SHA2_384
- OSPFv2_KEYED_MD5 with RP_AUTH_KEYED_MD5

The HMAC is preferred because it is the first entry in the list (remember that the list of security protocol is a user-ordered list). Also, the HMAC stronger than the MD5. A unicast KMP negotiates for these security protocols. We assume that the KMP successfully established an SA for using the OSPFv2_HMAC security protocol.

Example RPAD Entry For OSPF on NBMA Table 3 shows the simplified view of a RPAD entry. Following it is an equivalent XML representation of a RPAD entry.

RPAD Field	Value
local-id	192.168.0.1
Local Credentials	
encoding	raw_rsa_key
authdata	1231kjh#aasdf
CA	OPAQUE
auth-type	PRE-SHARED-KEY
rpad-peers List	
peer-id	192.168.0.2
group-id	OPAQUE

Remote	Credentials
encoding	raw_rsa_key
authdata	1231kjh#aasdf
CA	OPAQUE
key-management	
KMPtype	UNICAST
KMPID	Suitable KMP-ID
peer-id	192.168.0.3
group-id	OPAQUE
Remote Credentials	
encoding	raw_rsa_key
authdata	1231kjh#aasdf
CA	OPAQUE
key-management	
KMPtype	UNICAST
KMPID	Suitable KMP-ID
interface	serial-0/1
Lifetime	
sendlifetimestart	20140911000000Z
acceptlifetimestart	20140911000500Z
sendlifetimeend	20141111235000Z
acceptlifetimeend	20141112000000Z

Table 3: Example RPAD entry for OSPFv2 on NBMA

```
<?xml version="1.0" encoding="UTF-8"?>
<p:rpsec-rpad xmlns:p="http://rpsec-example.com/rpad"
```

```
xmlns:p1="urn:ietf:params:xml:ns:yang:rpsec-common-types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://rpsec-example.com/rpad rpad.xsd ">
  <p:rpad-entry>
    <p:local-id>192.168.0.1</p:local-id>
    <p:credential>
      <p:encoding>raw_rsa_key</p:encoding>
      <p:authdata>1231kjh#aasdf</p:authdata>
      <p:CA>OPAQUE</p:CA>
    </p:credential>
    <p:auth-type>PRE-SHARED-SECRET</p:auth-type>
    <p:rpad-peers>
      <p:peer-id>192.168.0.2</p:peer-id>
      <p:group-id>OPAQUE</p:group-id>
      <p:credential>
        <p:encoding>raw_rsa_key</p:encoding>
        <p:authdata>1231kjh#aasdf</p:authdata>
        <p:CA>OPAQUE</p:CA>
      </p:credential>
      <p:key-management>
        <p:KMPtype>UNICAST</p:KMPtype>
        <p:KMPID>Suitable KMP-ID</p:KMPID>
      </p:key-management>
    </p:rpad-peers>
    <p:rpad-peers>
      <p:peer-id>192.168.0.3</p:peer-id>
      <p:group-id>OPAQUE</p:group-id>
      <p:credential>
        <p:encoding>raw_rsa_key</p:encoding>
        <p:authdata>1231kjh#aasdf</p:authdata>
        <p:CA>OPAQUE</p:CA>
      </p:credential>
      <p:key-management>
        <p:KMPtype>UNICAST</p:KMPtype>
        <p:KMPID>Suitable KMP-ID</p:KMPID>
      </p:key-management>
    </p:rpad-peers>
    <p:interface>serial-0/1</p:interface>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:rpad-entry>
```

```
</p:rpsec-rpad>
```

Example RSPD Entry For OSPF on NBMA Table 4 shows the simplified view of an entry in RSPD. Following it is an equivalent XML representation of the RSPD entry.

RSPD Field	First Entry	Second Entry
src-ip-address	192.168.0.1	192.168.0.1
dest-ip-address	192.168.0.2	192.168.0.3
Source Port		
port-lower	OPAQUE	OPAQUE
port-upper	OPAQUE	OPAQUE
Destination Port		
port-lower	OPAQUE	OPAQUE
port-upper	OPAQUE	OPAQUE
next-layer-protocol	89	89
processing-behavior	PROTECT	PROTECT
interface	serial-0/1	serial-0/1
direction	SYMMETRIC	SYMMETRIC
Security Protocol		
protocol-id	OSPFv2_HMAC	OSPFv2_HMAC
mode	OPAQUE	OPAQUE
transform-type	INTEG	INTEG
transform-id	RP_AUTH_HMAC_SHA2_384	RP_AUTH_HMAC_SHA2_384
protocol-id	OSPFv2_KEYED_MD5	OSPFv2_KEYED_MD5
mode	OPAQUE	OPAQUE
transform-type	INTEG	INTEG

transform-id	RP_AUTH_KEYED_MD5	RP_AUTH_KEYED_MD5
Lifetime		
sendlifetimestart	20140911000000Z	20140911000000Z
acceptlifetimestart	20140911000500Z	20140911000500Z
sendlifetimeend	20141111235000Z	20141111235000Z
acceptlifetimeend	20141112000000Z	20141112000000Z

Table 4: Example RSPD entry for OSPFv2 on NBMA

```
<?xml version="1.0" encoding="UTF-8"?>
<p:rpsec-rspd xmlns:p="http://rpsec-example.com/rspd"
xmlns:p1="urn:ietf:params:xml:ns:yang:rpsec-common-types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://rpsec-example.com/rspd rspd.xsd ">
  <p:rpsec-entry>
    <p:src-ip-address>192.168.0.1</p:src-ip-address>
    <p:dest-ip-address>192.168.0.2</p:dest-ip-address>
    <p:next-layer-protocol>89</p:next-layer-protocol>
    <p:src-ipv6-add>OPAQUE</p:src-ipv6-add>
    <p:dest-ipv6-add>OPAQUE</p:dest-ipv6-add>
    <p:source-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:source-port>
    <p:destination-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:destination-port>
    <p:processing-behavior>PROTECT</p:processing-behavior>
    <p:interface>serial-0/1</p:interface>
    <p:direction>SYMMETRIC</p:direction>
    <p:security-protocol><!--User ordered list of security protocol-->
      <p:protocol-id>OSPFv2_HMAC</p:protocol-id>
      <p:mode>OPAQUE</p:mode>
      <p:transform-type>INTEG</p:transform-type>
      <p:transform-id>RP_AUTH_HMAC_SHA2_384</p:transform-id>
    </p:security-protocol>
    <p:security-protocol>
      <p:protocol-id>OSPFv2_KEYED_MD5</p:protocol-id>
      <p:mode>OPAQUE</p:mode>
      <p:transform-type>INTEG</p:transform-type>
      <p:transform-id>RP_AUTH_KEYED_MD5</p:transform-id>
    </p:security-protocol>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141110235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:rpsec-entry>

  <p:rpsec-entry>
   <!--Destination address is set to 192.168.0.3,
   rest of the fields are same as above entry-->
  </p:rpsec-entry>
</p:rpsec-rspd>
```

Example CKT Entry For OSPF on NBMA Table 5 shows the CKT entry for the OSPFv2 traffic on a NBMA netowrk. Followed by it is an equivalent XML representation of CKT entry.

CKT Field	First Entry	Second Entry
protocol	OSPFv2_HMAC	OSPFv2_KEYED_MD5
adminkeyname	OSPF-UNICAST-KEY	OSPF-UNICAST-KEY
localkeyname	119	123
peerkeyname	911	321
peers list		
peer-id	192.168.0.2	192.168.0.2
peer-id	192.168.0.3	192.168.0.3
mode	OPAQUE	OPAQUE
protocol-specific-info	OPAQUE	OPAQUE
interface	serial-0/1	serial-0/1
KDF	NONE	NONE

ALGID	RP_AUTH_HMAC_SHA2_384	RP_AUTH_KEYED_MD5
Key	24DEE	24DAB
direction	SYMMETRIC	SYMMETRIC
Lifetime		
sendlifetimestart	20140911000000Z	20140911000000Z
acceptlifetimestart	20140911000500Z	20140911000500Z
sendlifetimeend	20141111235000Z	20141111235000Z
acceptlifetimeend	20141112000000Z	20141112000000Z

Table 5: Example CKT entry for OSPFv2 on NBMA

```
<?xml version="1.0" encoding="UTF-8"?>
<p:rpsec-ckt xmlns:p="http://rpsec-example.com/ckt"
xmlns:p1="urn:ietf:params:xml:ns:yang:rpsec-common-types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://rpsec-example.com/ckt ckt.xsd ">
  <p:ckt-entry>
    <p:Protocol>OSPFv2_HMAC</p:Protocol>
    <p:adminkeyname>OSPF-UNCAST-KEY</p:adminkeyname>
    <p:localkeyname>119</p:localkeyname>
    <p:peerkeyname>911</p:peerkeyname>
    <p:peers>
      <p:peer-id>192.168.0.2</p:peer-id>
      <p:peer-id>192.168.0.3</p:peer-id>
    </p:peers>
    <p:mode>OPAQUE</p:mode>
    <p:protocol-specific-info>OPAQUE</p:protocol-specific-info>
    <p:interface>serial-0/1</p:interface>
    <p:KDF>NONE</p:KDF>
    <p:ALGID>RP_AUTH_HMAC_SHA2_384</p:ALGID>
    <p:Key>24DEE</p:Key>
    <p:direction>SYMMETRIC</p:direction>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:ckt-entry>
  <p:ckt-entry>
    <p:Protocol>OSPFv2_KEYED_MD5</p:Protocol>
    <p:adminkeyname>OSPF-UNCAST-KEY</p:adminkeyname>
    <p:localkeyname>123</p:localkeyname>
    <p:peerkeyname>321</p:peerkeyname>
    <p:peers>
      <p:peer-id>192.168.0.2</p:peer-id>
      <p:peer-id>192.168.0.3</p:peer-id>
    </p:peers>
    <p:mode>OPAQUE</p:mode>
    <p:protocol-specific-info>OPAQUE</p:protocol-specific-info>
    <p:interface>serial-0/1</p:interface>
    <p:KDF>NONE</p:KDF>
    <p:ALGID>RP_AUTH_KEYED_MD5</p:ALGID>
    <p:Key>24DAB</p:Key>
    <p:direction>SYMMETRIC</p:direction>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:ckt-entry>
</p:rpsec-ckt>
```

10.6.2 Example 2: OSPF on broadcast networks

In this example, we illustrate the RPsec entries for the OSPFv2 on a broadcast network such as Ethernet.

Scenario We have an OSPFv2 router 172.16.0.1, which is connected to two neighbors 172.16.0.2 and 172.16.0.3, via ethernet-0/1. The OSPFv2 on the broadcast networks uses two multicast addresses, 224.0.0.5 (ALL_OSPF_ROUTERS) and 224.0.0.6 (ALL_DESIGNATED_ROUTERS). Since two multicast addresses are used, we have two appropriate RSPD entries for the outgoing traffic. For incoming traffic, we have two RSPD policies that allow traffic to these multicast addresses from ANY source. The CKT has one entry for outgoing/incoming traffic to these mutlicast addresses.

The organization has decided to use following security policy in this case. The authentication type to use is pre-shared-secret (raw_rsa_key). The OSPFv2_HMAC and associated transform RP_AUTH_HMAC_SHA2_224, is the choice of the security protocol. A GCKS will provide the group keys for the multicast communication using a suitable GKMP.

Example RPAD Entry For OSPF on Broadcast Networks Table 6 shows the simplified view of a RPAD entry. Following it is an equivalent XML representationof this entry.

RPAD Field	Value
local-id	172.16.0.1
Local Credentials	
encoding	raw_rsa_key
authdata	154we@#aasdf
CA	OPAQUE
auth-type	PRE-SHARED-KEY
rpad-peers List	
peer-id	172.16.0.18 (GCKS-ID)
group-id	224.0.0.5
Remote	Credentials
encoding	raw_rsa_key
authdata	1231kjh#aasdf
CA	OPAQUE
key-management	
KMPtype	MULTICAST
KMPID	Suitable KMP-ID

peer-id	172.16.0.18 (GCKS-ID)
group-id	224.0.0.6
Remote Credentials	
encoding	raw_rsa_key
authdata	154we@#aasdf
CA	OPAQUE
key-management	
KMPtype	MULTICAST
KMPID	Suitable KMP-ID
interface	ethernet-0/1
Lifetime	
sendlifetimestart	20140911000000Z
acceptlifetimestart	20140911000500Z
sendlifetimeend	20141111235000Z
acceptlifetimeend	20141112000000Z

Table 6: Example RPAD entry for OSPFv2 on broadcast network

```
<?xml version="1.0" encoding="UTF-8"?>
<p:rpsec-rpad xmlns:p="http://rpsec-example.com/rpad"
xmlns:p1="urn:ietf:params:xml:ns:yang:rpsec-common-types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://rpsec-example.com/rpad rpad.xsd ">
  <p:rpad-entry>
    <p:local-id>172.16.0.1</p:local-id>
    <p:credential>
      <p:encoding>raw_rsa_key</p:encoding>
      <p:authdata>154we@#aasdf</p:authdata>
      <p:CA>OPAQUE</p:CA>
    </p:credential>
    <p:auth-type>PRE-SHARED-SECRET</p:auth-type>
    <p:rpad-peers>
      <p:peer-id>172.16.0.18</p:peer-id><!--GCKS-ID-->
      <p:group-id>224.0.0.5</p:group-id>
      <p:credential>
        <p:encoding>raw_rsa_key</p:encoding>
        <p:authdata>154we@#aasdf</p:authdata>
        <p:CA>OPAQUE</p:CA>
      </p:credential>
      <p:key-management>
        <p:KMPtype>MULTICAST</p:KMPtype>
        <p:KMPID>Suitable KMP-ID</p:KMPID>
      </p:key-management>
    </p:rpad-peers>

    <p:rpad-peers>
      <p:peer-id>172.16.0.18</p:peer-id><!--GCKS-ID-->
      <p:group-id>224.0.0.6</p:group-id>
      <p:credential>
        <p:encoding>raw_rsa_key</p:encoding>
        <p:authdata>154we@#aasdf</p:authdata>
        <p:CA>OPAQUE</p:CA>
      </p:credential>
      <p:key-management>
        <p:KMPtype>MULTICAST</p:KMPtype>
        <p:KMPID>Suitable KMP-ID</p:KMPID>
      </p:key-management>
    </p:rpad-peers>
    <p:interface>ethernet-0/1</p:interface>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:rpad-entry>
</p:rpsec-rpad>
```

Example RSPD Entry For OSPF Broadcast Network Table 7 and Table 8 show the simplified view of RSPD entries. Following it is an equivalent XML representation.

RSPD Field	First Entry	Second Entry
src-ip-address	172.16.0.1	172.16.0.1

dest-ip-address	224.0.0.5	224.0.0.6
Source Port		
port-lower	OPAQUE	OPAQUE
port-upper	OPAQUE	OPAQUE
Destination Port		
port-lower	OPAQUE	OPAQUE
port-upper	OPAQUE	OPAQUE
next-layer-protocol	89	89
processing-behavior	PROTECT	PROTECT
interface	ethernet-0/1	ethernet-0/1
direction	SENDER-ONLY	SENDER-ONLY
Security Protocol		
protocol-id	OSPFv2_HMAC	OSPFv2_HMAC
mode	OPAQUE	OPAQUE
transform-type	INTEG	INTEG
transform-id	RP_AUTH_HMAC_SHA2_224	RP_AUTH_HMAC_SHA2_224
Lifetime		
sendlifetimestart	20140911000000Z	20140911000000Z
acceptlifetimestart	20140911000500Z	20140911000500Z
sendlifetimeend	20141111235000Z	20141111235000Z
acceptlifetimeend	20141112000000Z	20141112000000Z

Table 7: Example RSPD entries for outgoing OSPFv2 traffic on the broadcast network

RSPD field	Third Entry	Fourth Entry
src-ip-address	ANY	ANY
dest-ip-address	224.0.0.5	224.0.0.6

Source Port		
port-lower	OPAQUE	OPAQUE
port-upper	OPAQUE	OPAQUE
Destination Port		
port-lower	OPAQUE	OPAQUE
port-upper	OPAQUE	OPAQUE
next-layer-protocol	89	89
processing-behavior	PROTECT	PROTECT
interface	ethernet-0/1	ethernet-0/1
direction	RECEIVER-ONLY	RECEIVER-ONLY
Security Protocol		
protocol-id	OSPFv2_HMAC	OSPFv2_HMAC
mode	OPAQUE	OPAQUE
transform-type	INTEG	INTEG
transform-id	RP_AUTH_HMAC_SHA2_224	RP_AUTH_HMAC_SHA2_224
Lifetime		
sendlifetimestart	20140911000000Z	20140911000000Z
acceptlifetimestart	20140911000500Z	20140911000500Z
sendlifetimeend	20141111235000Z	20141111235000Z
acceptlifetimeend	20141112000000Z	20141112000000Z

Table 8: Example RSPD entries for incoming OSPFv2 traffic on the broadcast network

```
<?xml version="1.0" encoding="UTF-8"?>
<p:rpsec-rspd xmlns:p="http://rpsec-example.com/rspd"
xmlns:p1="urn:ietf:params:xml:ns:yang:rpsec-common-types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:schemaLocation="http://rpsec-example.com/rspd rspd.xsd ">
<!--Outgoing OSPF traffic from this router-->
  <p:rpsec-entry>
    <p:src-ip-address>172.16.0.1</p:src-ip-address>
    <p:dest-ip-address>224.0.0.5</p:dest-ip-address>
    <p:next-layer-protocol>89</p:next-layer-protocol>
    <p:src-ipv6-add>OPAQUE</p:src-ipv6-add>
    <p:dest-ipv6-add>OPAQUE</p:dest-ipv6-add>
    <p:source-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:source-port>
    <p:destination-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:destination-port>
    <p:processing-behavior>PROTECT</p:processing-behavior>
    <p:interface>ethernet-0/1</p:interface>
    <p:direction>SENDER-ONLY</p:direction>
    <p:security-protocol>
      <p:protocol-id>OSPFv2_HMAC</p:protocol-id>
      <p:mode>OPAQUE</p:mode>
      <p:transform-type>INTEG</p:transform-type>
      <p:transform-id>RP_AUTH_HMAC_SHA2_224</p:transform-id>
    </p:security-protocol>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:rpsec-entry>

<!--Outgoing OSPF traffic from this router-->
  <p:rpsec-entry>
    <p:src-ip-address>172.16.0.1</p:src-ip-address>
    <p:dest-ip-address>224.0.0.6</p:dest-ip-address>
    <p:next-layer-protocol>89</p:next-layer-protocol>
    <p:src-ipv6-add>OPAQUE</p:src-ipv6-add>
    <p:dest-ipv6-add>OPAQUE</p:dest-ipv6-add>
    <p:source-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:source-port>
    <p:destination-port>
```

```
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:destination-port>
    <p:processing-behavior>PROTECT</p:processing-behavior>
    <p:interface>ethernet-0/1</p:interface>
    <p:direction>SENDER-ONLY</p:direction>
    <p:security-protocol>
      <p:protocol-id>OSPFv2_HMAC</p:protocol-id>
      <p:mode>OPAQUE</p:mode>
      <p:transform-type>INTEG</p:transform-type>
      <p:transform-id>RP_AUTH_HMAC_SHA2_224</p:transform-id>
    </p:security-protocol>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:rpsec-entry>

<!--Incoming OSPF traffic to this router-->
<p:rpsec-entry>
    <p:src-ip-address>ANY</p:src-ip-address>
    <p:dest-ip-address>224.0.0.5</p:dest-ip-address>
    <p:next-layer-protocol>89</p:next-layer-protocol>
    <p:src-ipv6-add>OPAQUE</p:src-ipv6-add>
    <p:dest-ipv6-add>OPAQUE</p:dest-ipv6-add>
    <p:source-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:source-port>
    <p:destination-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:destination-port>
    <p:processing-behavior>PROTECT</p:processing-behavior>
    <p:interface>ethernet-0/1</p:interface>
    <p:direction>RECEIVER-ONLY</p:direction>
    <p:security-protocol>
      <p:protocol-id>OSPFv2_HMAC</p:protocol-id>
      <p:mode>OPAQUE</p:mode>
      <p:transform-type>INTEG</p:transform-type>
      <p:transform-id>RP_AUTH_HMAC_SHA2_224</p:transform-id>
    </p:security-protocol>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:rpsec-entry>

<!--Incoming OSPF traffic to this router-->
<p:rpsec-entry>
    <p:src-ip-address>ANY</p:src-ip-address>
    <p:dest-ip-address>224.0.0.6</p:dest-ip-address>
    <p:next-layer-protocol>89</p:next-layer-protocol>
    <p:src-ipv6-add>OPAQUE</p:src-ipv6-add>
    <p:dest-ipv6-add>OPAQUE</p:dest-ipv6-add>
    <p:source-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:source-port>
    <p:destination-port>
      <p:lower-port>OPAQUE</p:lower-port>
      <p:upper-port>OPAQUE</p:upper-port>
    </p:destination-port>
    <p:processing-behavior>PROTECT</p:processing-behavior>
    <p:interface>ethernet-0/1</p:interface>
    <p:direction>RECEIVER-ONLY</p:direction>
    <p:security-protocol>
      <p:protocol-id>OSPFv2_HMAC</p:protocol-id>
      <p:mode>OPAQUE</p:mode>
      <p:transform-type>INTEG</p:transform-type>
      <p:transform-id>RP_AUTH_HMAC_SHA2_224</p:transform-id>
    </p:security-protocol>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:rpsec-entry>
</p:rpsec-rspd>
```

Example CKT Entry For OSPF on Broadcast Networks Table 9 shows the CKT entry for OSPFv2. Followed by it is an equivalent XML representation.

CKT Field	Value
protocol	OSPFv2_HMAC

adminkeyname	OSPF-MULTICAST-KEY
localkeyname	234
peerkeyname	423
peers list	
peer-id	224.0.0.5
peer-id	224.0.0.6
mode	OPAQUE
protocol-specific-info	OPAQUE
interface	ethernet-0/1
KDF	NONE
ALGID	RP_AUTH_HMAC_SHA2_224
Key	AE24D
direction	SYMMETRIC
Lifetime	
sendlifetimestart	20140911000000Z
acceptlifetimestart	20140911000500Z
sendlifetimeend	20141111235000Z
acceptlifetimeend	20141112000000Z

Table 9: Example CKT entry for OSPFv2 on broadcast network

```
<?xml version="1.0" encoding="UTF-8"?>
<p:rpsec-ckt xmlns:p="http://rpsec-example.com/ckt"
xmlns:p1="urn:ietf:params:xml:ns:yang:rpsec-common-types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://rpsec-example.com/ckt ckt.xsd ">
  <p:ckt-entry>
    <p:Protocol>OSPFv2_HMAC</p:Protocol>
    <p:adminkeyname>OSPF-MULTICAST-KEY</p:adminkeyname>
    <p:localkeyname>234</p:localkeyname>
    <p:peerkeyname>423</p:peerkeyname>
    <p:peers>
      <p:peer-id>224.0.0.5</p:peer-id>
      <p:peer-id>224.0.0.6</p:peer-id>
    </p:peers>
    <p:mode>OPAQUE</p:mode>
    <p:protocol-specific-info>OPAQUE</p:protocol-specific-info>
    <p:interface>ethernet-0/1</p:interface>
    <p:KDF>NONE</p:KDF>
    <p:ALGID>RP_AUTH_HMAC_SHA2_224</p:ALGID>
    <p:Key>AE24D</p:Key>
    <p:direction>SYMMETRIC</p:direction>
   <p:sendlifetimestart>20140911000000Z</p:sendlifetimestart>
   <p:acceptlifetimestart>20140911000500Z</p:acceptlifetimestart>
   <p:sendlifetimeend>20141111235000Z</p:sendlifetimeend>
   <p:acceptlifetimeend>20141112000000Z</p:acceptlifetimeend>
  </p:ckt-entry>
</p:rpsec-ckt>
```

Chapter 11

Conclusion and Future Work

This chapter summarizes the work done to accomplish the goal stated in Chapter 7.

As per guidelines in [28], RPsec addresses the following threats to routing protocols-
Outsider: By using the information in RPAD the validity of a routing peer can be established, thus protecting against the spoofing and the man-in-the-middle attacks. In case if a router has been compromised, the RPAD and the RSPD databases of the other routers can be updated to exclude the compromised router from the valid participants list.
Unauthorized Key Holder: To prevent an attacker from gaining access to the traffic key that is used to integrity protect the exchanged messages, the administrators should specify multiple entries in the RPsec databases such that no one key/SA is used for long. Considering that a fairly strong security protocol is being used, regularly changing/rolling-over the keys/SAs will reduce the vulnerability due to cryptanalysis of the routing protocol packets.
Terminated Employee: A terminated employee is an example of unauthorized key holder. Once an employee leaves the organization, the security keys must be changed in order to minimize the potential attack source.
Following summarizes the role of RPsec:

- It provisions authentication information for the routing protocol peers.
- It provides support for KMPs for dynamic negotiation, establishment and rekey/rollover of SAs for the routing protocols when available.
- Administrators can specify multiple security mechanisms in the RSPD for the routing protocol.
- It overcomes the problems faced by the operators using the present manual methods of security. One of the major problem that it overcomes is that it supports regular key changes for the routing protocols.
- Finally, we provide a way to represent the above information such that it can be easily modified, configured and distributed in the network.

We explained the way RPsec fits into the routing protocol environment, and the ways to use it with manual and automated key management protocols in Chapter 8. We then defined the Yang modules for RPsec. The RPsec consist of authorization information in RPAD, security policies for negotiation in RSPD and key parameters associated with the each RSPD entry in CKT. These databases can be configured using the robust NETCONF protocol. Finally, we provided an architecture for the distribution of the management information in the RPsec that can be implemented using the NETCONF. The suggested architecture can be implemented following the well established IETF guidelines in [43] and [37]. Such an implementation will allow the operators to manage the key/SA for all the routing protocols in a single management interface. This will benefit an organization economically when it does not want to engage its personnel in laboriously changing keys/SA on all the routers in the network. This will ease the job of the network operators and they will be motivated to regularly revise the security policy of the routing protocols. Whereas in manual methods, the laborious task of changing keys/SA for routing protocols held them from doing regular revision of the security policy of the routing protocols. This leads to a secure routing infrastructure, which in turn will insure safe routing on the Internet.

11.1 Future Work

The next step in providing a robust security for routing protocols is the development of KMP solutions using the RPsec framework. Other area that needs to be explored is integration of this framework with existing routing protocols for replacement of manual methods. Another interesting field that needs to be explored is the use of authentication information in the RPAD database directly by the routing protocol. In it, the routing protocol may use the RPAD directly (without a KMP) for adjacency management.

Appendix A

RPsec Common Types Module

```
module rpsec-common-types {
  namespace "urn:ietf:params:xml:ns:yang:rpsec-common-types";
  prefix rpsectype;

  import ietf-yang-types {
    prefix yang;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization "Master Thesis RPsec";
  contact
    "Authors:
     J. William Atwood
     Nitin Prajapati";
  description
    "This module contains all the essential data types needed in
     RPsec framework.";

  revision 2014-08-17 {
    description
      "Initial revision";
    reference
      "Thesis: A Security Framework for Routing Protocols
       (RPsec)";
  }
```

```
typedef variable-data {
  type string;
  description
    "This type contains variable length data.
     Other types that use it should define the constrains on the
     length of this data.";
}

typedef interface-id {
  type string;
  default "ALL";
  description
    "interface-id refers to one of the following:
     -configured interface is the one for which the configuration is
     available on the device but interface itself is not enabled.
     -ALL refers to all the available interfaces on the router.";
}

typedef default-field-value {
  type enumeration {
    enum "OPAQUE";
    enum "ANY";
  }
  description
    "It specifies the default field values for each selector in the
     RPsec component databases.
     -OPAQUE is used when ever a value is not available for the
      corresponding field.
     -ANY is used as '*' (a wild card) that matches any value in the
      corresponding field. It should be use as specified in RFC4301";
}

typedef direction {
  type enumeration {
    enum "SENDER-ONLY";
    enum "RECEIVER-ONLY";
    enum "SYMMETRIC";
  }
  description
    "The direction indicates should the corresponding entry be used
     for inbound traffic, outbound traffic or both.
     -SENDER-ONLY when only outbound traffic is protected by the
     corresponding policy.
```

```
      -RECEIVER-ONLY when only incoming traffic is expected to be
       protected by the corresponding policy.
      -SYMMETRIC when the same policy should be used for protecting
       the inbound and outbound traffic.
      This field is important for specifying direction for routing
      protocols using multicast communication";
}

typedef vendor-id {
  type yang:yang-identifier;
  description
    "A Vendor specific device identity.";
}

typedef email-add {
  type string;
  description
    "RFC822 standard Email address type";
}

typedef device-id {
  type union {
    type inet:ip-address;
    type inet:domain-name;
    type rpsectype:email-add;
    type rpsectype:vendor-id;
  }
  description
    "As per RFC4301, a host may have one of the following identity
     that should be used by an AKM for selecting an entry and
     referring to a device";
}

grouping lifetime {
  description
    "Lifetime describes the validity of an entry with which it is
     associated ";
  leaf sendlifetimestart {
    type yang:date-and-time;
    mandatory true;
    description
      "The earliest time at which the corresponding policy is valid
       to be sent for negotiation or use. The policy is valid for
```

```
         negotiation by an AKM at this time if corresponding entry
         lies in RSPD and policy is ready to be considered for use if
         the corresponding entry is in CKT database.";
    }
    leaf acceptlifetimestart {
      type yang:date-and-time;
      mandatory true;
      description
        "The earliest time at which the corresponding policy is valid
         for use when processing the routing protocol traffic of
         interest. In case of RSPD entry this time specifies to an AKM
         to consider this entry for negotiation. In case of CKT entry,
         it specifies the keys and associated security protocols that
         should be used for processing the traffic of interest.";
    }
    leaf sendlifetimeend {
      type yang:date-and-time;
      mandatory true;
      description
        "The latest time at which the corresponding policy is valid
         for use. If this field expries, the entry should not be used
         anymore for processing the traffic. In case of RSPD, an AKM
         should not use the entry for negotiation beyond this time.
         In case of CKT, the keys should not be used to process the
         outgoing traffic";
    }
    leaf acceptlifetimeend {
      type yang:date-and-time;
      mandatory true;
      description
        "The latest time at which the corresponding policy is valid
         for use. If this field expries, the entry should not be used
         anymore for processing the traffic.
         In case of RSPD, an AKM must stop negotiation for the
         corresponding entry beyond this time.
         In case of CKT, the keys must not be used to process the
         incoming traffic";
    }
  }
}
```

Appendix B

RPAD YANG Module

```
module rpad {
  namespace "http://rpsec-example.com/rpad";
  prefix rpad;

  import rpsec-common-types {
    prefix rpsectype;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization "Master Thesis RPsec";
  contact
    "Authors:
     J. William Atwood
     Nitin Prajapati";
  description
    "This is the RPsec's Routing Peer
     Authorizaation Database (RPAD) module.";

  revision 2014-08-17 {
    description
      "Initial revision";
    reference
      "Thesis: A Security Framework for
     Routing Protocols (RPsec)";
}

feature mrpad {
  description
    "This feature means the device supports the multicast RPAD
     as defined in thesis - A Security Framework for Routing
     Protocols (RPsec).";
}

typedef authtype {
  type enumeration {
    enum "PRE-SHARED-SECRET";
    enum "CERTIFICATE";
  }
  description
    "This field indicate the type of authentication to be
     performed. Presently two types of authentication are
     available---pre-shared secret and X.509 certificate.
     The encoding of these types of authentication data is
     described in \\xs{sec:ike-encoding}";
}

typedef cert-encoding {
  type enumeration {
    enum "X.509_cert_sign" {
      value 4;
    }
    enum "CRL" {
      value 7;
    }
    enum "raw_rsa_key" {
      value 11;
    }
    enum "hash_url_x.509_cert" {
      value 12;
    }
    enum "hash_url_x.509_bundle" {
      value 13;
    }
  }
  description
    "It indicates the type of certificate
```

```
         or certificate-related information contained in
         the certificate data field.
         Only five types of certificate types are
         included the syntax of which are specifed in
         RFC5996.
         -X.509 Certificate Signature
         (x.509_cert_sign)
         -Raw RSA Key (raw_rsa_key) used only when
         auth-type is PRE-SHARED-SECRET.
         -Hash and URL of x.509 certificate
         (hash_url_x.509_cert)
         -Hash and URL  of x.509 bundle
         (hash_url_x.509_bundle)
         Except Raw RSA Key, all other types are used
         when authtype is CERTIFICATE.
        ";
}

typedef cert-data {
  type rpsectype:variable-data;
  description
    "It is the actual encoding of
     the certification data the type of which is
     indicated by cert-encoding as mention in
     RFC5996.";
}

typedef cert-authority {
  type rpsectype:variable-data;
  description
    "Same as in RFC5996.
     The certificate authority field contains an
     indicator of the trusted authorities for the
     certificate type mentioned by field
     cert-encoding.
     Certification Authority (CA) values is a
     concatenated list of SHA-1 hashes of the
     public keys of trusted CAs. It is specified
     only for cert-encoding types 4, 12 and 13. ";
}

typedef KMP-type {
  type enumeration {
    enum "UNICAST";
    enum "GROUP";
  }
  description
    "It specifies the
     type of key management protocol:
     -UNICAST means unicast KMPs like IKE
     -GROUP means group KMPs like GDOI/GSAKMP";
}

typedef KMP-ID {
  type yang:yang-identifier;
  description
    "Identifier of the KMP protocol.";
}

grouping certificate {
  description
    "This grouping contains the certificate relevant
    information.";
  leaf encoding {
    type rpad:cert-encoding;
    mandatory true;
    description
      "Certificate encoding";
  }
  leaf authdata {
    type rpad:cert-data;
    mandatory true;
    description
      "Certificate data depending on the encoding type
       defined RFC 5996";
  }
  leaf CA {
    type rpad:cert-authority;
    description
      "A valid certificate authority for this certificate.";
  }
}

grouping credential {
  description
    "It specify the type of credentials to be used.
     It should be consistent with the auth-type field:
     -When authtype is PRE-SHARED-SECRET, the certificate
     contains encoding=raw_rsa_key, data=pre-shared-secret
     and CA is set to OPAQUE.
     -When authtype is CERTIFICATE, the certificate contains
     encoding=other than raw_rsa_key, data=actual certificate
     data and CA contains list of certificate authorities.
    ";
  uses rpad:certificate;
}

container rpsec-rpad {
  description
    "This container contains RPAD entries.";
  list rpad-entry {
    key "local-id";
    description
      "An rpad-entry can be fetched
       using local-id and peer-identity";
    leaf local-id {
      type rpsectype:device-id;
      mandatory true;
      description
        "Identity of the local device";
    }
    container credential {
      description
        "Credentials associated with the local
         device's id";
      uses rpad:credential;
    }
    leaf auth-type {
      type rpad:authtype;
      mandatory true;
      description
        "For each RPAD entry it indicates the
         authentication type for the ID asserted by the device";
    }
    list rpad-peers {
      key "peer-id";
      ordered-by user;
      description
        "This container specifies the RPAD entry for each peer
         associated with local-id.";
      leaf peer-id {
        type rpsectype:device-id;
        mandatory true;
        description
          "In case of unicast
           communication it specify the peer
           identity.
           List of peers and associated credentials
           corresponding to local-device’s id.
           The credentials of local device and all peers in this
           list must macth the auth-type described for this RPAD
           entry. In case mrpad feature is used,the peer id refers
           to a GCKS identity that represents the multicast group
           local device wishes to communicate with.";
      }
      leaf group-id {
        if-feature mrpad;
        type union {
          type inet:ipv4-address-no-zone;
          type inet:ipv6-address-no-zone;
        }
        description
          "Multicast IP address of the group.";
      }
      container credential {
        description
          "The credentials associated with the peer-id";
        uses rpad:credential;
      }
      container key-management {
        description
          "This container contains the information of KMP protocol.";
        leaf KMPtype {
          type KMP-type;
          mandatory true;
          description
            "Specifies unicast or group KMP";
        }
        leaf KMPID {
          type yang:yang-identifier;
          mandatory true;
          description
```

```
                "Identity of the KMP protocol.";
            }
          }
        }
        leaf interface {
          type rpsectype:interface-id;
          mandatory true;
          description
          "Interface on which to use the specified KMP.
          The peers are connected to same network as this interface.";
        }
        uses rpsectype:lifetime;
      }
    }
  }
```

Appendix C

RSPD YANG Module

```
module rspd {
  namespace "http://rpsec-example.com/rspd";
  prefix rspd;

  import rpsec-common-types {
    prefix rpsectype;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization "Master Thesis RPsec";
  contact
    "Authors:
     J. William Atwood
     Nitin Prajapati";
  description
    "This is the RPsec's Routing Security Policy
     Database (RSPD) module.";

  revision 2014-08-17 {
    description
      "Initial revision";
    reference
      "Thesis: A Security Framework for
     Routing Protocols (RPsec)";
}

feature ipv6 {
  description
    "Device supports ipv6.";
}

feature mrspd {
  description
    "Device supports multicast RSPD.";
}

grouping port-range {
  description
    "Specify the port range for matching the routing
     protocol traffic.";
  leaf lower-port {
    type union {
      type inet:port-number;
      type rpsectype:default-field-value;
    }
    mandatory true;
    description
      "Minimum value of the range";
  }
  leaf upper-port {
    type union {
      type inet:port-number;
      type rpsectype:default-field-value;
    }
    mandatory true;
    description
      "Maximum value of the range";
  }
}

container rpsec-rspd {
  description
    "Contains all rspd entries.";
  list rpsec-entry {
    key "src-ip-address dest-ip-address next-layer-protocol";
    description
  "Contain rspd-entries.
   A process will use following keys to
   search for an entry for traffic of interest
   -Source IP address
   -Destination IP address
   -Next Layer Protocol
  ";
leaf src-ip-address {
  type inet:ipv4-address-no-zone;
  mandatory true;
  description
    "Source IPv4 address.";
}
leaf dest-ip-address {
  type inet:ipv4-address-no-zone;
  mandatory true;
  description
    "Destination IPv4 address.";
}
leaf src-ipv6-add {
  type inet:ipv6-address-no-zone;
  if-feature ipv6;
  description
    "source IPv6 address";
}
leaf dest-ipv6-add {
  type inet:ipv6-address-no-zone;
  if-feature ipv6;
  description
    "source IPv6 address";
}
container source-port {
  description
    "Match the traffic origin.";
  uses port-range;
}
container destination-port {
  description
    "Match the traffic destination.";
  uses port-range;
}
leaf next-layer-protocol {
  type uint16;
  mandatory true;
  description
    "Net layer protocol field that matches the
     protocol field in IPv4 and next header value in IPv6";
}
leaf processing-behavior {
  type enumeration {
    enum "PROTECT";
    enum "BYPASS";
    enum "DISCARD";
  }
  mandatory true;
  description
    "This field specifies the processing behavior for the routing protocol t
}
leaf interface {
  type rpsectype:interface-id;
  mandatory true;
  description
    "As desribed in Section 8.1.2, Chapter 8 of this thesis.";
}
leaf direction {
  if-feature mrspd;
  type rpsectype:direction;
  description
    "Specify the directionality for multicast communications,
    as explained in Secton 8.2, Chapter 8 of this thesis.";
}
list security-protocol {
  key "protocol-id";
  ordered-by user;
  description
    "User ordered choice of security protocol for
     the traffic matching the traffic selector. It allows
     specifying multiple security protocols for matching
     traffic selectors. It provides agility to the choice of
     security protocols for routing protocols.";
  leaf protocol-id {
    type yang:yang-identifier;
    mandatory true;
    description
      "Security Protocol ID";
  }
```

```
        leaf mode {
          type union {
            type rpsectype:default-field-value;
            type enumeration {
              enum "TRANSPORT-MODE";
              enum "TUNNEL-MODE";
            }
          }
          description
            "Security protocol mode, as described in Section 8.1.2,
             Chapter 8.";
        }
        leaf transform-type {
          type enumeration {
            enum "INTEG";
            enum "ENCR";
            enum "INTEG_ENCR";
          }
          mandatory true;
          description
            "transform type
             -INTEG for intergrity type
             -ENCR for encryption type
             -INTEG_ENCR for both integrity and encryption";
        }
        leaf transform-id {
          type yang:yang-identifier;
          mandatory true;
          description
            "The transform id of type transform-type as supported by
             security protocol id. It is a user ordered list of
             transform ids.
             This order is followed when negotiating the most
             appropriate transforms between peers. The peers
             must choose one transform from the list.
             This was a leaf-list previously. Protocol-id+transform-id
             represents unique entry.";
        }
      }
      uses rpsectype:lifetime;
    }
  }
}
```

Appendix D

CKT YANG Module

```
module ckt {
  namespace "http://rpsec-example.com/ckt";
  prefix ckt;

  import ietf-yang-types {
    prefix yang;
  }
  import rpsec-common-types {
    prefix rpsectype;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization "Master Thesis RPsec";
  contact
    "Authors:
     J. William Atwood
     Nitin Prajapati";
  description
    "This module contains all the essential data types needed in
     RPsec framework.";

  revision 2014-08-17 {
    description
      "Initial revision";
    reference "Thesis:
    A Security Framework for Routing Protocols (RPsec)";
```

```
}

container rpsec-ckt {
  description
    "Contains all the CKT Entries";
  list ckt-entry {
    key "Protocol";
    description
      "List of ckt entries.
       A CKT entry can be fetched using
       peer-identity and security-protocol as
       described in RFC7210.";
    leaf adminkeyname {
      type yang:yang-identifier;
      description
        "As specified in RFC7210";
    }
    leaf localkeyname {
      type yang:yang-identifier;
      mandatory true;
      description
        "As specified in RFC7210";
    }
    leaf peerkeyname {
      type yang:hex-string;
      mandatory true;
      description
        "PeerKeyName as in RFC7210";
    }
    list peers {
      key "peer-id";
      mandatory true;
      leaf peer-id {
        type inet:ip-address-no-zone;
        description
          "IPv4 or IPv6 address";
      }
      description
        "list of peers as described in
         RFC7210.";
    }
    leaf Protocol {
      type yang:yang-identifier;
  mandatory true;
  description
    "Refers to seccurity protocol configuration as in RSPD.";
}
leaf mode {
  type string;
  default "OPAQUE";
  description
    "refers to security protocol configuration mode as in RSPD.";
}
leaf protocol-specific-info {
  type string;
  description
    "It contain information as specified in
     KeyTable Protocols, RFC7210.";
}
leaf interface {
  type rpsectype:interface-id;
  mandatory
  description
    "Interface as in RFC7210";
}
leaf KDF {
  type union {
    type yang:yang-identifier;
    type enumeration {
      enum "NONE";
    }
  }
  description
    "As specified in RFC7210";
}
leaf ALGID {
  type yang:yang-identifier;
  mandatory true;
  description
    "The value in this field will be the negotiated
     transform-id as specified in RSPD and in conformance
     with RFC7210";
}
leaf Key {
  type yang:hex-string;
  mandatory true;
```

```
          description
            "The master key, as specified in RFC7210";
        }
        leaf direction {
          type rpsectype:direction;
          mandatory true;
          description
            "As opposed to in/out/both/disabled,this
             field will contain values as:
             -out equivalent to SENDER-ONLY.
             -in equivalent to RECEIVER-ONLY.
             -both equivalent to SYMMETRIC.
            ";
        }
        uses rpsectype:lifetime;
      }
    }
  }
```

Appendix E

SA Requirements of Routing Protocols

This supplementary chapter outlines the SA requirements for security mechanisms of routing protocols as discussed in Section 2.6. The SA requirement for each solution is taken from its respective IETF standard document. The text in this chapter is explained in breif to acquaint the reader with existing SA parameters and is mostly included "as it is" to enhance understanding.

E.1 OSPFv2

E.1.1 Authentication Trailer [33]

AuthType– The value in this field shows the type of cryptographic authentication for the routing protocol.

Key ID–This field identifies the algorithm (which is MD5) and secret key used to create the message digest appended to the OSPF packet. Key Identifiers are unique per-interface (or equivalently, per-subnet). The length of authentication data generated is specified by MD5 algorithm. An interface may have multiple keys active at any one time for smooth transition from one key to another. Each key has a set lifetime. The lifetime is expressed as–

KeyStartAccept–The time that the router will start accepting packets that have been created with the given key.

KeyStartGenerate–The time that the router will start using the key for packet generation.

KeyStopGenerate–The time that the router will stop using the key for packet generation.

KeyStopAccept–The time that the router will stop accepting packets that have been created with the given key.

Cryptographic sequence number–An unsigned 32-bit non-decreasing sequence number. Used to guard against replay attacks. This information is chosen during the connection initiation.

E.2 OSPFv3

E.2.1 IPsec [18]

In order to implement this specification, the following IPsec capabilities are required–

Mode–IPsec in transport mode must be supported.

SPDs–The implementation must support multiple SPDs with an SPD selection function that provides an ability to choose a specific SPD based on interface.

Selectors–The implementation must be able to use source address, destination address, protocol, and direction as selectors in the SPD.

Interface ID–The implementation must be able to tag the inbound packets with the ID of the interface (physical or virtual) via which it arrived.

Manual key support–Manually configured keys must be able to secure the specified traffic.

Encryption and authentication algorithms–The AH and ESP security protocols are supported that are set manually.

Dynamic IPsec rule configuration–The routing module should be able to configure, modify, and delete IPsec rules on the fly. This is needed mainly for securing virtual links.

Encapsulation of ESP packet–IP encapsulation of ESP packets must be supported.

Different SAs for different Differentiated Services Code Points (DSCPs)– Multiple SAs between same sender and receiver allows the implementation to associate different classes of traffic with the same selector values in support of Quality of Service (QoS).

E.2.2 Authentication Trailer [12]

An OSPFv3 Security Association (SA) contains a set of parameters shared between any two legitimate OSPFv3 speakers. Parameters associated with an OSPFv3 AT SA are as follows–

AuthType–It shows the choice of cryptographic authentication for OSPFv3.

SA ID–It is used to uniquely identify an OSPFv3 SA, as manually configured by the network operator. The receiver determines the SA for incoming packet by looking at this field value. Using SA IDs makes changing keys while maintaining protocol operation convenient. Each SA ID specifies two independent parts, the authentication algorithm and the authentication Key, as explained below. Each SA ID can indicate a key with a different authentication algorithm. This allows the introduction of new authentication mechanisms without disrupting ongoing communication.

Authentication Algorithm–This signifies the authentication algorithm to be used with this OSPFv3 SA.

KeyStartAccept–The time that this OSPFv3 router will accept packets that have been created with this OSPFv3 SA.

KeyStartGenerate–The time that this OSPFv3 router will begin using this OSPFv3 SA for OSPFv3 packet generation.

KeyStopGenerate–The time that this OSPFv3 router will stop using this OSPFv3 SA for OSPFv3 packet generation.

KeyStopAccept–The time that this OSPFv3 router will stop accepting packets generated with this OSPFv3 SA.

Authentication Key–This value denotes the Cryptographic Authentication Key associated with this OSPFv3 SA. The length of this key is variable and depends upon the authentication algorithm specified by the OSPFv3 SA.

E.3 PIM-SM

In order to implement this specification, the following IPsec capabilities are required [6]–

Mode–IPsec in transport mode must be supported.

SPDs–The implementation must support multiple SPDs with an SPD selection function that provides an ability to choose a specific SPD based on interface.

Selectors–The implementation must be able to use source address, destination address, protocol, and direction as selectors in the SPD.

Interface ID–The implementation must be able to tag the inbound packets with the ID of the interface (physical or virtual) on which they arrived.

Manual key support–It must be possible to use manually configured keys to secure the specified traffic.

Encryption and authentication algorithms–The AH and ESP security protocols used with the IPsec. It is set using manual methods. Implementations must support ESP-NULL, and if providing confidentiality, must support the ESP transforms providing confidentiality required by RFC4835. Encapsulation of ESP packets–IP encapsulation of ESP packets must be supported.

If the automatic keying features of this specification are implemented, the following additional IPsec capabilities are required–

GSPD–The implementation must support the Group Security Policy Database that is described in [42].

Multiple Group Security Policy Databases–The implementation must support multiple GSPDs with a GSPD selection function that provides an ability to choose a specific GSPD based on interface.

Selectors–The implementation must be able to use source address, destination address, protocol and direction as selectors in the GSPD.

E.4 BGP

E.4.1 TCP-MD5 [33]

Key–This is the shared secret used as an input to the MD5 calculation of the BGP packet.

MD5 Hashing algorithm–This specification supports the MD5 algorithm for calculating the message digest.

TCP-AO [39]

MKT

A Master Key Tuple (MKT) describes TCP-AO properties to be associated with one or more connections. It is composed of the following–

TCP connection identifier–A TCP socket pair, i.e., a local IP address, a remote IP address, a TCP local port, and a TCP remote port. Values can be partially specified using ranges (e.g., 2-30), masks (e.g., 0xF0), wildcards (e.g., "*"), or any other suitable indication.

TCP option flag–This flag indicates whether TCP options other than TCP-AO are included in the MAC calculation.

IDs–The values used in the KeyID or RNextKeyID of TCP-AO. These IDs are used to differentiate MKTs in concurrent use (KeyID), as well as to indicate when MKTs are ready for use in the opposite direction (RNextKeyID). Each MKT has two IDs—a SendID and a RecvID.

The SendID is inserted as the KeyID of the TCP-AO option of outgoing segments, and the RecvID is matched against the TCP-AO KeyID of incoming segments.

Master key–It is used for generating traffic keys, this may be derived from a separate shared key by an external protocol over a separate channel.

KDF–Indicates the key derivation function and its parameters, as used to generate traffic keys from master keys.

MAC algorithm–Indicates the MAC algorithm and its parameters as used for this connection.

This document does not address how MKTs are created by users or processes. It assumes that the MKTs can be managed by a separate application protocol or manually.

KDF_Alg–Specify the KDFs to be used for deriving TCP-AO's traffic keys [29]. It is set manually for the BGP peers.

Context–A binary string containing information related to the specific connection for this derived keying material [39]. Output_Length–The length, in bits, of the key that the KDF will produce. This is specified by the KDF in use. MAC Algorithms–Each MAC_alg defined for TCP-AO has three fixed elements as part of its definition–KDF_Alg,Output-Length and MAC_Length.

E.5 RIPv2

Keyed-MD5 [7]

Following are the SA requirements for keyed-MD5 type security for RIPv2 [7]–
AuthType–Should be set to Keyed Message Digest.
Key ID–As with all security methods using MD5 with keys identified by this key identifier. Each key will have its own key identifier, which is stored locally. The combination of the key identifier and the interface associated with the message uniquely identifies the Authentication Algorithm and RIP-2 Authentication Key in use.

HMAC for RIPv2

The minimum data items in a RIPv2 Security Association are as follows [5]–
Key-ID–The key identifier is used to identify the RIPv2 Security Association in use for this packet.

The receiver uses the combination of the interface the packet was received upon and the Key-id value to uniquely identify the appropriate Security Association.

The sender selects which RIPv2 Security Association to use based on the outbound interface for this RIPv2 packet and then places the correct Key-ID value into that packet.
Auth-Alg–This specifies the cryptographic algorithm and algorithm mode used with the RIPv2 Security Association.
Auth-Key–This is the value of the cryptographic authentication key used with the associated Authentication Algorithm.
Sequence Number–This is an unsigned 32-bit number. For a given Key-ID value and sender, this number must NOT decrease. The initial value used in the sequence number is arbitrary.

Start Time–This is a local representation of the day and time that this Security Association first becomes valid.
Stop Time–This is a local representation of the day and time that this Security Association becomes invalid (i.e., when it expires). It is permitted, but not recommended, for an operator to configure this to ”never expire”.

Appendix F

Miscellaneous

F.1 Present Routing Protocol Integrity Algorithms

Table 10 shows the standard integrity algorithms for routing protocols. We assume

RP	Integrity Algorithm	RFC
OSPFv2	Keyed-MD5	RFC 2328
OSPFv2	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	RFC 5709
OSPFv3	IPsec	RFC 4552
OSPFv3	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	RFC 7166
BGP	Keyed-MD5	RFC 2385
BGP	HMAC-SHA-1-96, AES-128-CMAC-96	RFC 5926
RIPv2	KEYED-MD5, HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512	RFC 4822

Table 10: Routing protocol integrity algorithms

that routing protocols security protocols can be represented as shown in Table 11. The name of routing protocols are prefixed to their associated security algorithms. It is considered that IETF will come up with similar representations that can distinguish security protocols for routing protocols. These security protocols uses the integrity transforms as listed in the next section.

RP	Security Protocol	RFC
OSPFv2	OSPFV2_KEYED_MD5	RFC 2328
OSPFv2	OSPFV2_HMAC_SHA	RFC 5709
OSPFv3	OSPFV3_IPSEC	RFC 4552
OSPFv3	OSPFV3_HMAC_SHA	RFC 7166
PIM	PIM_IPSEC	RFC 5796
BGP	BGP_KEYED_MD5	RFC 2385
BGP	BGP_TCP_AO	RFC 5926
RIPv2	RIPV2_KEYED_MD5, RIPV2_HMAC_SHA	RFC 4822

Table 11: Routing protocol security protocols

F.2 Routing Protocol Integrity Transforms

Table 12 below shows possible representation of the list of integrity transforms that may be used in RPsec. These transforms are listed in Table 10.

Routing Protocol Integrity transforms
RP_AUTH_HMAC_SHA1
RP_AUTH_HMAC_SHA1_96
RP_AUTH_HMAC_SHA2_224
RP_AUTH_HMAC_SHA2_256
RP_AUTH_HMAC_SHA2_384
RP_AUTH_HMAC_SHA2_512
RP_AUTH_KEYED_MD5
RP_AUTH_MET_KEYED_SHA1
RP_AUTH_AES_128_CMAC_96
RP_AUTH_AH
RP_AUTH_ESP

Table 12: Routing protocol integrity transforms

The list is compiled from draft-tran-karp-mrmp

It is considered that IETF will come up with similar representations that can distinguish security protocols for routing protocols.

F.3 Routing Protocol KDFs

The table below shows the possible KDF algorithms for routing protocols that may be supported by RPsec.

Routing Protocol KDFs
KDF_HMAC_SHA1
KDF_AES_128_CMAC_96

Table 13: Routing protocol KDFs

The list is compiled from 5926

F.4 IKE Supported Identity

List of all the identities that are supported by IKE may also be supported in RPsec's RPAD database. These IDs must be supported by all the routing protocol KMPs:

IKE supported IDs
DNS name (specific or partial)
Distinguished Name (complete or sub-tree constrained)
RFC 822 email address (complete or partially qualified)
IPv4 address (range)
IPv6 address (range)
Key ID (exact match only)

Table 14: IKE supported identities

The list is compiled from RFC4301

F.5 IKE Supported Encoding of Authentication Data

This section lists the encoding types for IKE supported authentication data—pre-shared-secret and x.509 certificates.

Authentication Data	Encoding-value
X.509_cert_sign	4
CRL	7
raw_rsa_key	11
hash_url_x.509_cert	12
hash_url_x.509_bundle	13

Table 15: IKE supported authentication data encoding

The list is compiled from RFC4301

Bibliography

[1] Keying and authentication for routing protocols (KARP), IETF. http://datatracker.ietf.org/wg/karp/charter/.

[2] Yang central. http://www.yang-central.org/twiki/bin/view/Main/WebHome.

[3] The keyed-hash message authentication code. Technical report, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, https://www.usenix.org/legacy/event/lisa11/tech/full_papers/Wallin.pdf, July 2008. Category: Computer Security, Subcategory: Cryptography.

[4] Secure hash standard (SHS). Technical report, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf, March 2012. Category: Computer Security, Subcategory: Cryptography.

[5] R. Atkinson and M. Fanto. RIPv2 cryptographic authentication. RFC 4822, IETF, http://www.rfc-editor.org/rfc/rfc4822.txt, February 2007.

[6] W. Atwood, S. Islam, and M. Siami. Authentication and confidentiality in protocol independent multicast sparse mode (PIM-SM) link local messages. RFC 5796, IETF, http://www.rfc-editor.org/rfc/rfc5796.txt, March 2010.

[7] F. Baker and R. Atkinson. RIP-2 MD5 authentication. RFC 2082, IETF, http://www.rfc-editor.org/rfc/rfc2082.txt, January 1997.

[8] A. Barbir, S. Murphy, and Y. Yang. Generic threats to routing protocols. RFC 4593, IETF, http://www.rfc-editor.org/rfc/rfc4593.txt, October 2006.

[9] M. Baugher, R. Canetti, L. Dondeti, and F. Lindholm. Multicast security (MSEC) group key management architecture. RFC 4046, IETF, http://www.rfc-editor.org/rfc/rfc4046.txt, May 2005.

[10] S. Bellovin and R. Housley. Guidelines for cryptographic key management. RFC 4107, IETF, http://www.rfc-editor.org/rfc/rfc4107.txt, June 2005.

[11] M. Bhatia, V. Manral, M. Fanto, R. White, M. Barnes, T. Li, and R. Atkinson. OSPFv2 HMAC-SHA cryptographic authentication. RFC 5709, IETF, http://www.rfc-editor.org/rfc/rfc5709.txt, October 2009.

[12] M. Bhatia, V. Manral, and A. Lindem. Support authentication trailer for OSPFv3. RFC 6506, IETF, http://www.rfc-editor.org/rfc/rfc6506.txt, February 2012.

[13] M. Bjorklund. Yang - a data modeling language for the network configuration protocol (NETCONF). RFC 6020, IETF, http://www.rfc-editor.org/rfc/rfc6020.txt, October 2010.

[14] Ntanzi Carrilho and Neco Ventura. Distributed policy-based network management with NETCONF. Proceedings of SATNAC-2006 No. 292, Department of Electrical Engineering, University of Cape Town, Rondebosch, Cape Town, South Africa, http://www.satnac.org.za/proceedings/2006/papers/No%20292%20-%20Carrilho.pdf, September 2006. Network Management/OSS.

[15] U. Chunduri and J. Touch. A framework for RPs to use IKEv2 KMP. Internet-draft, IETF Secretariat, http://tools.ietf.org/id/draft-chunduri-karp-using-ikev2-with-tcp-ao-06.txt, January 2014.

[16] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman. Network configuration protocol (NETCONF). RFC 6241, IETF, http://www.rfc-editor.org/rfc/rfc6241.txt, June 2011.

[17] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas. Protocol independent multicast-sparse mode (PIM-SM): Protocol specification (revised). RFC 4601, IETF, http://www.rfc-editor.org/rfc/rfc4601.txt, August 2006.

[18] M. Gupta and N. Melam. Authentication/confidentiality for OSPFv3. RFC 4552, IETF, http://www.rfc-editor.org/rfc/rfc4552.txt, June 2006.

[19] D. Harkins and D. Carrel. The internet key exchange. RFC 2409, IETF, http://www.rfc-editor.org/rfc/rfc2409.txt, November 1998.

[20] H. Harney, U. Meth, A. Colegrove, and G. Gross. GSAKMP: Group secure association key management protocol. RFC 4535, IETF, http://www.rfc-editor.org/rfc/rfc4535.txt, June 2006.

[21] H. Harney and C. Muchenhir. Group key management protocol architecture. RFC 2094, IETF, http://www.rfc-editor.org/rfc/rfc2094.txt, July 1997.

[22] H. Harney and C. Muchenhir. Group key management protocol specification. RFC 2093, IETF, http://www.rfc-editor.org/rfc/rfc2093.txt, July 1997.

[23] R. Housley, T. Polk, S. Hartman, and D. Zhang. Database of long-lived symmetric cryptographic keys. RFC 7210, IETF, http://www.rfc-editor.org/rfc/rfc7210.txt, April 2014.

[24] M. Jethanandani, B. Weis, K. Patel, D. Zhang, S. Hartman, U. Chunduri, and J. Touch. Negotiation for keying pairwise routing protocols in IKEv2. Internet-draft (work in progress), IETF, `http://www.ietf.org/id/draft-mahesh-karp-rkmp-05.txt`, November 2013.

[25] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen. Internet key exchange protocol version 2 (IKEv2). RFC 5996, IETF, `http://www.rfc-editor.org/rfc/rfc5996.txt`, September 2010.

[26] S. Kent and K. Seo. Security architecture for the internet protocol. RFC 4301, IETF, `http://www.rfc-editor.org/rfc/rfc4301.txt`, December 2005.

[27] G. Lebovitz and M. Bhatia. Keying and authentication for routing protocols (KARP) design guidelines. RFC 6518, IETF, `http://www.rfc-editor.org/rfc/rfc6518.txt`, February 2012.

[28] G. Lebovitz, M. Bhatia, and B. Weis. Keying and authentication for routing protocols (KARP) overview, threats and requirements. RFC 6862, IETF, `http://www.rfc-editor.org/rfc/rfc6862.txt`, March 2013.

[29] G. Lebovitz and E. Rescorla. Cryptographic algorithms for the TCP authentication option (TCP-AO). RFC 5926, IETF, `http://www.rfc-editor.org/rfc/rfc5926.txt`, June 2010.

[30] B. Lengyel. Why we need a netconf-specific modeling language-yang. Internet-draft (work in progress), IETF, `http://www.yang-central.org/twiki/pub/Main/YangDocuments/draft-lengyel-why-yang-00.txt`, November 2007.

[31] D. Maughan, M. Schertler, M. Schneider, and J. Turner. Internet security association and key management protocol. RFC 2408, IETF, `http://www.rfc-editor.org/rfc/rfc2408.txt`, November 1998.

[32] D. Meyer, L. Zhang, and K. Fall. Report from the IAB workshop on routing and addressing. RFC 4948, IETF, `http://www.rfc-editor.org/rfc/rfc4984.txt`, September 2007.

[33] J. Moy. OSPF version 2. RFC 2328, IETF, `http://www.rfc-editor.org/rfc/rfc2328.txt`, April 1998.

[34] Dan Pei, Daniel Massey, and Lixia Zhang. A framework for resilient internet routing protocols. Network, IEEE (Volume:18, Issue:2), ISSN:0890-8044, DOI:10.1109/MNET.2004.1276605, March-April 2004.

[35] D. Piper. The internet ip security domain of interpretation for isakmp. RFC 2407, IETF, `http://www.rfc-editor.org/rfc/rfc2407.txt`, November 1998.

[36] L. Seitz and E. Rissanen. Netconf access control profile for XACML. Internet-draft (work in progress), IETF, `http://www.ietf.org/archive/id/draft-seitz-netconf-xacml-02.txt`, April 2008.

[37] P. Shafer. An architecture for network management using netconf and yang. RFC 6244, IETF, `http://www.rfc-editor.org/rfc/rfc6244.txt`, June 2011.

[38] Revathi Bangalore Somanatha. Design and validation of automated authentication, key and adjacency management for routing protocols. M.Comp.Sc. Thesis, Department of Computer Science and Software Engineering, Concordia University, Montreal, Quebec, Canada, August 2012.

[39] J. Touch, A. Mankin, and R.Bonica. The TCP authentication option. RFC 5925, IETF, `http://www.rfc-editor.org/rfc/rfc5925.txt`, June 2010.

[40] P. Tran and B. Weis. The use of G-IKEv2 for multicast router key management. Internet-draft (work in progress), IETF, `http://www.ietf.org/id/draft-tran-karp-mrmp-02.txt`, October 2012.

[41] Stefan Wallin and Claes Wikstrom. Automating network and service configuration using netconf and yang, The SANS Institute. `https://www.usenix.org/legacy/event/lisa11/tech/full_papers/Wallin.pdf`, 2011.

[42] B. Weis, G. Gross, and D. Ignjatic. Multicast extensions to the security architecture for the internet protocol. RFC 5374, IETF, `http://www.rfc-editor.org/rfc/rfc5374.txt`, November 2008.

[43] R. Yavatkar, D. Pendarakis, and R. Guerin. A framework for policy-based admission control. RFC 2753, IETF, `http://www.rfc-editor.org/rfc/rfc2753.txt`, January 2000.

[44] D. Zhang, S. Hartman, and W. Atwood. Routing authentication policy database. Internet-draft (work in progress), IETF Secretariat, `http://www.ietf.org/id/draft-zhang-karp-rapd-00.txt`, October 2013.