Operations Model for Router Keying
draft-ietf-karp-ops-model-01.txt

Abstract

Developing an operational and management model for routing protocol
security that works across protocols will be critical to the success of
routing protocol security efforts. This document discusses issues and
begins to consider development of these models.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

The KARP working group is designing improvements to the cryptographic authentication of IETF routing protocols. These improvements include improvements to how integrity functions are handled within each protocol as well as designing an automated key management solution.

This document discusses issues to consider when thinking about the operational and management model for KARP. Each implementation will take its own approach to management; this is one area for vendor differentiation. However, it is desirable to have a common baseline for the management objects allowing administrators, security architects and protocol designers to understand what management capabilities they can depend on in heterogeneous environments. Similarly, designing and deploying the protocol will be easier with thought paid to a common operational model. This will also help with the design of NetConf schemas or MIBs later.

2. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Breakdown of KARP configuration

There are multiple ways of structuring configuration information. One factor to consider is the scope of the configuration information. Several protocols are peer-to-peer routing protocols where a different key could potentially be used for each neighbor. Other protocols require the same group key to be used for all nodes in an administrative domain or routing area. In other cases, the same group key needs to be used for all routers on an interface, but different group keys can be used for each interface.
Within situations where a per-interface, per-area or per-peer key can be used for manually configured long-term keys, that flexibility may not be desirable from an operational standpoint. For example consider OSPF [RFC2328]. Each OSPF link needs to use the same authentication configuration, including the set of keys used for reception and the set of keys used for transmission, but may use different keys for different links. The most general management model would be to configure keys per link. However for deployments where the area uses the same key it would be strongly desirable to configure the key as a property of the area. If the keys are configured per-link, they can get out of sync. In order to support generality of configuration and common operational situations, it would be desirable to have some sort of inheritance where default configurations are made per-area unless overridden per-interface.
As described in [I-D.housley-saag-crypto-key-table], the cryptographic keys are separated from the interface configuration into their own configuration store. This document should specify how key selection interacts with the key table. One possible approach would be to assume that all keys that permit use on a given interface would be used on that interface with no additional configuration steps. If this model is adopted then the key table draft should be expanded to permit specification of domains and areas as well. It's not clear why "all" is

permitted as an interface specification in this model; it seems
unlikely that it would be desirable to use the same set of keys for two
different instances of an IGP or across autonomous system boundaries.
Another model is that the interface specification in the key table is a
restriction that limits keys on top of other configuration enabling
them. Then a set of keys from the key table is attached to an
interface, area or routing domain using an additional configuration
step. This avoids the previous problems at the expense of significant
complexity of configuration.
Operational Requirements: KARP MUST support configuration of keys at
the most general scope for the underlying protocol; protocols
supporting per-peer keys MUST permit configuration of per-peer keys,
protocols supporting per-interface keys MUST support configuration of
per-interface keys, and so on. KARP MUST NOT permit configuration of an
inappropriate key scope. For example, configuration of separate keys
per interface MUST NOT be supported for a protocol requiring per-area
keys.

3.1. Integrity of the Key Table

The routing key table [I-D.housley-saag-crypto-key-table] provides a
very general mechanism to abstract the storage of keys for routing
protocols. To avoid misconfiguration and simplify problem
determination, the router MUST verify the internal consistency of
entries added to the table. At a minimum, the router MUST verify:

   *The cryptographic algorithms are valid for the protocol.

   *The key derivation function is valid for the protocol.

   *The direction is valid for the protocol; for example protocols
    that require the same session key be used in both directions MUST
    have a direction of both.

   *The peer and interface specification is consistent with the
    protocol.

Other checks are possible. For example the router could verify that if
a key is associated with a peer, that peer is a configured peer for the
specified protocol. However, this may be undesirable. It may be
desirable to load a key table when some peers have not yet been
configured. Also, it may be desirable to share portions of a key table
across devices even when their current configuration does not require
an adjacency with a particular peer in the interest of uniform
configuration or preparing for fail-over.

3.2. Management of Key Table

Several management operations will be quite common. For service
provider deployments the configuration management system can simply

update the key table. However, for smaller deployments, efficient management operations are important.

As part of adding a new key it is typically desirable to set an expiration time for an old key. The management interface SHOULD provide a mechanism to easily update the expiration time for a current key used with a given peer or interface. Also when adding a key it is desirable to push the key out to nodes that will need it, allowing use for receiving packets then later enabling transmit. This can be accomplished automatically by providing a delay between when a key becomes valid for reception and transmission. However, some environments may not be able to predict when all the necessary changes will be made. In these cases having a mechanism to enable a key for sending is desirable.

3.3. Protocol Limitations from the Key Table

The format of the key table imposes a few limitations on routing protocols. The first is that the key ID is 16 bits; some routing protocols have 32-bit key identifiers. A key mapping table as discussed in 4.1 of [I-D.polk-saag-rtg-auth-keytable] could be used to map to the larger key identifier. However it's probably desirable to either decide that only 16 bits of the key ID space is to be used or to expand the identifier space in the key table. From a management standpoint we need to make concrete requirements around whether a key ID is per-protocol or whether subspaces in the key ID space are reserved for each protocol. This is necessary so that implementations from different vendors can be managed consistently.

The second requirement that the key table places is that the key ID is scoped fairly broadly. At least within some protocols such as OSPF, the key ID might only need to be unique per-link or per-peer. That is, packets sent on two different interfaces could use key ID 32 even if the keys were different for these interfaces. An implementation could use the interface and the key ID as a lookup to find the right key. However, the key table draft requires that a key ID be sufficient to look up a key, meaning that the key ID is a globally scoped identifier. There is nothing wrong with this restriction, but it does need to be noted when assigning key IDs for a domain.

Consideration is required for how an automated key management protocol will assign key IDs for group keys. All members of the group may need to use the same key ID. This requires careful coordination of global key IDs. Interactions with the peer key ID field may make this easier; this requires additional study.

Automated key management protocols also assign keys for single peers. If the key ID is global and needs to be coordinated between the receiver and transmitter, then there is complexity in key management protocols.

3.4. VRFs

Many core and enterprise routers support multiple routing instances.
For example a router serving multiple VPNs is likely to have a
forwarding/routing instance for each of these VPNs. We need to decide
how the key table and other configuration information for KARP
interacts with this. The obvious first-order answer is that each
routing instance gets its own key table. However, we need to consider
how these instances interact with each other and confirm this makes
sense.

4. Credentials and Authorization

Several methods for authentication have been proposed for KARP. The
simplest is preshared keys used directly as traffic keys. In this mode,
the traffic integrity keys are directly configured. This is the mode
supported by today's routing protocols.
As discussed in [I-D.polk-saag-rtg-auth-keytable], preshared keys can
be used as the input to a key derivation function (KDF) to generate
traffic keys. For example the TCP Authentication Option (TCP-AO)
[RFC5925] derives keys based on the initial TCP session state.
Typically a KDF will combine a long-term key with public inputs
exchanged as part of the protocol to form fresh session keys. a KDF
could potentially be used with some inputs that are configured along
with the long-term key. Also, it's possible that inputs to a KDF will
be private and exchanged as part of the protocol, although this will be
uncommon in KARP's uses of KDFs.
Preshared keys could also be used by an automated key management
protocol. In this mode, preshared keys would be used for
authentication. However traffic keys would be generated by some key
agreement mechanism or transported in a key encryption key derived from
the preshared key. This mode may provide better replay protection.
Also, in the absence of active attackers, key agreement strategies such
as Diffie-Hellman can be used to produce high-quality traffic keys even
from relatively weak preshared keys.
Public keys can be used for authentication. The design guide [I-D.ietf-
karp-design-guide] describes a mode in which routers have the hashes of
peer routers' public keys. In this mode, a traditional public-key
infrastructure is not required. The advantage of this mode is that a
router only contains its own keying material, limiting the scope of a
compromise. The disadvantage is that when a router is added or deleted
from the set of authorized routers, all routers that peer need to be
updated. Note that self-signed certificates are a common way of
communicating public-keys in this style of authentication.
Certificates signed by a certification authority or some other PKI
could be used. The advantage of this approach is that routers may not
need to be directly updated when peers are added or removed. The
disadvantage is that more complexity and cost is required.

Each of these approaches has a different set of management and operational requirements. Key differences include how authorization is handled and how identity works. This section discusses these differences.

4.1. Preshared Keys

In the protocol, manual preshared keys are either unnamed or named by a small integer (typically 16 or 32 bits) key ID. Implementations that support multiple keys for protocols that have no names for keys need to try all possible keys before deciding a packet cannot be validated [RFC4808]. Typically key IDs are names used by one group or peer. Manual preshared keys are often known by a group of peers rather than just oneother peer. This is an interesting security property: unlike with digitally signed messages or protocols where symmetric keys are known only to two parties, it is impossible to identify the peer sending a message cryptographically. However, it is possible to show that the sender of a message is one of the parties who knows the preshared key. Within the routing threat model the peer sending a message can be identified only because peers are trusted and thus can be assumed to correctly label the packets they send. This contrasts with a protocol where cryptographic means such as digital signatures are used to verify the origin of a message. As a consequence, authorization is typically based on knowing the preshared key rather than on being a particular peer. Note that once an authorization decision is made, the peer can assert its identity; this identity is trusted just as the routing information from the peer is trusted. Doing an additional check for authorization based on the identity included in the packet would provide little value: an attacker who somehow had the key could claim the identity of an authorized peer and an attacker without the key should be unable to claim the identity of any peer. Such a check is not required by the KARP threat model: inside attacks are not in scope.
Preshared keys used with key derivation function similarly to manual preshared keys. However to form the actual traffic keys, session or peer specific information is combined with the key. From an authorization standpoint, the derivation key works the same as a manual key. An additional routing protocol step or transport step forms the key that is actually used.
Preshared keys that are used via automatic key management have not been specified for KARP. Their naming and authorization may differ from existing uses of preshared keys in routing protocols. In particular, such keys may end up being known only by two peers. Alternatively they may also be known by a group of peers. Authorization could potentially be based on peer identity, although it is likely that knowing the right key will be sufficient. There does not appear to be a compelling reason to decouple the authorization of a key for some purpose from authorization of peers holding that key to perform the authorized function.

Care needs to be taken when symmetric keys are used for multiple purposes. Consider the implications of using the same preshared key for two interfaces: it becomes impossible to cryptographically distinguish a router on one interface from a router on another interface. So, a router that is trusted to participate in a routing protocol on one interface becomes implicitly trusted for the other interfaces that share the key. For many cases, such as link-state routers in the same routing area, there is no significant advantage that an attacker could gain from this trust within the KARP threat model. However, distance-vector protocols, such as BGP and RIP, permit routes to be filtered across a trust boundary. For these protocols, participation in one interface might be more advantageous than another. Operationally, when this trust distinction is important to a deployment, different keys need to be used on each side of the trust boundary. Key derivation can help prevent this problem in cases of accidental misconfiguration. However, key derivation cannot protect against a situation where a system was incorrectly trusted to have the key used to perform the derivation. To the extent that there are multiple zones of trust and a routing protocol is determining whether a particular router is within a certain zone, the question of untrusted actors is within the scope of the routing threat model.

Key derivation can be part of a management solution to a desire to have multiple keys for different zones of trust. A master key could be combined with peer, link or area identifiers to form a router-specific preshared key that is loaded onto routers. Provided that the master key lives only on the management server and not the individual routers, trust is preserved. However in many cases, generating independent keys for the routers and storing the result is more practical. If the master key were somehow compromised, all the resulting keys would need to be changed. However if independent keys are used, the scope of a compromise may be more limited.

More subtle problems with key separation can appear in protocol design. Two protocols that use the same traffic keys may work together in unintended ways permitting one protocol to be used to attack the other. Consider two hypothetical protocols. Protocol A starts its messages with a set of extensions that are ignored if not understood. Protocol B has a fixed header at the beginning of its messages but ends messages with extension information. It may be that the same message is valid both as part of protocol A and protocol B. An attacker may be able to gain an advantage by getting a router to generate this message with one protocol under situations where the other protocol would not generate the message. This hypothetical example is overly simplistic; real-world attacks exploiting key separation weaknesses tend to be complicated and involve specific properties of the cryptographic functions involved. The key point is that whenever the same key is used in multiple protocols, attacks may be possible. All the involved protocols need to be analyzed to understand the scope of potential attacks.

Key separation attacks interact with the KARP operational model in a number of ways. Administrators need to be aware of situations where

using the same manual traffic key with two different protocols (or the same protocol in different contexts) creates attack opportunities. Design teams should consider how their protocol might interact with other routing protocols and describe any attacks discovered so that administrators can understand the operational implications. When designing automated key management or new cryptographic authentication within routing protocols, we need to be aware that administrators expect to be able to use the same preshared keys in multiple contexts. As a result, we should use appropriate key derivation functions so that different cryptographic keys are used even when the same initial input key is used.

4.2. Asymmetric Keys

Outside of a PKI, public keys are expected to be known by the hash of a key or (potentially self-signed) certificate. The Session Description Protocol provides a standardized mechanism for naming keys (in that case certificates) based on hashes (section 5 [RFC4572]). KARP SHOULD adopt this approach or another approach already standardized within the IETF rather than inventing a new mechanism for naming public keys.
A public key is typically expected to belong to one peer. As a peer generates new keys and retires old keys, its public key may change. For this reason, from a management standpoint, peers should be thought of as associated with multiple public keys rather than as containing a single public key hash as an attribute of the peer object.
Authorization of public keys could be done either by key hash or by peer identity. Performing authorizations by peer identity should make it easier to update the key of a peer without risk of losing authorizations for that peer. However management interfaces need to be carefully designed to avoid making this extra level of indirection complicated for operators.

4.3. Public Key Infrastructure

When a PKI is used, certificates are used. The certificate binds a key to a name of a peer. The key management protocol is responsible for exchanging certificates and validating them to a trust anchor. Authorization needs to be done in terms of peer identities not in terms of keys. One reason for this is that when a peer changes its key, the new certificate needs to be sufficient for authentication to continue functioning even though the key has never been seen before.
Potentially authorization could be performed in terms of groups of peers rather than single peers. An advantage of this is that it may be possible to add a new router with no authentication related configuration of the peers of that router. For example, a domain could decide that any router with a particular keyPurposeID signed by the organization's certificate authority is permitted to join the IGP. Just as in configurations where cryptographic authentication is not used,

automatic discovery of this router can establish appropriate
adjacencies.
Assuming that potentially self-signed certificates are used by routers
that wish to use public keys but that do not need a PKI, then PKI and
the infrastructureless mode of public-key operation described in the
previous section can work well together. One router could identify its
peers based on names and use certificate validation. Another router
could use hashes of certificates. This could be very useful for border
routers between two organizations. Smaller organizations could use
public keys and larger organizations could use PKI.

4.4. The role of Central Servers

An area to explore is the role of central servers like RADIUS or
directories. As discussed in the design-guide, a system where keys are
pushed by a central management system is undesirable as an end result
for KARP. However central servers may play a role in authorization and
key rollover. For example a node could send a hash of a public key to a
RADIUS server.
If central servers do play a role it will be critical to make sure that
they are not required during routine operation or a cold-start of a
network. They are more likely to play a role in enrollment of new peers
or key migration/compromise.
Another area where central servers may play a role is for group key
agreement. As an example, [I-D.liu-ospfv3-automated-keying-req]
discusses the potential need for key agreement servers in OSPF. Other
routing protocols that use multicast or broadcast such as IS-IS are
likely to need a similar approach.

5. Grouping Peers Together

One significant management consideration will be the grouping of
management objects necessary to determine who is authorized to act as a
peer for a given routing action. As discussed previously, the following
objects are potentially required:

   *Key objects are required. Symmetric keys may be preshared.
    Asymmetric public keys may be used directly for authorization as
    well. During key transitions more than one key may refer to a
    given peer. Group preshared keys may refer to multiple peers.

   *A peer is a router that this router might wish to communicate
    with. Peers may be identified by names or keys.

   *Groups of peers may be authorized for a given routing protocol.

Establishing a management model is difficult because of the complex
relationships between each set of objects. As discussed there may be
more than one key for a peer. However in the preshared key case, there
may be more than one peer for a key. This is true both for group

security association protocols such as an IGP or one-to-one protocols where the same key is used administratively. In some of these situations, it may be undesirable to explicitly enumerate the peers in the configuration; for example IGP peers are auto-discovered for broadcast links but not for non-broadcast multi-access links.

Peers may be identified either by name or key. If peers are identified by key it is probably strongly desirable from an operational standpoint to consider any peer identifiers or name to be a local matter and not require the names or identifiers to be synchronized. Obviously if peers are identified by names (for example with certificates in a PKI), identifiers need to be synchronized between the authorized peer and the peer making the authorization decision.

In many cases peers will explicitly be identified. In these cases it is possible to attach the authorization information (keys or identifiers) to the peer's configuration object. Two cases do not involve enumerating peers. The first is the case where preshared keys are shared among a group of peers. It is likely that this case can be treated from a management standpoint as a single peer representing all the peers that share the keys. The other case is one where certificates in a PKI are used to introduce peers to a router. In this case, rather than configuring peers, , the router needs to be configured with information on what certificates represent acceptable peers.

Another consideration is what routing protocols share peers. For example it may be common for LDP peers to also be peers of some other routing protocol. Also, RSVP-TE may be associated with some TE-based IGP. In some of these cases it would be desirable to use the same authorization information for both routing protocols.

In order to develop a management model for authorization, the working group needs to consider several questions. What protocols support auto-discovery of peers? What protocols require more configuration of a peer than simply the peer's authorization information and network address? What management operations are going to be common as security information for peers is configured and updated? What operations will be common while performing key transitions or while migrating to new security technologies?

6. Administrator Involvement

One key operational question is what areas will administrator involvement be required. Likely areas where involvement may be useful includes enrollment of new peers. Fault recovery should also be considered.

6.1. Enrollment

One area where the management of routing security needs to be optimized is the deployment of a new router. In some cases a new router may be deployed on an existing network where routing to management servers is already available. In other cases, routers may be deployed as part of

connecting or creating a site. Here, the router and infrastructure may
not be available until the router has securely authenticated. This
problem is similar to the problem of getting initial configuration of
routing instances onto the router. However, especially in cases where
asymmetric keys or per-peer preshared keys are used, the configuration
of other routers needs to be modified to bring up the security
association. Also, there has been discussion of generating keys on
routers and not allowing them to leave devices. This also impacts what
strategies are possible. For example this might mean that routers need
to be booted in a secure environment where keys can be generated, and
public keys copied to a management server to push out the new public
key to potential peers. Then, the router needs to be packaged, moved to
where it will be deployed and set up.Alternatives are possible; it is
critical that we understand how what we propose impacts operators.
We need to work through examples with operators familiar with specific
real-world deployment practices and understand how proposed security
mechanisms will interact with these practices.

6.2. Handling Faults

Faults may interact with operational practice in at least two ways.
First, security solutions may introduce faults. For example if
certificates expire in a PKI, previous adjacencies may no longer form.
Operational practice will require a way of repairing these errors. This
may end up being very similar to deploying a router that is connecting
a new site as the security fault may have partitioned the network.
However, unlike a new deployment, the event is unplanned. Strategies
such as configuring a router and shipping it to a site may not be
appropriate for recovering a fault even though they may be more useful
for new deployments.
Monitoring will play a critical role in avoiding security faults such
as certificate expiration. However, the protocols MUST still have
adequate operational mechanisms to recover from these situations. Also,
some faults, such as those resulting from a compromise or actual attack
on a facility are inherent and may not be prevented.
A second class of faults is equipment faults that impact security. For
example if keys are stored on a router and never moved from that
device, failure of a router implies a need to update security
provisioning on the replacement router and its peers.
To address these operational considerations, we should identify
circumstances surrounding recovery from today's faults and understand
how protocols will impact mechanisms used today.

7. Upgrade Considerations

It needs to be possible to deploy automated key management in an
organization without either having to disable existing security or
disrupting routing. As a result, it needs to be possible to perform a
phased upgrade from manual keying to automated key management.

For peer-to-peer protocols such as BGP, this is likely to be relatively easy. First, code that supports automated key management needs to be loaded on both peers. Then the adjacency can be upgraded. The configuration can be updated to switch to automated key management when the second router reboots.

The situation is more complicated for multicast protocols. It's probably not reasonable to bring down an entire link to reconfigure it as using automated key management. Two approaches should be considered. One is to support key table rows from the automated key management and manually configured for the same link at the same time. Coordinating this may be tricky. Another possibility is for the automated key management protocol to actually select the same traffic key that is being used manually

8. Related Work

Discuss draft-housley-saag-*, draft-polk-saag-*, the discussions in the KARP framework, etc.

9. Security Considerations

This document does not define a protocol. It does discuss the operational and management implications of several security technologies.

10. Acknowledgments

Funding for Sam Hartman's work on this memo is provided by Huawei. The authors would like to thank Gregory Lebovitz, Russ White and Bill Atwood for valuable reviews.

11. References

11.1. Normative References

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[I-D.housley-saag-crypto-key-table]	Housley, R and T Polk, "Database of Long-Lived Symmetric Cryptographic Keys", Internet-Draft draft-housley-saag-crypto-key-table-04, October 2010.

11.2. Informative References

[I-D.liu-ospfv3-automated-keying-req]	Liu, Y, "OSPFv3 Automated Group Keying Requirements", Internet-Draft draft-liu-ospfv3-automated-keying-req-01, July 2007.
	Lebovitz, G and M Bhatia, "Keying and Authentication for Routing Protocols (KARP)

[I-D.ietf-karp-design-guide]	Design Guidelines", Internet-Draft draft-ietf-karp-design-guide-08, November 2011.
[RFC2328]	Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.
[RFC4808]	Bellovin, S., "Key Change Strategies for TCP-MD5", RFC 4808, March 2007.
[RFC4572]	Lennox, J., "Connection-Oriented Media Transport over the Transport Layer Security (TLS) Protocol in the Session Description Protocol (SDP)", RFC 4572, July 2006.
[I-D.polk-saag-rtg-auth-keytable]	Polk, T and R Housley, "Routing Authentication Using A Database of Long-Lived Cryptographic Keys", Internet-Draft draft-polk-saag-rtg-auth-keytable-05, November 2010.
[RFC5925]	Touch, J., Mankin, A. and R. Bonica, "The TCP Authentication Option", RFC 5925, June 2010.

Authors' Addresses

Sam Hartman Hartman Painless Security EMail: hartmans-ietf@mit.edu

Dacheng Zhang Zhang Huawei EMail: zhangdacheng@huawei.com