Network Working Group
Request for Comments: 4322
Category: Informational

M. Richardson
SSW
D.H. Redelmeier
Mimosa

December 2005

Opportunistic Encryption using the Internet Key Exchange (IKE)

Status of This Memo

Copyright Notice

Abstract

This document describes opportunistic encryption (OE) as designed and implemented by the Linux FreeS/WAN project. OE uses the Internet Key Exchange (IKE) and IPsec protocols. The objective is to allow encryption for secure communication without any pre-arrangement specific to the pair of systems involved. DNS is used to distribute the public keys of each system involved. This is resistant to passive attacks. The use of DNS Security (DNSSEC) secures this system against active attackers as well.

As a result, the administrative overhead is reduced from the square of the number of systems to a linear dependence, and it becomes possible to make secure communication the default even when the partner is not known in advance.

Table of Contents

1. Introduction 3
   1.1. Motivation 3
   1.2. Encryption Regimes 4
   1.3. Peer Authentication in Opportunistic Encryption 4
   1.4. Use of RFC 2119 Terms 5
2. Overview 6
   2.1. Reference Diagram 6
   2.2. Terminology 6
   2.3. Model of Operation 8

3. Protocol Specification 9
3.1. Forwarding Plane State Machine 9
3.2. Keying Daemon -- Initiator 12
3.3. Keying Daemon -- Responder 20
3.4. Renewal and Teardown 22
4. Impacts on IKE 24
4.1. ISAKMP/IKE Protocol 24
4.2. Gateway Discovery Process 24
4.3. Self Identification 24
4.4. Public Key Retrieval Process 25
4.5. Interactions with DNSSEC 25
4.6. Required Proposal Types 25
5. DNS Issues 26
5.1. Use of KEY Record 26
5.2. Use of TXT Delegation Record 27
5.3. Use of FQDN IDs 29
5.4. Key Roll-Over 29
6. Network Address Translation Interaction 30
6.1. Co-Located NAT/NAPT 30
6.2. Security Gateway behind a NAT/NAPT 30
6.3. End System behind a NAT/NAPT 31
7. Host Implementations 31
8. Multi-Homing 31
9. Failure Modes 33
9.1. DNS Failures 33
9.2. DNS Configured, IKE Failures 33
9.3. System Reboots 34
10. Unresolved Issues 34
10.1. Control of Reverse DNS 34
11. Examples 34
11.1. Clear-Text Usage (Permit Policy) 34
11.2. Opportunistic Encryption 36
12. Security Considerations 39
12.1. Configured versus Opportunistic Tunnels 39
12.2. Firewalls versus Opportunistic Tunnels 40
12.3. Denial of Service 41
13. Acknowledgements 41
14. References 41
14.1. Normative References 41
14.2. Informative References 42

1. Introduction

1.1. Motivation

The objective of opportunistic encryption is to allow encryption without any pre-arrangement specific to the pair of systems involved. Each system administrator adds public key information to DNS records to support opportunistic encryption and then enables this feature in the nodes' IPsec stack. Once this is done, any two such nodes can communicate securely.

This document describes opportunistic encryption as designed and implemented by the Linux FreeS/WAN project in revisions up and including 2.00. Note that 2.01 and beyond implements [RFC3445] in a backward compatible way. A future document [IPSECKEY] will describe a variation that complies with RFC 3445. For project information, see http://www.freeswan.org.

The Internet Architecture Board (IAB) and Internet Engineering Steering Group (IESG) have taken a strong stand that the Internet should use powerful encryption to provide security and privacy [RFC1984]. The Linux FreeS/WAN project attempts to provide a practical means to implement this policy.

The project uses the IPsec, ISAKMP/IKE, DNS, and DNSSEC protocols because they are standardized, widely available, and can often be deployed very easily without changing hardware or software, or retraining users.

The extensions to support opportunistic encryption are simple. No changes to any on-the-wire formats are needed. The only changes are to the policy decision making system. This means that opportunistic encryption can be implemented with very minimal changes to an existing IPsec implementation.

Opportunistic encryption creates a "fax effect". The proliferation of the fax machine was possible because it did not require that everyone buy one overnight. Instead, as each person installed one, the value of having one increased because there were more people that could receive faxes. Once opportunistic encryption is installed, it automatically recognizes other boxes using opportunistic encryption, without any further configuration by the network administrator. So, as opportunistic encryption software is installed on more boxes, its value as a tool increases.

This document describes the infrastructure to permit deployment of Opportunistic Encryption.

The term S/WAN is a trademark of RSA Data Systems, and is used with permission by this project.

1.2. Encryption Regimes

To aid in understanding the relationship between security processing and IPsec, we divide policies controlling network traffic into four categories. The traffic is categorized by destination address using longest prefix match. Therefore, each category is enumerated by a set of network prefixes. The categories are mutually exclusive; a particular prefix should only occur in one category.

* Deny: network prefixes to which traffic is always forbidden.
* Permit: network prefixes to which traffic in the clear is permitted.
* Opportunistic tunnel: network prefixes to which traffic is encrypted if possible, when it otherwise might be sent in the clear.
* Configured tunnel: network prefixes to which traffic must be encrypted, and traffic in the clear is never permitted. A traditionally defined Virtual Private Network (VPN) is a form of configured tunnel.

Traditional firewall devices handle the first two categories. No authentication is required. The permit policy is currently the default on the Internet.

This document describes the third category: opportunistic tunnel, which is proposed as the new default for the Internet.

Category four's policy is a very strict "encrypt it or drop it" policy, which requires authentication of the endpoints. As the number of endpoints is typically bounded and is typically under a single authority, arranging for distribution of authentication material, while difficult, does not require any new technology. The mechanism described here, however, does provides an additional way to distribute the authentication materials; it is a public key method that does not require deployment of an X.509 based infrastructure.

1.3. Peer Authentication in Opportunistic Encryption

Opportunistic encryption creates tunnels between nodes that are essentially strangers. This is done without any prior bilateral arrangement. Therefore, there is the difficult question of how one knows to whom one is talking.

One possible answer is that since no useful authentication can be done, none should be tried. This mode of operation is named "anonymous encryption". An active man-in-the-middle attack can be used to thwart the privacy of this type of communication. Without peer authentication, there is no way to prevent this kind of attack.

Although it is a useful mode, anonymous encryption is not the goal of this project. Simpler methods are available that can achieve anonymous encryption only, but authentication of the peer is a desirable goal. Authentication of the peer is achieved through key distribution in DNS, leveraging upon the authentication of the DNS in DNSSEC.

Peers are, therefore, authenticated with DNSSEC when available. Local policy determines how much trust to extend when DNSSEC is not available.

An essential premise of building private connections with strangers is that datagrams received through opportunistic tunnels are no more special than datagrams that arrive in the clear. Unlike in a VPN, these datagrams should not be given any special exceptions when it comes to auditing, further authentication, or firewalling.

When initiating outbound opportunistic encryption, local configuration determines what happens if tunnel setup fails. The packet may go out in the clear, or it may be dropped.

1.4. Use of RFC 2119 Terms

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [RFC2119]

2. Overview

2.1. Reference Diagram

The following network diagram is used in the rest of this document as the canonical diagram:

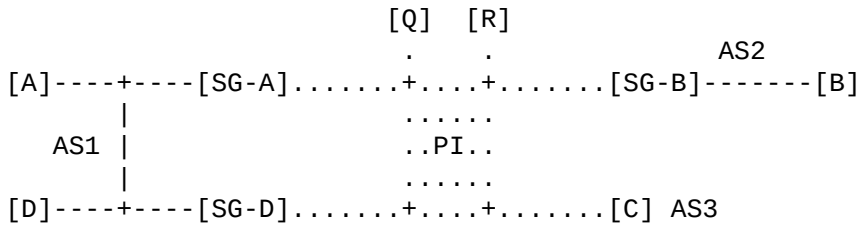


Figure 1: Reference Network Diagram

In this diagram, there are four end-nodes: A, B, C, and D. There are three security gateways, SG-A, SG-B, SG-D. A, D, SG-A, and SG-D are part of the same administrative authority, AS1. SG-A and SG-D are on two different exit paths from organization 1. SG-B and B are part of an independent organization, AS2. Nodes Q and R are nodes on the Internet. PI is the Public Internet ("The Wild").

2.2. Terminology

Note: The network numbers used in this document are for illustrative purposes only. This document could not use the reserved example network numbers of [RFC3330] because multiple address ranges were needed.

The following terminology is used in this document:

Security gateway (or simply gateway): a system that performs IPsec tunnel mode encapsulation/decapsulation. [SG-x] in the diagram.

Alice: node [A] in the diagram. When an IP address is needed, this is 192.1.0.65.

Bob: node [B] in the diagram. When an IP address is needed, this is 192.2.0.66.

Carol: node [C] in the diagram. When an IP address is needed, this is 192.1.1.67.

Dave: node [D] in the diagram. When an IP address is needed, this is 192.3.0.68.

SG-A: Alice's security gateway. Internally it is 192.1.0.1, externally it is 192.1.1.4.

SG-B: Bob's security gateway. Internally it is 192.2.0.1, externally it is 192.1.1.5.

SG-D: Dave's security gateway. Also Alice's backup security gateway. Internally it is 192.3.0.1, externally it is 192.1.1.6.

Configured tunnel: a tunnel that is directly and deliberately hand-configured on participating gateways. Configured tunnels are typically given a higher level of trust than opportunistic tunnels.

Road warrior tunnel: a configured tunnel connecting one node with a fixed IP address and one node with a variable IP address. A road warrior (RW) connection must be initiated by the variable node, since the fixed node cannot know the current address for the road warrior.

Anonymous encryption: the process of encrypting a session without any knowledge of who the other parties are. No authentication of identities is done.

Opportunistic encryption: the process of encrypting a session with authenticated knowledge of who the other party is without prearrangement.

Lifetime: the period in seconds (bytes or datagrams) for which a security association will remain alive before rekeying is needed.

Lifespan: the effective time for which a security association remains useful. A security association with a lifespan shorter than its lifetime would be removed when no longer needed. A security association with a lifespan longer than its lifetime would need to be re-keyed one or more times.

Phase 1 SA: an ISAKMP/IKE security association sometimes referred to as a keying channel.

Phase 2 SA: an IPsec security association.

Tunnel: another term for a set of phase 2 SA (one in each direction).

NAT: Network Address Translation (see [RFC2663]).

NAPT: Network Address and Port Translation (see [RFC2663]).

AS: an autonomous system.

FQDN: Fully-Qualified Domain Name

Default-free zone: a set of routers that maintain a complete set of routes to all currently reachable destinations. Having such a list, these routers never make use of a default route. A datagram with a destination address not matching any route will be dropped by such a router.

2.3. Model of Operation

The opportunistic encryption security gateway (OE gateway) is a regular gateway node, as described in [RFC0791] section 2.4 and [RFC1812], with the additional capabilities described here and in [RFC2401]. The algorithm described here provides a way to determine, for each datagram, whether or not to encrypt and tunnel the datagram. Two important things that must be determined are whether or not to encrypt and tunnel and, if so, the destination address or name of the tunnel endpoint that should be used.

2.3.1. Tunnel Authorization

The OE gateway determines whether or not to create a tunnel based on the destination address of each packet. Upon receiving a packet with a destination address not recently seen, the OE gateway performs a lookup in DNS for an authorization resource record (see Section 5.2). The record is located using the IP address to perform a search in the in-addr.arpa (IPv4) or ip6.arpa (IPv6) maps. If an authorization record is found, the OE gateway interprets this as a request for a tunnel to be formed.

2.3.2. Tunnel Endpoint Discovery

The authorization resource record also provides the address or name of the tunnel endpoint that should be used.

The record may also provide the public RSA key of the tunnel end point itself. This is provided for efficiency only. If the public RSA key is not present, the OE gateway performs a second lookup to find a KEY resource record for the endpoint address or name.

Origin and integrity protection of the resource records is provided by DNSSEC (see [RFC4033]). Section 3.2.4.1 documents an optional restriction on the tunnel endpoint if DNSSEC signatures are not available for the relevant records.

2.3.3. Caching of Authorization Results

The OE gateway maintains a cache, in the forwarding plane, of source/destination pairs for which opportunistic encryption has been attempted. This cache maintains a record of whether or not OE was successful so that subsequent datagrams can be forwarded properly without additional delay.

Successful negotiation of OE instantiates a new security association. Failure to negotiate OE results in creation of a forwarding policy entry either to deny or permit transmission in the clear future datagrams. This negative cache is necessary to avoid the possibly lengthy process of repeatedly looking up the same information.

The cache is timed out periodically, as described in Section 3.4. This removes entries that are no longer being used and permits the discovery of changes in authorization policy.

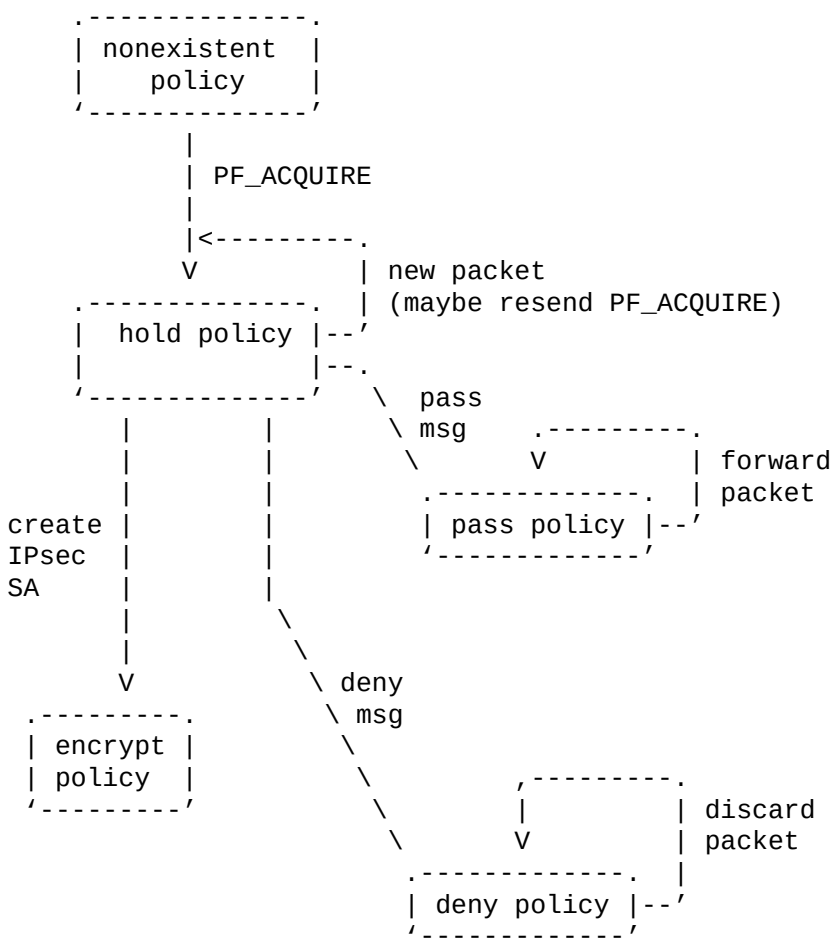
3. Protocol Specification

The OE gateway is modeled to have a forwarding plane and a control plane. A control channel, such as PF_KEY [RFC2367], connects the two planes.

The forwarding plane performs per-datagram operations. The control plane contains a keying daemon, such as ISAKMP/IKE, and performs all authorization, peer authentication, and key derivation functions.

3.1. Forwarding Plane State Machine

Let the OE gateway maintain a collection of objects -- a superset of the security policy database (SPD) specified in [RFC2401]. For each combination of source and destination address, an SPD object exists in one of five following states. Prior to forwarding each datagram, the responder uses the source and destination addresses to pick an entry from the SPD. The SPD then determines if and how the packet is forwarded.



3.1.1. Nonexistent Policy

If the gateway does not find an entry, then this policy applies. The gateway creates an entry with an initial state of "hold policy" and requests keying material from the keying daemon. The gateway does not forward the datagram; rather, it SHOULD attach the datagram to the SPD entry as the "first" datagram and retain it for eventual transmission in a new state.

3.1.2. Hold Policy

The gateway requests keying material. If the interface to the keying system is lossy (PF_KEY, for instance, can be), the implementation SHOULD include a mechanism to retransmit the keying request at a rate limited to less than 1 request per second. The gateway does not forward the datagram. The gateway SHOULD attach the datagram to the SPD entry as the "last" datagram, where it is retained for eventual

transmission. If there is a datagram already stored in this way, then that already-stored datagram is discarded.

The rationale behind saving the "first" and "last" datagrams are as follows: The "first" datagram is probably a TCP SYN packet. Once there is keying established, the gateway will release this datagram, avoiding the need for the endpoint to retransmit the datagram. In the case where the connection was not a TCP connection, but was instead a streaming protocol or a DNS request, the "last" datagram that was retained is likely the most recent data. The difference between "first" and "last" may also help the endpoints determine which data was dropped while negotiation took place.

3.1.3. Pass-Through Policy

The gateway forwards the datagram using the normal forwarding table. The gateway enters this state only by command from the keying daemon, and upon entering this state, also forwards the "first" and "last" datagrams.

3.1.4. Deny Policy

The gateway discards the datagram. The gateway enters this state only by command from the keying daemon, and upon entering this state, discards the "first" and "last" datagrams. An implementation MAY provide the administrator with a control to determine if further datagrams cause ICMP messages to be generated (i.e., ICMP Destination Unreachable, Communication Administratively Prohibited. type=3, code=13).

3.1.5. Encrypt Policy

The gateway encrypts the datagram using the indicated security association database (SAD) entry. The gateway enters this state only by command from the keying daemon, and upon entering this state, releases and forwards the "first" and "last" datagrams using the new encrypt policy.

If the associated SAD entry expires because of byte, packet or time limits, then the entry returns to the Hold policy, and an expire message is sent to the keying daemon.

All states may be created directly by the keying daemon while acting as a gateway.

3.2. Keying Daemon -- Initiator

Let the keying daemon maintain a collection of objects. Let them be called "connections" or "conn"s. There are two categories of connection objects: classes and instances. A class represents an abstract policy (i.e., what could be). An instance represents an actual connection (i.e., what is running at the time).

Let there be two further subtypes of connections: keying channels (Phase 1 SAs) and data channels (Phase 2 SAs). Each data channel object may have a corresponding SPD and SAD entry maintained by the datagram state machine.

For the purposes of opportunistic encryption, there MUST, at least, be connection classes known as "deny", "always-clear-text", "OE-permissive", and "OE-paranoid". The latter two connection classes define a set of destination prefixes for which opportunistic encryption will be attempted. The administrator MAY set policy options in a number of additional places. An implementation MAY create additional connection classes to further refine these policies.

The simplest system may need only the "OE-permissive" connection, and would list its own (single) IP address as the source address of this policy and the wild-card address 0.0.0.0/0 as the destination IPv4 address. That is, the simplest policy is to try opportunistic encryption with all destinations.

This simplest policy SHOULD be offered as a preconfigured default.

The distinction between permissive and paranoid Opportunistic Encryption ("OE-paranoid" below) use will become clear in the state transition differences.

In brief, an OE-permissive policy means to permit traffic to flow in the clear when there is a failure to find and/or use the encryption keys. OE-permissive permits the network to function, even if in an insecure manner.

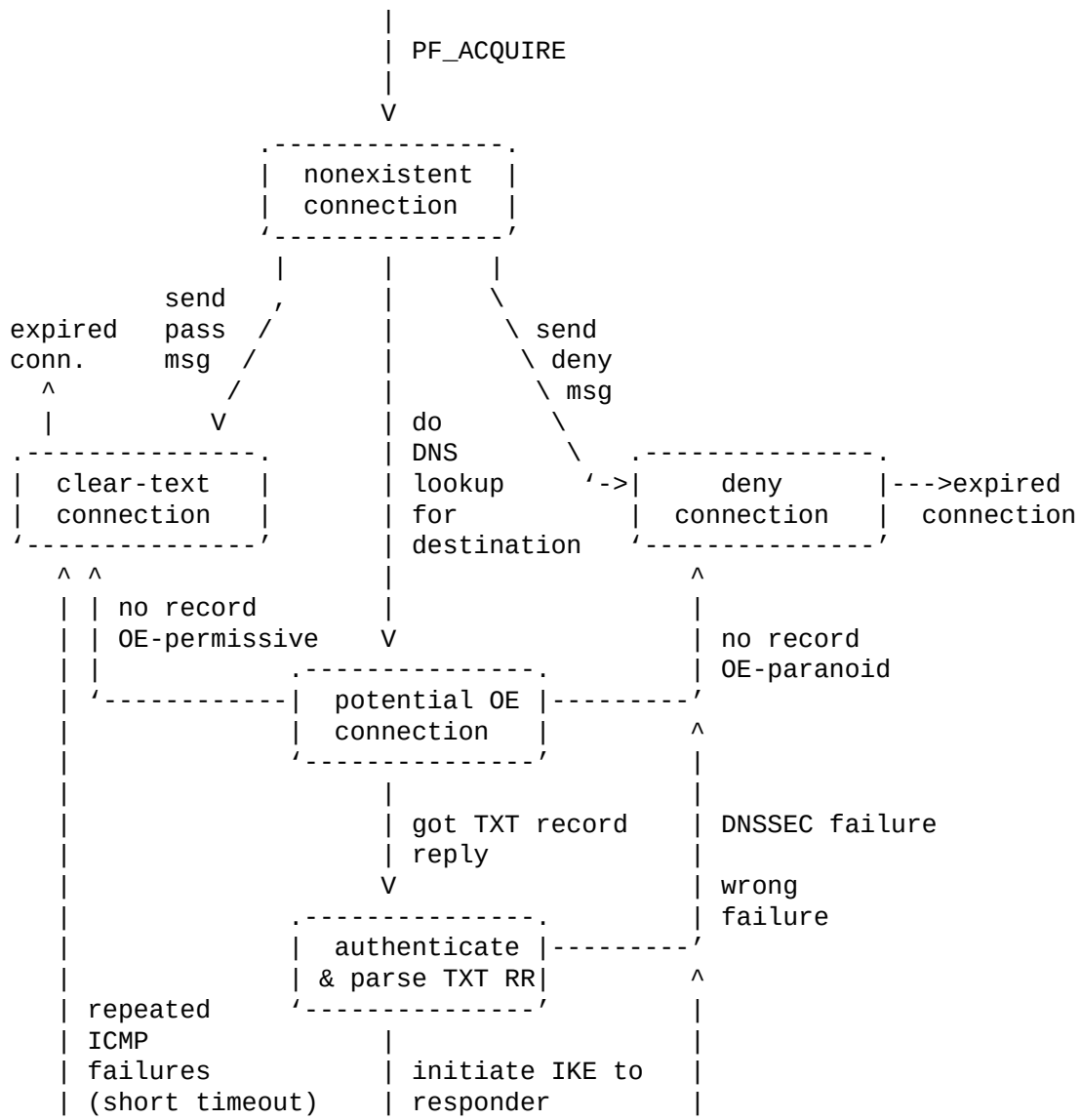
On failure, a paranoid OE ("OE-paranoid") will install a drop policy. OE-paranoid permits traffic to flow only when appropriate security is available.

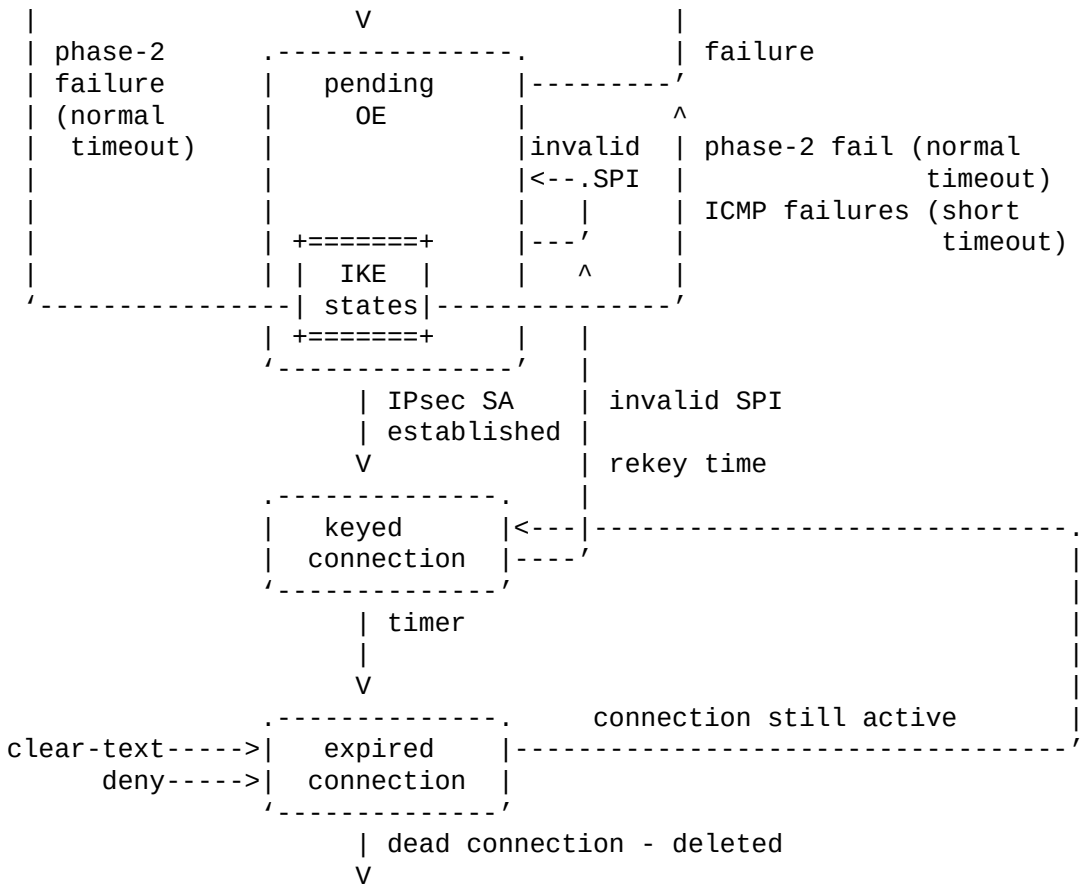
In this description of the keying machine’s state transitions, the states associated with the keying system itself are omitted because they are best documented in the keying system ([RFC2407], [RFC2408], and [RFC2409] for ISAKMP/IKE), and the details are keying system specific. Opportunistic encryption is not dependent upon any

specific keying protocol, but this document does provide requirements for those using ISAKMP/IKE to assure that implementations inter-operate.

The state transitions that may be involved in communicating with the forwarding plane are omitted. PF_KEY and similar protocols have their own set of states required for message sends and completion notifications.

Finally, the retransmits and recursive lookups that are normal for DNS are not included in this description of the state machine.





3.2.1. Nonexistent Connection

There is no connection instance for a given source/destination address pair. Upon receipt of a request for keying material for this source/destination pair, the initiator searches through the connection classes to determine the most appropriate policy. Upon determining an appropriate connection class, an instance object is created of that type. Both of the OE types result in a potential OE connection.

Failure to find an appropriate connection class results in an administrator-defined default.

In each case, when the initiator finds an appropriate class for the new flow, an instance connection is made of the class that matched.

3.2.2. Clear-Text Connection

The nonexistent connection makes a transition to this state when an always-clear-text class is instantiated, or when an OE-permissive connection fails. During the transition, the initiator creates a pass-through policy object in the forwarding plane for the appropriate flow.

Timing out is the only way to leave this state (see Section 3.2.7).

3.2.3. Deny Connection

The empty connection makes a transition to this state when a deny class is instantiated, or when an OE-paranoid connection fails. During the transition, the initiator creates a deny policy object in the forwarding plane for the appropriate flow.

Timing out is the only way to leave this state (see Section 3.2.7).

3.2.4. Potential OE Connection

The empty connection makes a transition to this state when one of either OE class is instantiated. During the transition to this state, the initiator creates a hold policy object in the forwarding plane for the appropriate flow.

In addition, when making a transition into this state, DNS lookup is done in the reverse-map for a TXT delegation resource record (see Section 5.2). The lookup key is the destination address of the flow.

There are three ways to exit this state:

1. DNS lookup finds a TXT delegation resource record.

2. DNS lookup does not find a TXT delegation resource record.

3. DNS lookup times out.

Based upon the results of the DNS lookup, the potential OE connection makes a transition to the pending OE connection state. The conditions for a successful DNS look are:

1. DNS finds an appropriate resource record.

2. It is properly formatted according to Section 5.2.

3. If DNSSEC is enabled, then the signature has been vouched for.

Note that if the initiator does not find the public key present in the TXT delegation record, then the public key must be looked up as a sub-state. Only successful completion of all the DNS lookups is considered a success.

If DNS lookup does not find a resource record or if DNS times out, then the initiator considers the receiver not OE capable. If this is an OE-paranoid instance, then the potential OE connection makes a transition to the deny connection state. If this is an OE-permissive instance, then the potential OE connection makes a transition to the clear-text connection state.

If the initiator finds a resource record, but it is not properly formatted, or if DNSSEC is enabled and reports a failure to authenticate, then the potential OE connection makes a transition to the deny connection state. This action SHOULD be logged. If the administrator wishes to override this transition between states, then an always-clear class can be installed for this flow. An implementation MAY make this situation a new class.

3.2.4.1. Restriction on Unauthenticated TXT Delegation Records

An implementation SHOULD also provide an additional administrative control on delegation records and DNSSEC. This control would apply to delegation records (the TXT records in the reverse-map) that are not protected by DNSSEC. Records of this type are only permitted to delegate to their own address as a gateway. When this option is enabled, an active attack on DNS will be unable to redirect packets to other than the original destination.

3.2.5. Pending OE Connection

The potential OE connection makes a transition to this state when the initiator determines that all the information required from the DNS lookup is present. Upon entering this state, the initiator attempts to initiate keying to the gateway provided.

Exit from this state occurs with either a successfully created IPsec SA or a failure of some kind. Successful SA creation results in a transition to the key connection state.

Three failures have caused significant problems. They are clearly not the only possible failures from keying.

Note that if there are multiple gateways available in the TXT delegation records, then a failure can only be declared after all of them have been tried. Further, creation of a phase 1 SA does not constitute success. A set of phase 2 SAs (a tunnel) is considered success.

The first failure occurs when an ICMP port unreachable is consistently received without any other communication, or when there is silence from the remote end. This usually means that either the gateway is not alive, or the keying daemon is not functional. For an OE-permissive connection, the initiator makes a transition to the clear-text connection, but with a low lifespan. For an OE-pessimistic connection, the initiator makes a transition to the deny connection again with a low lifespan. The lifespan in both cases is kept low because the remote gateway may be in the process of rebooting or be otherwise temporarily unavailable.

The length of time to wait for the remote keying daemon to wake up is a matter of some debate. If there is a routing failure, 5 minutes is usually long enough for the network to re-converge. Many systems can reboot in that amount of time as well. However, 5 minutes is far too long for most users to wait to hear that they can not connect using OE. Implementations SHOULD make this a tunable parameter.

The second failure occurs after a phase 1 SA has been created, but there is either no response to the phase 2 proposal, or the initiator receives a negative notify (the notify must be authenticated). The remote gateway is not prepared to do OE at this time. As before, the initiator makes a transition to the clear-text or the deny connection based upon connection class, but this time with a normal lifespan.

The third failure occurs when there is signature failure while authenticating the remote gateway. This can occur when there has been a key roll-over, but DNS has not caught up. In this case again, the initiator makes a transition to the clear-text or the deny connection based upon the connection class. However, the lifespan depends upon the remaining time to live in the DNS. (Note that DNSSEC signed resource records have a different expiry time from non-signed records.)

3.2.6. Keyed Connection

The pending OE connection makes a transition to this state when session keying material (the phase 2 SAs) is derived. The initiator creates an encrypt policy in the forwarding plane for this flow.

There are three ways to exit this state. The first is by receipt of an authenticated delete message (via the keying channel) from the peer. This is normal teardown and results in a transition to the expired connection state.

The second exit is by expiry of the forwarding plane keying material. This starts a re-key operation with a transition back to pending OE connection. In general, the soft expiry occurs with sufficient time left to continue using the keys. A re-key can fail, which may result in the connection failing to clear-text or deny as appropriate. In the event of a failure, the forwarding plane policy does not change until the phase 2 SA (IPsec SA) reaches its hard expiry.

The third exit is in response to a negotiation from a remote gateway. If the forwarding plane signals the control plane that it has received an unknown SPI from the remote gateway, or an ICMP is received from the remote gateway indicating an unknown SPI, the initiator should consider that the remote gateway has rebooted or restarted. Since these indications are easily forged, the implementation must exercise care. The initiator should make a cautious (rate-limited) attempt to re-key the connection.

3.2.7. Expiring Connection

The initiator will periodically place each of the deny, clear-text, and keyed connections into this sub-state. See Section 3.4 for more details of how often this occurs. The initiator queries the forwarding plane for last use time of the appropriate policy. If the last use time is relatively recent, then the connection returns to the previous deny, clear-text or keyed connection state. If not, then the connection enters the expired connection state.

The DNS query and answer that lead to the expiring connection state are also examined. The DNS query may become stale. (A negative, i.e., no such record, answer is valid for the period of time given by the MINIMUM field in an attached SOA record. See [RFC1034] section 4.3.4.) If the DNS query is stale, then a new query is made. If the results change, then the connection makes a transition to a new state as described in potential OE connection state.

Note that when considering how stale a connection is, both outgoing SPD and incoming SAD must be queried as some flows may be unidirectional for some time.

Also note that the policy at the forwarding plane is not updated unless there is a conclusion that there should be a change.

3.2.8. Expired Connection

Entry to this state occurs when no datagrams have been forwarded recently via the appropriate SPD and SAD objects. The objects in the forwarding plane are removed (logging any final byte and packet counts, if appropriate) and the connection instance in the keying plane is deleted.

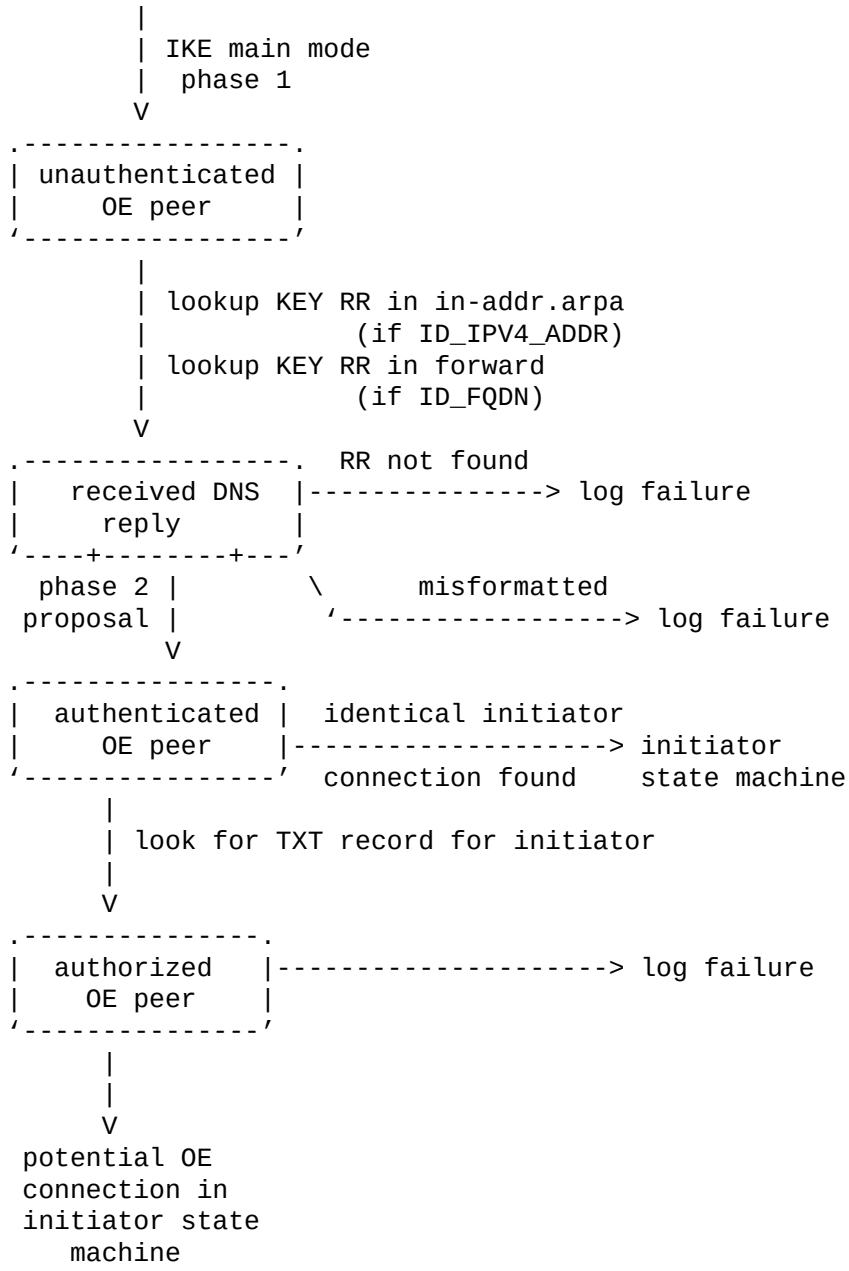
The initiator sends an ISAKMP/IKE delete to clean up the phase 2 SAs as described in Section 3.4.

Whether or not to delete the phase 1 SAs at this time is left as a local implementation issue. Implementations that do delete the phase 1 SAs MUST send authenticated delete messages to indicate that they are doing so. There is an advantage to keeping the phase 1 SAs until they expire: they may prove useful again in the near future.

3.3. Keying Daemon -- Responder

The responder has a set of objects identical to those of the initiator.

The responder receives an invitation to create a keying channel from an initiator.



3.3.1. Unauthenticated OE Peer

Upon entering this state, the responder starts a DNS lookup for a KEY record for the initiator. The responder looks in the reverse-map for a KEY record for the initiator if the initiator has offered an ID_IPV4_ADDR, and in the forward map if the initiator has offered an ID_FQDN type. (See [RFC2407] section 4.6.2.1.)

The responder exits this state upon successful receipt of a KEY from DNS, and use of the key to verify the signature of the initiator.

Successful authentication of the peer results in a transition to the authenticated OE Peer state.

Note that the unauthenticated OE peer state generally occurs in the middle of the key negotiation protocol. It is really a form of pseudo-state.

3.3.2. Authenticated OE Peer

The peer will eventually propose one or more phase 2 SAs. The responder uses the source and destination address in the proposal to finish instantiating the connection state using the connection class table. The responder MUST search for an identical connection object at this point.

If an identical connection is found, then the responder deletes the old instance, and the new object makes a transition to the pending OE connection state. This means that new ISAKMP connections with a given peer will always use the latest instance, which is the correct one if the peer has rebooted in the interim.

If an identical connection is not found, then the responder makes the transition according to the rules given for the initiator: it installs appropriate policy: clear, drop, or OE.

If OE, and the phase 2 ID (source IP) is different than the phase 1 ID, then additional authorization is required. A TXT record associated with the proposed phase 2 source IP is requested. This is used to confirm authorization for the phase 1 identity to encrypt on behalf of the phase 2. Successful retrieval results in a transition to "Authorized OE Peer".

Note that if the initiator is in OE-paranoid mode and the responder is in either always-clear-text or deny, then no communication is possible according to policy. An implementation is permitted to create new types of policies such as "accept OE but do not initiate it". This is a local matter.

3.3.3. Authorized OE Peer

This state is entered from the Authenticated OE Peer state, upon successful retrieval of the TXT record. The contents of the record are confirmed -- any failures lead to errors, as indicated in Section 3.2.4.

3.4. Renewal and Teardown

3.4.1. Aging

A potentially unlimited number of tunnels may exist. In practice, only a few tunnels are used during a period of time. Unused tunnels MUST, therefore, be torn down. Detecting when tunnels are no longer in use is the subject of this section.

There are two methods for removing tunnels: explicit deletion or expiry.

Explicit deletion requires an IKE delete message. The deletes MUST be authenticated, so both ends of the tunnel must maintain the keying channel (phase 1 ISAKMP SA). An implementation that refuses to either maintain or recreate the keying channel SA will be unable to use this method.

The tunnel expiry method simply allows the IKE daemon to expire normally without attempting to re-key it.

Regardless of which method is used to remove tunnels, the implementation MUST use a method to determine if the tunnel is still in use. The specifics are a local matter, but the FreeS/WAN project uses the following criteria. These criteria are currently implemented in the key management daemon, but could also be implemented at the SPD layer using an idle timer.

Set a short initial (soft) lifespan of 1 minute since many net flows last only a few seconds.

At the end of the lifespan, check to see if the tunnel was used by traffic in either direction during the last 30 seconds. If so, assign a longer tentative lifespan of 20 minutes, after which, look again. If the tunnel is not in use, then close the tunnel.

The expiring state in the key management system (see Section 3.2.7) implements these timeouts. The timer above may be in the forwarding plane, but then it must be resettable.

The tentative lifespan is independent of re-keying; it is just the time when the tunnel's future is next considered. (The term lifespan is used here rather than lifetime for this reason.) Unlike re-keying, this tunnel use check is not costly and should happen reasonably frequently.

A multi-step back-off algorithm is not considered worth the effort here.

If the security gateway and the client host are the same, and not a Bump-in-the-Stack or Bump-in-the-Wire implementation, tunnel teardown decisions MAY pay attention to TCP connection status as reported by the local TCP layer. A still-open TCP connection is almost a guarantee that more traffic is expected. Closing of the only TCP connection through a tunnel is a strong hint that no more traffic is expected.

3.4.2. Teardown and Cleanup

Teardown should always be coordinated between the two ends of the tunnel by interpreting and sending delete notifications. There is a detailed sub-state in the expired connection state of the key manager that relates to retransmits of the delete notifications, but this is considered to be a keying system detail.

On receiving a delete for the outbound SAs of a tunnel (or some subset of them), tear down the inbound ones also and notify the remote end with a delete. If the local system receives a delete for a tunnel that is no longer in existence, then two delete messages have crossed paths. Ignore the delete. The operation has already been completed. Do not generate any messages in this situation.

Tunnels are to be considered as bidirectional entities, even though the low-level protocols don't treat them this way.

When the deletion is initiated locally, rather than as a response to a received delete, send a delete for (all) the inbound SAs of a tunnel. If the local system does not receive a responding delete for the outbound SAs, try re-sending the original delete. Three tries spaced 10 seconds apart seems a reasonable level of effort. A failure of the other end to respond after 3 attempts indicates that the possibility of further communication is unlikely. Remove the outgoing SAs. (The remote system may be a mobile node that is no longer present or powered on.)

After re-keying, transmission should switch to using the new outgoing SAs (ISAKMP or IPsec) immediately, and the old leftover outgoing SAs should be cleared out promptly (delete should be sent for the

outgoing SAs) rather than waiting for them to expire. This reduces clutter and minimizes confusion for the operator doing diagnostics.

4. Impacts on IKE

4.1. ISAKMP/IKE Protocol

The IKE wire protocol needs no modifications. The major changes are implementation issues relating to how the proposals are interpreted, and from whom they may come.

As opportunistic encryption is designed to be useful between peers without prior operator configuration, an IKE daemon must be prepared to negotiate phase 1 SAs with any node. This may require a large amount of resources to maintain cookie state, as well as large amounts of entropy for nonces, cookies, and so on.

The major changes to support opportunistic encryption are at the IKE daemon level. These changes relate to handling of key acquisition requests, lookup of public keys and TXT records, and interactions with firewalls and other security facilities that may be co-resident on the same gateway.

4.2. Gateway Discovery Process

In a typical configured tunnel, the address of SG-B is provided via configuration. Furthermore, the mapping of an SPD entry to a gateway is typically a 1:1 mapping. When the 0.0.0.0/0 SPD entry technique is used, then the mapping to a gateway is determined by the reverse DNS records.

The need to do a DNS lookup and wait for a reply will typically introduce a new state and a new event source (DNS replies) to IKE. Although a synchronous DNS request can be implemented for proof of concept, experience is that it can cause very high latencies when a queue of queries must all timeout in series.

Use of an asynchronous DNS lookup will also permit overlap of DNS lookups with some of the protocol steps.

4.3. Self Identification

SG-A will have to establish its identity. Use an IPv4 (IPv6) ID in phase 1.

There are many situations where the administrator of SG-A may not be able to control the reverse DNS records for SG-A's public IP address. Typical situations include dialup connections and most residential-

type broadband Internet access (ADSL, cable-modem) connections. In these situations, a fully qualified domain name that is under the control of SG-A's administrator may be used when acting as an initiator only. The FQDN ID should be used in phase 1. See Section 5.3 for more details and restrictions.

4.4. Public Key Retrieval Process

Upon receipt of a phase 1 SA proposal with either an IPv4 (IPv6) ID or an FQDN ID, an IKE daemon needs to examine local caches and configuration files to determine if this is part of a configured tunnel. If no configured tunnels are found, then the implementation should attempt to retrieve a KEY record from the reverse DNS in the case of an IPv4/IPv6 ID, or from the forward DNS in the case of FQDN ID.

It is reasonable that if other non-local sources of policy are used (COPS, LDAP), they be consulted concurrently, but that some clear ordering of policy be provided. Note that due to variances in latency, implementations must wait for positive or negative replies from all sources of policy before making any decisions.

4.5. Interactions with DNSSEC

The implementation described (FreeS/WAN 1.98) neither uses DNSSEC directly to explicitly verify the authenticity of zone information, nor uses the NSEC records to provide authentication of the absence of a TXT or KEY record. Rather, this implementation uses a trusted path to a DNSSEC-capable caching resolver.

To distinguish between an authenticated and an unauthenticated DNS resource record, a stub resolver capable of returning DNSSEC information MUST be used.

4.6. Required Proposal Types

4.6.1. Phase 1 Parameters

Main mode MUST be used.

The initiator MUST offer at least one proposal using some combination of: 3DES, HMAC-MD5 or HMAC-SHA1, DH group 2 or 5. Group 5 SHOULD be proposed first. (See [RFC3526])

The initiator MAY offer additional proposals, but the cipher MUST not be weaker than 3DES. The initiator SHOULD limit the number of proposals such that the IKE datagrams do not need to be fragmented.

The responder MUST accept one of the proposals. If any configuration of the responder is required, then the responder is not acting in an opportunistic way.

The initiator SHOULD use an ID_IPV4_ADDR (ID_IPV6_ADDR for IPv6) of the external interface of the initiator for phase 1. (There is an exception, see Section 5.3.) The authentication method MUST be RSA public key signatures. The RSA key for the initiator SHOULD be placed into a DNS KEY record in the reverse space of the initiator (i.e., using in-addr.arpa or ip6.arpa).

4.6.2. Phase 2 Parameters

The initiator MUST propose a tunnel between the ultimate sender ("Alice" or "A") and ultimate recipient ("Bob" or "B") using 3DES-CBC mode, MD5, or SHA1 authentication. Perfect Forward Secrecy MUST be specified.

Tunnel mode MUST be used.

Identities MUST be ID_IPV4_ADDR_SUBNET with the mask being /32.

Authorization for the initiator to act on Alice's behalf is determined by looking for a TXT record in the reverse-map at Alice's IP address.

Compression SHOULD NOT be mandatory. It MAY be offered as an option.

5. DNS Issues

5.1. Use of KEY Record

In order to establish their own identities, security gateways SHOULD publish their public keys in their reverse DNS via DNSSEC's KEY record. See section 3 of RFC 2535 [RFC2535].

For example:

KEY 0x4200 4 1 AQNJjkKlIk9...nYyUkKK8

0x4200: The flag bits, indicating that this key is prohibited for confidentiality use (it authenticates the peer only, a separate Diffie-Hellman exchange is used for confidentiality), and that this key is associated with the non-zone entity whose name is the RR owner name. No other flags are set.

4: This indicates that this key is for use by IPsec.

1: An RSA key is present.

AQNJjkKlIk9...nYyUkKK8: The public key of the host as described in [RFC3110].

Use of several KEY records allows for key roll-over. The SIG Payload in IKE phase 1 SHOULD be accepted if the public key, given by any KEY RR, validates it.

5.2. Use of TXT Delegation Record

If, for example, machine Alice wishes SG-A to act on her behalf, then she publishes a TXT record to provide authorization for SG-A to act on Alice's behalf. This is done similarly for Bob and SG-B.

These records are located in the reverse DNS (in-addr.arpa or ip6.arpa) for their respective IP addresses. The reverse DNS SHOULD be secured by DNSSEC. DNSSEC is required to defend against active attacks.

If Alice's address is P.Q.R.S, then she can authorize another node to act on her behalf by publishing records at:

```
S.R.Q.P.in-addr.arpa
```

The contents of the resource record are expected to be a string that uses the following syntax, as suggested in RFC1464 [RFC1464]. (Note that the reply to query may include other TXT resource records used by other applications.)

```
X-IPsec-Server(P)=A.B.C.D public-key
```

Figure 2: Format of reverse delegation record

P: Specifies a precedence for this record. This is similar to MX record preferences. Lower numbers have stronger preference.

A.B.C.D: Specifies the IP address of the Security Gateway for this client machine.

public-key: Is the encoded RSA Public key of the Security Gateway. The public-key is provided here to avoid a second DNS lookup. If this field is absent, then a KEY resource record should be looked up in the reverse-map of A.B.C.D. The key is transmitted in base64 format.

The fields of the record MUST be separated by whitespace. This MAY be: space, tab, newline, or carriage return. A space is preferred.

In the case where Alice is located at a public address behind a security gateway that has no fixed address (or no control over its reverse-map), then Alice may delegate to a public key by domain name.

```
X-IPsec-Server(P)=@FQDN public-key
```

Figure 3: Format of reverse delegation record (FQDN version)

P: Is as above.
FQDN: Specifies the FQDN that the Security Gateway will identify itself with.
public-key: Is the encoded RSA Public key of the Security Gateway.

If there is more than one such TXT record with strongest (lowest numbered) precedence, one Security Gateway is picked arbitrarily from those specified in the strongest-preference records.

5.2.1. Long TXT Records

When packed into wire-format, TXT records that are longer than 255 characters are divided into smaller <character-strings>. (See [RFC1035] section 3.3 and 3.3.14.) These MUST be reassembled into a single string for processing. Whitespace characters in the base64 encoding are to be ignored.

5.2.2. Choice of TXT Record

It has been suggested to use the KEY, OPT, CERT, or KX records instead of a TXT record. None is satisfactory.

The KEY RR has a protocol field that could be used to indicate a new protocol, and an algorithm field that could be used to indicate different contents in the key data. However, the KEY record is clearly not intended for storing what are really authorizations, it is just for identities. Other uses have been discouraged.

OPT resource records, as defined in [RFC2671], are not intended to be used for storage of information. They are not to be loaded, cached or forwarded. They are, therefore, inappropriate for use here.

CERT records [RFC2538] can encode almost any set of information. A custom type code could be used permitting any suitable encoding to be stored, not just X.509. According to the RFC, the certificate RRs are to be signed internally, which may add undesirable and unnecessary bulk. Larger DNS records may require TCP instead of UDP transfers.

At the time of protocol design, the CERT RR was not widely deployed and could not be counted upon. Use of CERT records will be investigated, and may be proposed in a future revision of this document.

KX records are ideally suited for use instead of TXT records, but had not been deployed at the time of implementation.

5.3. Use of FQDN IDs

Unfortunately, not every administrator has control over the contents of the reverse-map. Where the initiator (SG-A) has no suitable reverse-map, the authorization record present in the reverse-map of Alice may refer to a FQDN instead of an IP address.

In this case, the client's TXT record gives the fully qualified domain name (FQDN) in place of its security gateway's IP address. The initiator should use the ID_FQDN ID-payload in phase 1. A forward lookup for a KEY record on the FQDN must yield the initiator's public key.

This method can also be used when the external address of SG-A is dynamic.

If SG-A is acting on behalf of Alice, then Alice must still delegate authority for SG-A to do so in her reverse-map. When Alice and SG-A are one and the same (i.e., Alice is acting as an end-node) then there is no need for this when initiating only.

However, Alice must still delegate to herself if she wishes others to initiate OE to her. See Figure 3.

5.4. Key Roll-Over

Good cryptographic hygiene says that one should replace public/private key pairs periodically. Some administrators may wish to do this as often as daily. Typical DNS propagation delays are determined by the SOA Resource Record MINIMUM parameter, which controls how long DNS replies may be cached. For reasonable operation of DNS servers, administrators usually want this value to be at least several hours, sometimes as a long as a day. This presents a problem: a new key MUST not be used prior to its propagation through DNS.

This problem is dealt with by having the Security Gateway generate a new public/private key pair, at least MINIMUM seconds in advance of using it. It then adds this key to the DNS (both as a second KEY

record and in additional TXT delegation records) at key generation time. Note: only one key is allowed in each TXT record.

When authenticating, all gateways MUST have available all public keys that are found in DNS for this entity. This permits the authenticating end to check both the key for "today" and the key for "tomorrow". Note that it is the end which is creating the signature (possesses the private key) that determines which key is to be used.

6. Network Address Translation Interaction

There are no fundamentally new issues for implementing opportunistic encryption in the presence of network address translation. Rather, there are only the regular IPsec issues with NAT traversal.

There are several situations to consider for NAT.

6.1. Co-Located NAT/NAPT

If a security gateway is also performing network address translation on behalf of an end-system, then the packet should be translated prior to being subjected to opportunistic encryption. This is in contrast to typically configured tunnels, which often exist to bridge islands of private network address space. The security gateway will use the translated source address for phase 2, and so the responding security gateway will look up that address to confirm SG-A's authorization.

In the case of NAT (1:1), the address space into which the translation is done MUST be globally unique, and control over the reverse-map is assumed. Placing of TXT records is possible.

In the case of NAPT (m:1), the address will be the security gateway itself. The ability to get KEY and TXT records in place will again depend upon whether or not there is administrative control over the reverse-map. This is identical to situations involving a single host acting on behalf of itself. For initiators (but not responders), an FQDN-style ID can be used to get around a lack of a reverse-map.

6.2. Security Gateway behind a NAT/NAPT

If there is a NAT or NAPT between the security gateways, then normal IPsec NAT traversal problems occur. In addition to the transport problem, which may be solved by other mechanisms, there is the issue of what phase 1 and phase 2 IDs to use. While FQDN could be used during phase 1 for the security gateway, there is no appropriate ID for phase 2. Due to the NAT, the end systems live in different IP address spaces.

6.3. End System behind a NAT/NAPT

If the end system is behind a NAT (perhaps SG-B), then there is, in fact, no way for another end system to address a packet to this end system. Not only is opportunistic encryption impossible, but it is also impossible for any communication to be initiated to the end system. It may be possible for this end system to initiate such communication. This creates an asymmetry, but this is common for NAPT.

7. Host Implementations

When Alice and SG-A are components of the same system, they are considered to be a host implementation. The packet sequence scenario remains unchanged.

Components marked Alice are the upper layers (TCP, UDP, the application), and SG-A is the IP layer.

Note that tunnel mode is still required.

As Alice and SG-A are acting on behalf of themselves, no TXT based delegation record is necessary for Alice to initiate. She can rely on FQDN in a forward map. This is particularly attractive to mobile nodes such as notebook computers at conferences. To respond, Alice/SG-A will still need an entry in Alice's reverse-map.

8. Multi-Homing

If there are multiple paths between Alice and Bob (as illustrated in the diagram with SG-D), then additional DNS records are required to establish authorization.

In Figure 1, Alice has two ways to exit her network: SG-A and SG-D. Previously, SG-D has been ignored. Postulate that there are routers between Alice and her set of security gateways (denoted by the + signs and the marking of an autonomous system number for Alice's network). Datagrams may, therefore, travel to either SG-A or SG-D en route to Bob.

As long as all network connections are in good order, it does not matter how datagrams exit Alice's network. When they reach either security gateway, the security gateway will find the TXT delegation record in Bob's reverse-map, and establish an SA with SG-B.

SG-B has no problem establishing that either of SG-A or SG-D may speak for Alice, because Alice has published two equally weighted TXT delegation records:

```
X-IPsec-Server(10)=192.1.1.5 AQMM...3s1Q==
X-IPsec-Server(10)=192.1.1.6 AAJN...j8r9==
```

Figure 4: Multiple gateway delegation example for Alice

Alice's routers can now do any kind of load sharing needed. Both SG-A and SG-D send datagrams addressed to Bob through their tunnel to SG-B.

Alice's use of non-equal weight delegation records to show preference of one gateway over another, has relevance only when SG-B is initiating to Alice.

If the precedences are the same, then SG-B has a more difficult time. It must decide which of the two tunnels to use. SG-B has no information about which link is less loaded, nor which security gateway has more cryptographic resources available. SG-B, in fact, has no knowledge of whether both gateways are even reachable.

The Public Internet's default-free zone may well know a good route to Alice, but the datagrams that SG-B creates must be addressed to either SG-A or SG-D; they can not be addressed to Alice directly.

SG-B may make a number of choices:

1. It can ignore the problem and round robin among the tunnels. This causes losses during times when one or the other security gateway is unreachable. If this worries Alice, she can change the weights in her TXT delegation records.
2. It can send to the gateway from which it most recently received datagrams. This assumes that routing and reachability are symmetrical.
3. It can listen to BGP information from the Internet to decide which system is currently up. This is clearly much more complicated, but if SG-B is already participating in the BGP peering system to announce Bob, the results data may already be available to it.
4. It can refuse to negotiate the second tunnel. (It is unclear whether or not this is even an option.)
5. It can silently replace the outgoing portion of the first tunnel with the second one while still retaining the incoming portions of both. Thus, SG-B can accept datagrams from either SG-A or SG-D, but send only to the gateway that most recently re-keyed with it.

Local policy determines which choice SG-B makes. Note that even if SG-B has perfect knowledge about the reachability of SG-A and SG-D, Alice may not be reachable from either of these security gateways because of internal reachability issues.

FreeS/WAN implements option 5. Implementing a different option is being considered. The multi-homing aspects of OE are not well developed and may be the subject of a future document.

9. Failure Modes

9.1. DNS Failures

If a DNS server fails to respond, local policy decides whether or not to permit communication in the clear as embodied in the connection classes in Section 3.2. It is easy to mount a denial of service attack on the DNS server responsible for a particular network's reverse-map. Such an attack may cause all communication with that network to go in the clear if the policy is permissive, or fail completely if the policy is paranoid. Please note that this is an active attack.

There are still many networks that do not have properly configured reverse-maps. Further, if the policy is not to communicate, the above denial of service attack isolates the target network. Therefore, the decision of whether or not to permit communication in the clear MUST be a matter of local policy.

9.2. DNS Configured, IKE Failures

DNS records claim that opportunistic encryption should occur, but the target gateway either does not respond on port 500, or refuses the proposal. This may be because of a crash or reboot, a faulty configuration, or a firewall filtering port 500.

The receipt of ICMP port, host or network unreachable messages indicates a potential problem, but MUST NOT cause communication to fail immediately. ICMP messages are easily forged by attackers. If such a forgery caused immediate failure, then an active attacker could easily prevent any encryption from ever occurring, possibly preventing all communication.

In these situations a log should be produced and local policy should dictate if communication is then permitted in the clear.

9.3. System Reboots

Tunnels sometimes go down because the remote end crashes, disconnects, or has a network link break. In general there is no notification of this. Even in the event of a crash and successful reboot, other SGs don't hear about it unless the rebooted SG has specific reason to talk to them immediately. Over-quick response to temporary network outages is undesirable. Note that a tunnel can be torn down and then re-established without any effect visible to the user except a pause in traffic. On the other hand, if one end reboots, the other end can't get datagrams to it at all (except via IKE) until the situation is noticed. So a bias toward quick response is appropriate, even at the cost of occasional false alarms.

A mechanism for recovery after reboot is a topic of current research and is not specified in this document.

A deliberate shutdown should include an attempt, using delete messages, to notify all other SGs currently connected by phase 1 SAs that communication is about to fail. Again, a remote SG will assume this is a teardown. Attempts by the remote SGs to negotiate new tunnels as replacements should be ignored. When possible, SGs should attempt to preserve information about currently-connected SGs in non-volatile storage, so that after a crash, an Initial-Contact can be sent to previous partners to indicate loss of all previously established connections.

10. Unresolved Issues

10.1. Control of Reverse DNS

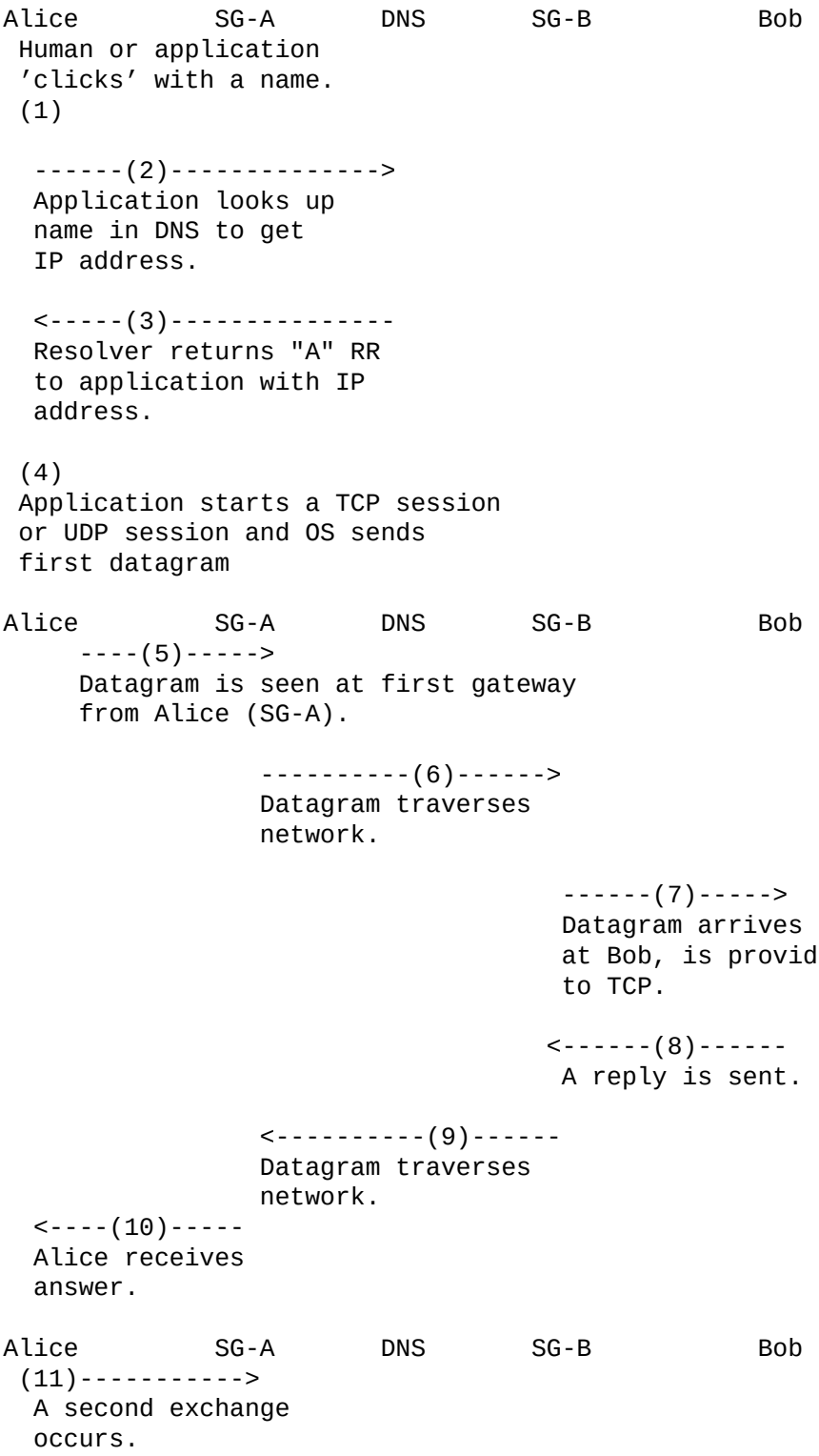
The method of obtaining information by reverse DNS lookup causes problems for people who cannot control their reverse DNS bindings. This is an unresolved problem in this version, and is out of scope.

11. Examples

11.1. Clear-Text Usage (Permit Policy)

Two example scenarios follow. In the first example, GW-A (Gateway A) and GW-B (Gateway B) have always-clear-text policies, and in the second example they have an OE policy. The clear-text policy serves as a reference for what occurs in TCP/IP in the absence of Opportunistic Encryption.

Alice wants to communicate with Bob. Perhaps she wants to retrieve a web page from Bob's web server. In the absence of opportunistic encryptors, the following events occur:


Alice SG-A DNS SG-B Bob
Human or application
'clicks' with a name.
(1)
------(2)-------------->
Application looks up
name in DNS to get
IP address.
<-----(3)---------------
Resolver returns "A" RR
to application with IP
address.
(4)
Application starts a TCP session
or UDP session and OS sends
first datagram
Alice SG-A DNS SG-B Bob
----(5)----->
Datagram is seen at first gateway
from Alice (SG-A).
----------(6)------>
Datagram traverses
network.
------(7)----->
Datagram arrives
at Bob, is provided
to TCP.
<------(8)------
A reply is sent.
<----------(9)------
Datagram traverses
network.
<----(10)-----
Alice receives
answer.
Alice SG-A DNS SG-B Bob
(11)----------->
A second exchange
occurs.

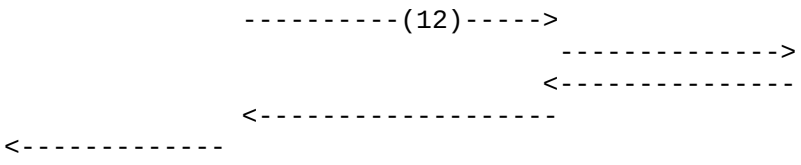


Figure 5: Timing of regular transaction

11.2. Opportunistic Encryption

In the presence of properly configured opportunistic encryptors, the event list is extended. Only changes are annotated.

The following symbols are used in the time-sequence diagram:

- - A single dash represents clear-text datagrams.
- = An equals sign represents phase 2 (IPsec) cipher-text datagrams.
- ~ A single tilde represents clear-text phase 1 datagrams.
- # A hash sign represents phase 1 (IKE) cipher-text datagrams.

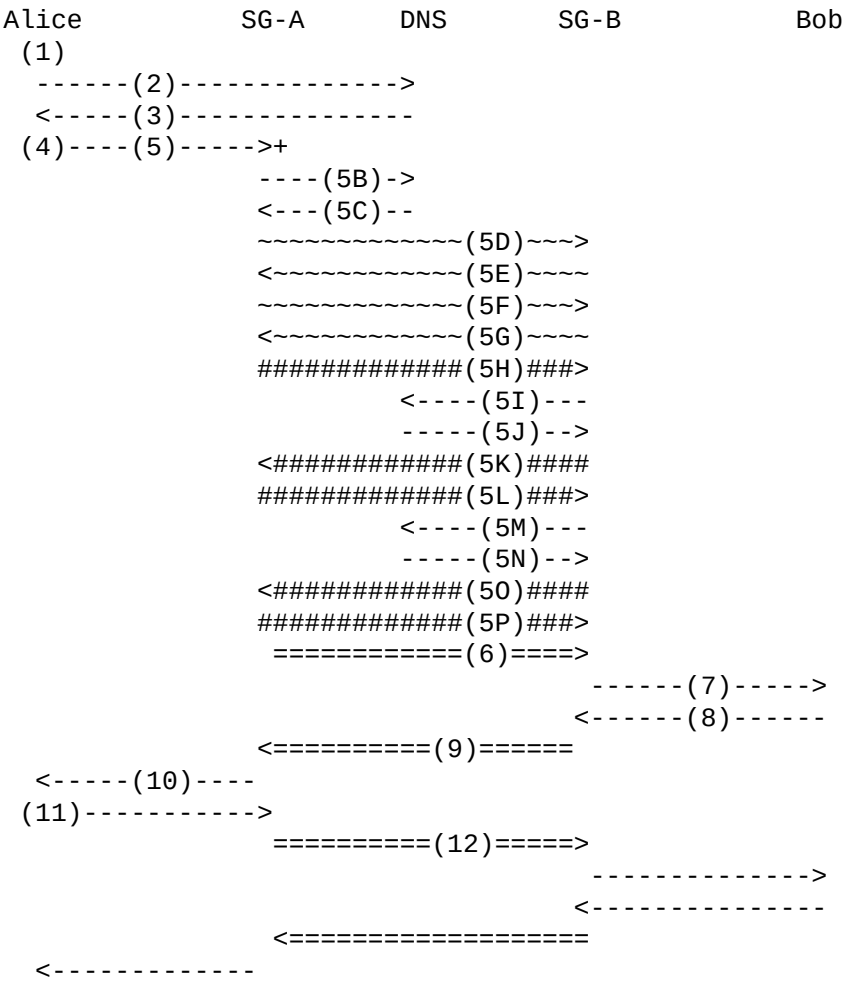


Figure 6: Timing of opportunistic encryption transaction

For the purposes of this section, we will describe only the changes that occur between Figure 5 and Figure 6. This corresponds to time points 5, 6, 7, 9, and 10 on the list above.

At point (5), SG-A intercepts the datagram because this source/destination pair lacks a policy (the nonexistent policy state). SG-A creates a hold policy, and buffers the datagram. SG-A requests keys from the keying daemon.

(5B) DNS query for TXT record.
(5C) DNS response for TXT record.
(5D) Initial IKE message to responder.
(5E) Message 2 of phase 1 exchange.
SG-B receives the message. A new connection instance is created in the unauthenticated OE peer state.
(5F) Message 3 of phase 1 exchange.
SG-A sends a Diffie-Hellman exponent. This is an internal state of the keying daemon.
(5G) Message 4 of phase 1 exchange.
SG-B responds with a Diffie-Hellman exponent. This is an internal state of the keying protocol.
(5H) Message 5 of phase 1 exchange.
SG-A uses the phase 1 SA to send its identity under encryption. The choice of identity is discussed in Section 4.6.1. This is an internal state of the keying protocol.
(5I) Responder lookup of initiator key. SG-B asks DNS for the public key of the initiator. DNS looks for a KEY record by IP address in the reverse-map. That is, a KEY resource record is queried for 4.1.1.192.in-addr.arpa (recall that SG-A's external address is 192.1.1.4). SG-B uses the resulting public key to authenticate the initiator. See Section 5.1 for further details.
(5J) DNS replies with public key of initiator.
Upon successfully authenticating the peer, the connection instance makes a transition to authenticated OE peer on SG-B. The format of the TXT record returned is described in Section 5.2.
Responder replies with ID and authentication.
SG-B sends its ID along with authentication material, completing the phase 1 negotiation.
(5L) IKE phase 2 negotiation.
Having established mutually agreeable authentications (via KEY) and authorizations (via TXT), SG-A proposes to create an IPsec tunnel for datagrams transiting from Alice to Bob. This tunnel is established only for the Alice/Bob combination, not for any subnets that may be behind SG-A and SG-B.

(5M) Authorization for SG-A to speak for Alice.
While the identity of SG-A has been established, its authority to speak for Alice has not yet been confirmed. SG-B does a reverse lookup on Alice's address for a TXT record.
(5N) Responder determines initiator's authority.
A TXT record is returned. It confirms that SG-A is authorized to speak for Alice.
Upon receiving this specific proposal, SG-B's connection instance makes a transition into the potential OE connection state. SG-B may already have an instance, and the check is made as described above.
(5O) Responder agrees to proposal.
SG-B, satisfied that SG-A is authorized, proceeds with the phase 2 exchange.
The responder MUST setup the inbound IPsec SAs before sending its reply.
(5P) Final acknowledgement from initiator.
The initiator agrees with the responder's choice of proposal and sets up the tunnel. The initiator sets up the inbound and outbound IPsec SAs.
Upon receipt of this message, the responder may now setup the outbound IPsec SAs.
(6) IPsec succeeds and sets up a tunnel for communication between Alice and Bob.

SG-A sends the datagram saved at step (5) through the newly created tunnel to SG-B, where it gets decrypted and forwarded. Bob receives it at (7) and replies at (8). SG-B already has a tunnel up with G1 and uses it. At (9), SG-B has already established an SPD entry mapping Bob->Alice via a tunnel, so this tunnel is simply applied. The datagram is encrypted to SG-A, decrypted by SG-A, and passed to Alice at (10).

12. Security Considerations

12.1. Configured versus Opportunistic Tunnels

Configured tunnels are setup using bilateral mechanisms: exchanging public keys (raw RSA, DSA, PKIX), pre-shared secrets, or by referencing keys that are in known places (distinguished name from LDAP, DNS). These keys are then used to configure a specific tunnel.

A pre-configured tunnel may be on all the time, or may be keyed only when needed. The endpoints of the tunnel are not necessarily static; many mobile applications (road warrior) are considered to be configured tunnels.

The primary characteristic is that configured tunnels are assigned specific security properties. They may be trusted in different ways relating to exceptions to firewall rules, exceptions to NAT processing, and to bandwidth or other quality of service restrictions.

Opportunistic tunnels are not inherently trusted in any strong way. They are created without prior arrangement. As the two parties are strangers, there MUST be no confusion of datagrams that arrive from opportunistic peers and those that arrive from configured tunnels. A security gateway MUST take care that an opportunistic peer cannot impersonate a configured peer.

Ingress filtering MUST be used to make sure that only datagrams authorized by negotiation (and the concomitant authentication and authorization) are accepted from a tunnel. This is to prevent one peer from impersonating another.

An implementation suggestion is to treat opportunistic tunnel datagrams as if they arrive on a logical interface distinct from other configured tunnels. As the number of opportunistic tunnels that may be created automatically on a system is potentially very high, careful attention to scaling should be taken into account.

As with any IKE negotiation, opportunistic encryption cannot be secure without authentication. Opportunistic encryption relies on DNS for its authentication information and, therefore, cannot be fully secure without a secure DNS. Without secure DNS, opportunistic encryption can protect against passive eavesdropping but not against active man-in-the-middle attacks.

12.2. Firewalls versus Opportunistic Tunnels

Typical usage of per datagram access control lists is to implement various kinds of security gateways. These are typically called "firewalls".

Typical usage of a virtual private network (VPN) within a firewall is to bypass all or part of the access controls between two networks. Additional trust (as outlined in the previous section) is given to datagrams that arrive in the VPN.

Datagrams that arrive via opportunistically configured tunnels MUST not be trusted. Any security policy that would apply to a datagram arriving in the clear SHOULD also be applied to datagrams arriving opportunistically.

12.3. Denial of Service

There are several different forms of denial of service that an implementor should be concerned with. Most of these problems are shared with security gateways that have large numbers of mobile peers (road warriors).

The design of ISAKMP/IKE, and its use of cookies, defend against many kinds of denial of service. Opportunism changes the assumption that if the phase 1 (ISAKMP) SA is authenticated, that it was worthwhile creating. Because the gateway will communicate with any machine, it is possible to form phase 1 SAs with any machine on the Internet.

13. Acknowledgements

Substantive portions of this document are based upon previous work by Henry Spencer. [OEspec]

Thanks to Tero Kivinen, Sandy Harris, Wes Hardarker, Robert Moskowitz, Jakob Schlyter, Bill Sommerfeld, John Gilmore, and John Denker for their comments and constructive criticism.

Sandra Hoffman and Bill Dickie did the detailed proof reading and editing.

14. References

14.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2401] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[RFC2407] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[RFC2408] Maughan, D., Schneider, M., and M. Schertler, "Internet Security Association and key Management Protocol (ISAKMP)", RFC 2408, November 1998.

[RFC2409] Harkins, D. and D. Carrel, "The Internet key Exchange (IKE)", RFC 2409, November 1998.

[RFC2535] Eastlake, D., "Domain Name System Security Extensions", RFC 2535, March 1999.

[RFC3110] Eastlake, D., "RSA/SHA-1 SIGs and RSA KEYs in the Domain Name System (DNS)", RFC 3110, May 2001.

14.2. Informative References

[IPSECKEY] Richardson, M., "A Method for Storing IPsec keying Material in DNS", RFC 4025, March 2005.

[OEspec] H. Spencer and Redelmeier, D., "Opportunistic Encryption", paper, http://www.freeswan.org/oeid/opportunism-spec.txt, May 2001.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1464] Rosenbaum, R., "Using the Domain Name System To Store Arbitrary String Attributes", RFC 1464, May 1993.

[RFC1812] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[RFC1984] IAB, IESG, Carpenter, B., and F. Baker, "IAB and IESG Statement on Cryptographic Technology and the Internet", RFC 1984, August 1996.

[RFC2367] McDonald, D., Metz, C. and B. Phan, "PF_KEY Key Management API, Version 2", RFC 2367, July 1998.

[RFC2538] Eastlake, D. and O. Gudmundsson, "Storing Certificates in the Domain Name System (DNS)", RFC 2538, March 1999.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

[RFC3330] IANA, "Special-Use IPv4 Addresses", RFC 3330, September 2002.

[RFC3445] Massey, D. and S. Rose, "Limiting the Scope of the KEY Resource Record (RR)", RFC 3445, December 2002.

[RFC3526] Kivinen, T. and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)", RFC 3526, May 2003.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

Authors' Addresses

Michael C. Richardson
Sandelman Software Works
470 Dawson Avenue
Ottawa, ON K1Z 5V7
CA

EMail: mcr@sandelman.ottawa.on.ca
URI: http://www.sandelman.ottawa.on.ca/

D. Hugh Redelmeier
Mimosa Systems Inc.
29 Donino Avenue
Toronto, ON M4N 2W6
CA

EMail: hugh@mimosa.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.