pki4ipsec
Internet-Draft
Expires: March 4, 2005

B. Korver
Xythos Software, Inc.

September 3, 2004

The Internet IP Security PKI Profile of IKEv1/ISAKMP, IKEv2, and PKIX
draft-ietf-pki4ipsec-ikecert-profile-02

Status of this Memo

Copyright Notice

Abstract

IKE/IPsec and PKIX both provide frameworks that must be profiled for use in a given application. This document provides a profile of IKE/IPsec and PKIX that defines the requirements for using PKI technology in the context of IKE/IPsec. The document complements protocol specifications such as IKEv1 and IKEv2, which assume the existence of public key certificates and related keying materials, but which do not address PKI issues explicitly. This document addresses those issues.

Table of Contents

1. Introduction

IKE [2], ISAKMP [5] and IKEv2 [3] provide a secure key exchange mechanism for use with IPSEC [4]. In many cases the peers authenticate using digital certificates as specified in PKIX [7]. Unfortunately, the combination of these standards leads to an underspecified set of requirements for the use of certificates in the context of IPsec.

ISAKMP references PKIX but in many cases merely specifies the contents of various messages without specifying their syntax or semantics. Meanwhile, PKIX provides a large set of certificate mechanisms which are generally applicable for Internet protocols, but little specific guidance for IPsec. Given the numerous underspecified choices, interoperability is hampered if all implementers do not make similar choices, or at least fail to account for implementations which have chosen differently.

This profile of the IKE and PKIX frameworks is intended to provide an agreed-upon standard for using PKI technology in the context of IPsec by profiling the PKIX framework for use with IKE and IPsec, and by documenting the contents of the relevant IKE payloads and further specifying their semantics.

In addition to providing a profile of IKE and PKIX, this document attempts to incorporate lessons learned from recent experience with both implementation and deployment, as well as the current state of related protocols and technologies.

Material from ISAKMP, IKEv1, IKEv2, or PKIX is not repeated here, and readers of this document are assumed to have read and understood those documents. The requirements and security aspects of those documents are fully relevant to this document as well.

This document is organized as follows. Section 2 defines special terminology used in the rest of this document, Section 3 provides the profile of IKEv1/ISAKMP and IKEv2, and Section 4 provides the profile of PKIX. Section 5 covers conventions for the out-of-band exchange of keying materials for configuration purposes.

This document is being discussed on the pki4ipsec@icsalabs.com mailing list.

2. Terms and Definitions

Except for those terms which are defined immediately below, all terms used in this document are defined in either the PKIX [7], ISAKMP [5], IKEv1 [2], IKEv2 [3], or DOI [1] documents.

- Peer source address: The source address in packets from a peer. This address may be different from any addresses asserted as the "identity" of the peer.
- FQDN: Fully qualified domain name.
- ID_USER_FQDN: IKEv2 renamed ID_USER_FQDN to ID_RFC822_ADDR. Both are referred to as ID_USER_FQDN in this document.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [9].

3. Profile of IKEv1/ISAKMP and IKEv2

3.1 Identification Payload

The Identification (ID) Payload is used to indicate the identity that the agent claims to be speaking for. The receiving agent can then use the ID as a lookup key for policy and whatever certificate store or directory that it has available. Our primary concern in this document is to profile the ID payload so that it can be safely used to generate or lookup policy. IKE mandates the use of the ID payload in Phase 1.

The DOI [1] defines the 11 types of Identification Data that can be used and specifies the syntax for these types. These are discussed

below in detail.

The ID payload requirements in this document cover only the portion of the explicit policy checks that deal with the Identification Payload specifically. For instance, in the case where ID does not contain an IP address, checks such as verifying that the peer source address is permitted by the relevant policy are not addressed here as they are out of the scope of this document.

Implementations SHOULD populate ID with identity information that is contained within the end entity certificate (This SHOULD does not contradict text in IKEv2 [3] Section 3.5 that implies a looser binding between these two). Populating ID with identity information from the end entity certificate enables recipients to use ID as a lookup key to find the peer end entity certificate.

Because implementations may use ID as a lookup key to determine which policy to use, all implementations MUST be especially careful to verify the truthfulness of the contents by verifying that they correspond to some keying material demonstrably held by the peer. Failure to do so may result in the use of an inappropriate or insecure policy. The following sections describe the methods for performing this binding.

The following table summarizes the binding of the Identification Payload to the contents of end-entity certificates and of identity information to policy. Each ID type is covered more thoroughly in the following sections.

ID type	Support for send	Correspond PKIX Attrib	Cert matching	SPD lookup rules

IP*_ADDR	MUST [1]	SubjAltName iPAddress	MUST [2]	[3], [4]
FQDN	MUST [1]	SubjAltName dNSName	MUST [2]	[3], [4]
USER_FQDN	MUST [1]	SubjAltName rfc822Name	MUST [2]	[3], [4]
DN	MUST [1]	Entire Subject, bitwise compare	MUST [2]	MUST support lookup on any combination of C, CN, O, or OU
IP range	MUST NOT	n/a	n/a	n/a
KEY_ID	MUST NOT	n/a	n/a	n/a

[1] = Implementation MUST have the configuration option to send this ID type in the ID payload. Whether or not the ID type is used is a matter of local configuration.

[2] = The ID in the ID payload MUST match the contents of the corresponding field (listed) in the certificate exactly, with no other lookup. The matched ID MAY be used for SPD lookup, but is not required to be used for this.

[3] = At a minimum, Implementation MUST be able to be configured to perform exact matching of the ID payload contents to an entry in the local SPD.

[4] = In addition, the implementation MAY also be configurable to perform substring or wildcard matches of ID payload contents to entries in the local SPD. (More on this in Section 3.1.5).

When sending an IPV4_ADDR, IPV6_ADDR, FQDN, or USER_FQDN,

implementations MUST be able to be configured to send the same string as appears in the corresponding SubjectAltName attribute. This document RECOMMENDS that deployers use this configuration option. All these ID types are treated the same: as strings that can be compared easily and quickly to a corresponding string in an explicit attribute in the certificate. Of these types, FQDN and USER_FQDN are RECOMMENDED over IP addresses (see discussion in Section 3.1.1).

When sending a DN as ID, implementations MUST send the entire DN in ID. Also, implementations MUST support at least the C, CN, O, and OU attributes for SPD matching. See Section 3.1.5 for more details about DN, including SPD matching.

Recipients MUST be able to perform SPD matching on the exact contents of the ID, and this SHOULD be the default setting. In addition, implementations MAY use substrings or wildcards in local policy configuration to do the SPD matching against the ID contents. In other words, implementations MUST be able to do exact matches of ID to SPD, but MAY also be configurable to do substring or wildcard matches of ID to SPD.

IKEv2 adds an optional IDr payload in the second exchange that the initiator may send to the responder in order to specify which of the responder's multiple identities should be used. The responder MAY choose to send an IDr in the 3rd exchange that differs in type or content from the initiator-generated IDr. The initiator MUST be able to receive a responder-generated IDr that is different from the one the initiator generated. Whether or not to accept such a response and continue with IKE processing is a matter of local policy.

3.1.1 ID_IPV4_ADDR and ID_IPV6_ADDR

Implementations MUST support either the ID_IPV4_ADDR or ID_IPV6_ADDR ID type. These addresses MUST be stored in "network byte order," as specified in IP [8]: The least significant bit (LSB) of each octet is the LSB of the corresponding byte in the network address. For the ID_IPV4_ADDR type, the payload MUST contain exactly four octets [8]. For the ID_IPV6_ADDR type, the payload MUST contain exactly sixteen octets [13].

Note that this document does NOT RECOMMEND populating the ID payload with IP addresses due to interoperability issues such as problems with NAT traversal, and problems with IP verification behavior.

Deployments may only want to consider using the IP address as IKE_ID if the following are true:

- the peer's IP address is fixed, not dynamically changing

- o the peer is NOT behind a NAT'ing device
- o the administrator intends the implementation to verify that the IP address in the peer's source matches the IP address in the IKE_ID received, and that of the certificate's iPAddress field in the subjectAltName extension.

Implementations MUST be capable of verifying that the IP address presented in IKE_ID matches via bitwise comparison the IP address present in the certificate's iPAddress field in the subjectAltName extension. Implementations MUST perform this verification by default. When comparing the contents of ID with the iPAddress field in the subjectAltName extension for equality, binary comparison MUST be performed. If the default is enabled, then a mismatch between the two MUST be treated as an error and security association setup MUST be aborted. This event SHOULD be auditable. Implementations MAY provide a configuration option to (i.e. local policy configuration can enable) skip that verification step, but that option MUST be off by default. We include the "option-to-skip" in order to permit better interoperability, as today implementations vary greatly in how they behave on this topic of verification between IKE_ID and cert contents.

Implementations MUST be capable of verifying that the address contained in the ID is the same as the peer source address, contained in the outer most IP header. If IKE_ID is one of the IP address types, then implementations MUST perform this verification by default. If this default is enabled, then a mismatch MUST be treated as an error and security association setup MUST be aborted. This event SHOULD be auditable. Implementations MAY provide a configuration option to (i.e. local policy configuration can enable) skip that verification step, but that option MUST be off by default. We include the "option-to-skip-validation" in order to permit better interoperability, as today implementations vary greatly in how they behave on this topic of verification to source IP.

If the default for both the verifications above are enabled, then, by transitive property, the implementation will also be verifying that the peer source IP address matches via a bitwise comparison the contents of the iPAddress field in the subjectAltName extension in the certificate. In addition, implementations MAY allow administrators to configure a local policy that explicitly requires that the peer source IP address match via a bitwise comparison the contents of the iPAddress field in the subjectAltName extension in

the certificate. Implementations SHOULD allow administrators to configure a local policy that skips this validation check.

Implementations MAY support substring, wildcard, or regular expression matching of the IKE_ID to contents in the SPD, and such

would be a matter of local security policy configuration.

Implementations MAY use the IP address found in the header of packets received from the peer to lookup the policy, but such implementations MUST still perform verification of the ID payload. Although packet IP addresses are inherently untrustworthy and must therefore be independently verified, it is often useful to use the apparent IP address of the peer to locate a general class of policies that will be used until the mandatory identity-based policy lookup can be performed.

For instance, if the IP address of the peer is unrecognized, a VPN gateway device might load a general "road warrior" policy that specifies a particular CA that is trusted to issue certificates which contain a valid rfc822Name which can be used by that implementation to perform authorization based on access control lists (ACLs) after the peer's certificate has been validated. The rfc822Name can then be used to determine the policy that provides specific authorization to access resources (such as IP addresses, ports, and so forth).

As another example, if the IP address of the peer is recognized to be a known peer VPN endpoint, policy may be determined using that address, but until the identity (address) is validated by validating the peer certificate, the policy MUST NOT be used to authorize any IPsec traffic.

3.1.2 ID_FQDN

Implementations MUST support the ID_FQDN ID type, generally to support host-based access control lists for hosts without fixed IP addresses. However, implementations SHOULD NOT use the DNS to map the FQDN to IP addresses for input into any policy decisions, unless that mapping is known to be secure, such as when [DNSSEC] is employed.

Implementations MUST be capable of verifying that the identity

contained in the ID payload matches identity information contained in the peer end entity certificate, in the dNSName field in the subjectAltName extension. Implementations MUST perform this verification by default. When comparing the contents of ID with the dNSName field in the subjectAltName extension for equality, caseless string comparison MUST be performed. Substring, wildcard, or regular expression matching MUST NOT be performed for this comparison. If this default is enabled, then a mismatch MUST be treated as an error and security association setup MUST be aborted. This event SHOULD be auditable. Implementations MAY provide a configuration option to (i.e. local policy configuration can enable) skip that verification step, but that option MUST be off by default. We include the

"option-to-skip-validation" in order to permit better interoperability, as today implementations vary greatly in how they behave on this topic.

Implementations MAY support substring, wildcard, or regular expression matching of the IKE_ID to contents in the SPD, and such would be a matter of local security policy configuration.

3.1.3 ID_USER_FQDN

Implementations MUST support the ID_USER_FQDN ID type, generally to support user-based access control lists for users without fixed IP addresses. However, implementations SHOULD NOT use the DNS to map the FQDN portion to IP addresses for input into any policy decisions, unless that mapping is known to be secure, such as when [DNSSEC] is employed.

Implementations MUST be capable of verifying that the identity contained in the ID payload matches identity information contained in the peer end entity certificate, in the rfc822Name field in the subjectAltName extension. Implementations MUST perform this verification by default. When comparing the contents of ID with the rfc822Name field in the subjectAltName extension for equality, caseless string comparison MUST be performed. Substring, wildcard, or regular expression matching MUST NOT be performed for this comparison. If this default is enabled, then a mismatch MUST be treated as an error and security association setup MUST be aborted. This event SHOULD be auditable. Implementations MAY provide a configuration option to (i.e. local policy configuration can enable)

skip that verification step, but that option MUST be off by default. We include the "option-to-skip-validation" in order to permit better interoperability, as today implementations vary greatly in how they behave on this topic.

Implementations MAY support substring, wildcard, or regular expression matching of the IKE_ID to contents in the SPD, and such would be a matter of local security policy configuration.

3.1.4 ID_IPV4_ADDR_SUBNET, ID_IPV6_ADDR_SUBNET, ID_IPV4_ADDR_RANGE, ID_IPV6_ADDR_RANGE

As there is currently no standard method for putting address subnet or range identity information into certificates, the use of these ID types is currently undefined. Implementations MUST NOT generate these ID types.

Note that work in SBGP [15] for defining blocks of addresses using the certificate extension identified by:

```
id-pe-ipAddrBlock OBJECT IDENTIFIER ::= { id-pe 7 }
```

is experimental at this time.

3.1.5 ID_DER_ASN1_DN

Implementations MUST support receiving the ID_DER_ASN1_DN ID type. Implementations MUST be capable of generating this type, and the decision to do so will be a matter of local security policy configuration. When generating this type, implementations MUST populate the contents of ID with the Subject Name from the end entity certificate, and MUST do so such that a binary comparison of the two will succeed. If there is not a match, this MUST be treated as an error and security association setup MUST be aborted. This event SHOULD be auditable. For instance, if the certificate was erroneously created such that the encoding of the Subject Name DN varies from the constraints set by DER, that non-conformant DN MUST be used to populate the ID payload: in other words, implementations MUST NOT re-encode the DN for the purposes of making it DER if it

does not appear in the certificate as DER.

Implementations MUST NOT populate ID with the Subject Name from the end entity certificate if it is empty, as described in the "Subject" section of PKIX.

Regarding SPD matching, implementations MUST be able to perform matching based on a bitwise comparison of the entire DN in ID to its entry in the SPD. However, operational experience has shown that using the entire DN in local configuration is difficult, especially in large scale deployments. Therefore, implementations also MUST be able to perform SPD matches of any combination of one or more of the C, CN, O, OU attributes within Subject DN in the ID to the same in the SPD. Implementations MAY support matching using additional DN attributes in any combination, although interoperability is far from certain and dubious. Implementations MAY also support performing substring, wildcard, or regular expression matches for any of its supported DN attributes from ID, in any combination, to the SPD. Such flexibility allows deployers to create one SPD entry on the gateway for an entire department of a company (e.g. O=Foobar Inc., OU=Engineering) while still allowing them to draw out other details from the DN (e.g. CN=John Doe) for auditing purposes. All the above is a matter of local implementation and local policy definition and enforcement capability, not bits on the wire, but will have a great impact on interoperability.

3.1.6 ID_DER_ASN1_GN

Implementations MUST NOT generate this type.

3.1.7 ID_KEY_ID

The ID_KEY_ID type used to specify pre-shared keys and thus is out of scope.

3.1.8 Selecting an Identity from a Certificate

Implementations MUST support certificates that contain more than a single identity. In many cases a certificate will contain an identity such as an IP address in the subjectAltName extension in addition to a non-empty Subject Name.

The identity with which an implementation chooses to populate the IKE_ID payload is a local matter. For compatibility with non-conformant implementations, implementations SHOULD populate ID with whichever identity is likely to be named in the peer's policy. In practice, this generally means FQDN, or USER_FQDN.

3.1.9 Transitively Binding Identity to Policy

In the presence of certificates that contain multiple identities, implementations MUST select the most appropriate identity from the certificate and populate the ID with that. The responder MUST use the identity sent as a first key when selecting the policy. Responder MUST also use most specific policy from that database if there are overlapping policies caused by wildcards (or the implementation can de-correlate the policy database so there will not be overlapping entries, or it can also forbid creation of overlapping policies and leave the de-correlation process to the administrator, but as this moves the problem to the administrator it is NOT RECOMMENDED).

For example, imagine that a peer is configured with a certificate that contains both a non-empty Subject Name and a dNSName. The initiator MUST know by policy which of those to use, and it indicates the policy in the other end by selecting the correct ID. If the responder has both a specific policy for the dNSName for this host, and generic wildcard rule for some attributes present in the subject Name, it will match a different policy depending which ID is sent. As the initiator knows why it wanted to connect the responder, it also knows what identity it should use to match the policy it needs to the operation it tries to perform; it is the only party who can select the ID adequately.

In the event the policy cannot be found in the responder's SPD using the ID sent by the initiator, then the responder MAY use the other identities in the certificate when attempting to match a suitable policy. For example, say the certificate contains both non-empty Subject Name, dNSName and iPAddress. The initiator sends ID of iPAddress, but the responder does not have that in the policy database. If the responder has a rule for the dNSName it MAY use policy based on that.

If overlapping policies are found in this step, the responder cannot know which one of those should be selected, i.e. if the responder does have rules for both Subject Name and for dNSName, and it would need to select one of those policies, but it cannot know which one to select. One or both of those rules could also be wildcard rules.

The responder cannot use de-correlation or forbidding the overlapping policies, as there is no way to detect those overlaps exist before the arrival of the certificate that makes the overlapping a reality. In the case where overlapping policies exist, the responder SHOULD terminate the negotiation with error, which informs the other end that administrative modification to its policy must be performed (i.e. it needs to use some other identity).

3.2 Certificate Request Payload

The Certificate Request (CERTREQ) Payload allows an implementation to request that a peer provide some set of certificates or certificate revocation lists. It is not clear from ISAKMP exactly how that set should be specified or how the peer should respond. We describe the semantics on both sides.

3.2.1 Certificate Type

The Certificate Type field identifies to the peer the type of certificate keying materials that are desired. ISAKMP defines 10 types of Certificate Data that can be requested and specifies the syntax for these types. For the purposes of this document, only the following types are relevant:

- X.509 Certificate - Signature
- Revocation Lists (CRL and ARL)
- PKCS #7 wrapped X.509 certificate
- IKEv2's Hash and URL of X.509 certificate

The use of the other types:

- X.509 Certificate - Key Exchange
- PGP Certificate
- DNS Signed Key

- Kerberos Tokens

- o SPKI Certificate
- o X.509 Certificate Attribute
- o IKEv2's Raw RSA Key
- o IKEv2's Hash and URL of X.509 bundle

are out of the scope of this document.

3.2.2 X.509 Certificate - Signature

This type requests that the end entity certificate be a signing certificate.

3.2.3 Revocation Lists (CRL and ARL)

ISAKMP and IKEv2 do not support Certificate Payload sizes over approximately 64K, which is too small for many CRLs. In addition, the acquisition of revocation material is to be dealt with out of band of IKE. For this and other reasons, implementations SHOULD NOT generate CERTREQs where the Certificate Type is "Certificate Revocation List (CRL)" or "Authority Revocation List (ARL)". Implementations that do generate such CERTREQs MUST NOT expect the responder to send a CRL or ARL, and MUST NOT fail for not receiving it. Upon receipt of such a CERTREQ, implementations MAY ignore the request.

In lieu of exchanging entire revocation lists in band, a pointer to revocation checking SHOULD be listed in either the Certificate Distribution Point (CDP) or the Authority Information Access (AIA) attributes of the certificate extensions (see Section 4 for details.) Implementations MUST be able to process these attributes, and from them be able to identify cached revocation material, or retrieve the relevant revocation material from a URL, for validation processing. In addition, implementations MUST have the ability to configure validation checking information for each certificate authority. Regardless of the method (CDP, AIA, or static configuration), the acquisition of revocation material occurs out of band of IKE.

3.2.4 PKCS #7 wrapped X.509 certificate

This ID type defines a particular encoding (not a particular certificate), some current implementations may ignore CERTREQs they receive which contain this ID type, and the authors are unaware of any implementations that generate such CERTREQ messages. Therefore, the use of this type is deprecated. Implementations SHOULD NOT require CERTREQs that contain this Certificate Type. Implementations which receive CERTREQs which contain this ID type MAY treat such payloads as synonymous with "X.509 Certificate - Signature".

3.2.5 IKEv2's Hash and URL of X.509 certificate

This ID type defines a request for the peer to send a hash and URL of it X.509 certificate, instead of the actual certificate itself. This is a particularly useful mechanism when the peer is a device with little memory and lower bandwidth, e.g. a mobile handset or consumer electronics device.

If the IKEv2 peer supports HTTP lookups, and prefers an HTTP-based URL to receiving the actual certificate, then the peer will want to send a notify of type HTTP_CERT_LOOKUP_SUPPORTED. From IKEv2 [3], section 3.10.1, "This notification MAY be included in any message that can include a CERTREQ payload and indicates that the sender is capable of looking up certificates based on an HTTP-based URL (and hence presumably would prefer to receive certificate specifications in that format)."

3.2.6 Presence or Absence of Certificate Request Payloads

When in-band exchange of certificate keying materials is desired, implementations MUST inform the peer of this by sending at least one CERTREQ. An implementation which does not send any CERTREQs during an exchange SHOULD NOT expect to receive any CERT payloads.

3.2.7 Certificate Requests

3.2.7.1 Specifying Certificate Authorities

Implementations MUST generate CERTREQs for every peer trust anchor that local policy explicitly deems trusted during a given exchange. For IKEv1, implementations MUST populate the Certificate Authority field with the Subject Name of the trust anchor, populated such that binary comparison of the Subject Name and the Certificate Authority will succeed. For IKEv2, implementations MUST populate the Certificate Authority field as specified in IKEv2 [3].

Upon receipt of a CERTREQ, implementations MUST respond by sending the end entity certificate corresponding to the Certificate Authority listed in the CERTREQ. Implementations SHOULD NOT send any certificates other than the appropriate end entity certificate (see Section 3.3 for discussion).

Note, in the case where multiple end entity certificates may be available, implementations SHOULD resort to local heuristics to determine which end entity is most appropriate to use for generating the CERTREQ. Such heuristics are out of the scope of this document.

3.2.7.2 Empty Certificate Authority Field

Implementations SHOULD generate CERTREQs where the Certificate Type is "X.509 Certificate - Signature" and where an entry exits in the Certificate Authority field. However, implementations MAY generate CERTREQs with an empty Certificate Authority field under special conditions. Though PKIX prohibits certificates with empty issuer name fields, there does exist a use case where doing so is appropriate, and carries special meaning in the IKE context. This has become a convention within the IKE interoperability tests and usage space, and so its use is specified, explained and RECOMMENDED here for the sake of interoperability.

USE CASE: Consider the case where you have a gateway with multiple policies for a large number of IKE peers.'some of these peers are business partners, some are remote access employees, some are teleworkers, some are branch offices, and/or the gateway may be simultaneously serving many many customers (e.g. Virtual Routers). The total number of certificates, and corresponding trust anchors, is very high, say hundreds. Each of these policies is configured with one or more acceptable trust anchors, so that in total, the gateway has one hundred (100) trust anchors that could possibly used to authenticate an incoming connection. Assume that many of those connections originate from hosts/gateways with dynamically assigned IP addresses, so that the source IP of the IKE initiator is not known to the gateway, nor is the identity of the intiator (until it is revealed in Main Mode message 5). In IKE main mode message 4, the responder gateway will need to send a CERTREQ to the initiator. Given this example, the gateway will have no idea which of the hundred possible Certificate Authorities to send in the CERTREQ. Sending all possible Certificate Authorities will cause significant processing delays, bandwidth consumption, and UDP fragmentation, so this tactic is ruled out.

In such a deployment, the responder gateway implementation should be able to all it can to indicate a Certificate Authority in the CERTREQ. This means the responder SHOULD first check SPD to see if it can match the source IP, and find some indication of which CA is associated with that IP. If this fails (because the source IP is not

familiar, as in the case above), then the responder SHOULD have a configuration option specifying which CA's are the default CAs to indicate in CERTREQ during such ambiguous connections (e.g. send CERTREQ with these N CAs if there is an unknown source IP). If such a fall-back is not configured or impractical in a certain deployment scenario, then the responder implementation SHOULD have both of the following configuration options:

- o send a CERTREQ payload with an empty Certificate Authority field, or

- o terminate the negotiation with an appropriate error message and audit log entry.

Receiving a CERTREQ payload with an empty Certificate Authority field indicates that the initiator peer should send all/any certificates it has, regardless of the trust anchor. The initiator should be aware of what policy and which identity it will use, as it initiated the connection on a matched policy to begin with, and can thus respond with the appropriate certificate. If multiple certificates are sent, they MUST have the same public key, otherwise the responder does not know which key was used in the Main Mode message 5.

If, after sending an empty CERTREQ in Main Mode message 4, a responder receives a certificate in message 5 from a trust anchor that the responder either (a) does NOT support, or (b) was not configured for the policy (that policy was now able to be matched due to having the initiators certificate present), then the responder SHOULD terminate the exchange with proper error message and audit log entry.

Instead of sending a empty CERTREQ, the responder implementation may be configured to terminate the negotiation on the grounds of a conflict with locally configured security policy.

The decision of which to configure is a matter of local security policy, this document RECOMMENDS that both options be presented to administrators.

More examples, and explanation on this issue are included in "More on Empty CERTREQs" (Appendix C).

3.2.8 Robustness

3.2.8.1 Unrecognized or Unsupported Certificate Types

Implementations MUST be able to deal with receiving CERTREQs with unsupported Certificate Types. Absent any recognized and supported CERTREQs, implementations MAY treat them as if they are of a supported type with the Certificate Authority field left empty, depending on local policy. ISAKMP [5] Section 5.10 "Certificate Request Payload Processing" specifies additional processing.

3.2.8.2 Undecodable Certificate Authority Fields

Implementations MUST be able to deal with receiving CERTREQs with undecodable Certificate Authority fields. Implementations MAY ignore such payloads, depending on local policy. ISAKMP specifies other actions which may be taken.

3.2.8.3 Ordering of Certificate Request Payloads

Implementations MUST NOT assume that CERTREQs are ordered in any way.

3.2.9 Optimizations

3.2.9.1 Duplicate Certificate Request Payloads

Implementations SHOULD NOT send duplicate CERTREQs during an exchange.

3.2.9.2 Name Lowest 'Common' Certification Authorities

When a peer's certificate keying materials have been cached, an implementation can send a hint to the peer to elide some of the certificates the peer would normally respond with. In addition to the normal set of CERTREQs that are sent specifying the trust anchors, an implementation MAY send CERTREQs containing the Issuer Name of the relevant cached end entity certificates. When sending these hints, it is still necessary to send the normal set of CERTREQs because the hints do not sufficiently convey all of the information required by the peer. Specifically, either the peer may not support this optimization or there may be additional chains that could be used in this context but will not be specified if only supplying the issuer of the end entity certificate.

No special processing is required on the part of the recipient of such a CERTREQ, and the end entity certificates will still be sent. On the other hand, the recipient MAY elect to elide certificates based on receipt of such hints.

CERTREQs must contain information that identifies a Certification Authority certificate, which results in the peer always sending at least the end entity certificate. This mechanism allows implementations to determine unambiguously when a new certificate is being used by the peer, perhaps because the previous certificate has just expired, which will result in a failure because the needed keying materials are not available to validate the new end entity certificate. Implementations which implement this optimization MUST recognize when the end entity certificate has changed and respond to it by not performing this optimization when the exchange is retried.

3.2.9.3 Example

Imagine that an implementation has previously received and cached the peer certificate chain TA->CA1->CA2->EE. If during a subsequent exchange this implementation sends a CERTREQ containing the Subject Name in certificate TA, this implementation is requesting that the

peer send at least 3 certificates: CA1, CA2, and EE. On the other hand, if this implementation also sends a CERTREQ containing the Subject Name of CA2, the implementation is providing a hint that only 1 certificate needs to be sent: EE. Note that in this example, the fact that TA is a trust anchor should not be construed to imply that TA is a self-signed certificate.

3.3 Certificate Payload

The Certificate (CERT) Payload allows the peer to transmit a single certificate or CRL. The following practice is explicitly deprecated: Some implementations also transmit each certificate in the chain above the end entity certificate up to and including the certificate whose Issuer Name matches the name specified in the Certificate Authority field. This practice is deprecated because the chaining certificates and validation material has now become a responsibility of the certificate management and lifecycle protocols between the IKE/IPsec peer and the PKI system, and not the transmission within

IKE. For backwards compatibility reasons, implementations MAY send intermediate CA certificates in addition to the appropriate end entity certificate, but SHOULD NOT send any CRLs, ARLs, or Trust Anchors. The reason for transmitting the intermediate CA certificates, CRL, ARL, and Trust anchors in the certificate management protocol instead of IKE is to:

- o simplify the IKE exchange
- o reduce bandwidth requirements for IKE exchanges
- o increase speed of completion (reduce latency) in IKE
- o decrease UDP fragmentation

Multiple certificates should be transmitted in multiple payloads. However, not all certificate forms that are legal in PKIX make sense in the context of IPsec. The issue of how to represent IKE-meaningful name-forms in a certificate is especially problematic. This document provides a profile for a subset of PKIX that makes sense for IKEv1/ISAKMP and IKEv2.

3.3.1 Certificate Type

The Certificate Type field identifies to the peer the type of certificate keying materials that are included. ISAKMP defines 10 types of Certificate Data that can be sent and specifies the syntax for these types. For the purposes of this document, only the following types are relevant:

- o X.509 Certificate - Signature
- o Revocation Lists (CRL and ARL)
- o PKCS #7 wrapped X.509 certificate
- o IKEv2's Hash and URL of X.509 certificate

The use of the other types:

- o X.509 Certificate - Key Exchange
- o PGP Certificate
- o DNS Signed Key
- o Kerberos Tokens
- o SPKI Certificate
- o X.509 Certificate Attribute
- o IKEv2's Raw RSA Key
- o IKEv2's Hash and URL of X.509 bundle

are out of the scope of this document.

3.3.2 X.509 Certificate - Signature

This type specifies that Certificate Data contains a certificate used for signing. Implementations SHOULD only send an end entity signature certificate.

3.3.3 Revocation Lists (CRL and ARL)

These types specify that Certificate Data contains an X.509 CRL or ARL. These types SHOULD NOT be sent in IKE. See Section 3.2.3 for discussion.

3.3.4 IKEv2's Hash and URL of X.509 certificate

This type specifies that Certificate Data contains a hash and the URL to a repository where an X.509 certificate can be retrieved.

3.3.5 PKCS #7 wrapped X.509 certificate

This type defines a particular encoding, not a particular certificate type. Implementations SHOULD NOT generate CERTs that contain this Certificate Type. Implementations SHOULD accept CERTs that contain this Certificate Type because several implementations are known to generate them. Note that those implementations may include entire certificate hierarchies inside a single CERT PKCS #7 payload, which violates the requirement specified in ISAKMP that this payload contain a single certificate.

3.3.6 Certificate Payloads Not Mandatory

An implementation which does not receive any CERTREQs during an exchange SHOULD NOT send any CERT payloads, except when explicitly configured to proactively send CERT payloads in order to interoperate with non-compliant implementations. This MUST NOT be the default behavior of implementations.

Implementations whose local security policy configuration expects that a peer must receive certificates through out-of-band means SHOULD ignore any CERTREQ messages that are received.

Implementations that receive CERTREQs from a peer which contain only unrecognized Certification Authorities SHOULD NOT continue the exchange, in order to avoid unnecessary and potentially expensive cryptographic processing, denial of service (resource starvation) attacks.

3.3.7 Response to Multiple Certificate Authority Proposals

In response to multiple CERTREQs which contain different Certificate Authority identities, implementations MAY respond using an end entity certificate which chains to a CA that matches any of the identities provided by the peer.

3.3.8 Using Local Keying Materials

Implementations MAY elect to skip parsing or otherwise decoding a given set of CERTs if equivalent keying materials are available via some preferable means, such as the case where certificates from a previous exchange have been cached.

3.3.9 Robustness

3.3.9.1 Unrecognized or Unsupported Certificate Types

Implementations MUST be able to deal with receiving CERTs with unrecognized or unsupported Certificate Types. Implementations MAY discard such payloads, depending on local policy. ISAKMP [5] Section 5.10 "Certificate Request Payload Processing" specifies additional processing.

3.3.9.2 Undecodable Certificate Data Fields

Implementations MUST be able to deal with receiving CERTs with undecodable Certificate Data fields. Implementations MAY discard such payloads, depending on local policy. ISAKMP specifies other actions which may be taken.

3.3.9.3 Ordering of Certificate Payloads

For IKEv1, implementations MUST NOT assume that CERTs are ordered in any way. For IKEv2, implementations MUST NOT assume that any except the first CERT is ordered in any way. IKEv2 specifies that the first CERT contain the end entity certificate which is to be used to authenticate the peer.

3.3.9.4 Duplicate Certificate Payloads

Implementations MUST support receiving multiple identical CERTs during an exchange.

3.3.9.5 Irrelevant Certificates

Implementations MUST be prepared to receive certificates and CRLs which are not relevant to the current exchange. Implementations MAY discard such extraneous certificates and CRLs.

Implementations MAY send certificates which are irrelevant to an exchange. One reason for including certificates which are irrelevant to an exchange is to minimize the threat of leaking identifying information in exchanges where CERT is not encrypted. It should be noted, however, that this probably provides rather poor protection against leaking the identity.

Another reason for including certificates that seem irrelevant to an exchange is that there may be two chains from the Certificate Authority to the end entity, each of which is only valid with certain validation parameters (such as acceptable policies). Since the end entity doesn't know which parameters the relying party is using, it should send the certs needed for both chains (even if there's only one CERTREQ).

Although implementations SHOULD NOT send multiple end entity certificates if the receipient cannot determine the correct certificate to use for authentication by using either the contents of the ID payload to match the certificate or, in IKEv2, the correct certificate is contained in the first CERT. In other words, receipients SHOULD NOT be expected to iterate over multiple end entity certs.

3.3.10 Optimizations

3.3.10.1 Duplicate Certificate Payloads

Implementations SHOULD NOT send duplicate CERTs during an exchange. Such payloads should be suppressed.

3.3.10.2 Send Only End Entity Certificates

When multiple CERTREQs are received which specify certificate authorities within the end entity certificate chain, implementations SHOULD send always and only the relevant end entity certificate, as chaining will take place out-of-band of IKE, between the IPsec peer

and the PKI system. Implementations SHOULD NOT send the chain.

3.3.10.3 Ignore Duplicate Certificate Payloads

Implementations MAY employ local means to recognize CERTs that have been received in the past, whether part of the current exchange or not, for which keying material is available and SHOULD discard these duplicate CERTs.

3.3.10.4 Hash Payload

IKEv1 specifies the optional use of the Hash Payload to carry a pointer to a certificate in either of the Phase 1 public key encryption modes. This pointer is used by an implementation to locate the end entity certificate that contains the public key that a peer should use for encrypting payloads during the exchange.

Implementations SHOULD include this payload whenever the public portion of the keypair has been placed in a certificate.

4. Profile of PKIX

Except where specifically stated in this document, implementations MUST conform to the requirements of PKIX [7].

4.1 X.509 Certificates

4.1.1 Versions

Although PKIX states that "implementations SHOULD be prepared to accept any version certificate", in practice this profile requires certain extensions that necessitate the use of Version 3 certificates for all but self-signed certificates used as trust anchors. Implementations that conform to this document MAY therefore reject Version 1 and Version 2 certificates in all other cases.

4.1.2 Subject Name

Certificate Authority implementations MUST be able to create certificates with Subject Name fields with at least the following four attributes: CN, C, O, OU. Implementations MAY support other Subject Name attributes as well. The contents of these attributes SHOULD be configurable on a certificate by certificate basis, as these fields will likely be used by IKE implementations to match SPD policy.

See Section 3.1.5 for details on how IKE implementations need to be able to process Subject Name field attributes for SPD policy lookup.

4.1.2.1 Empty Subject Name

Implementations MUST accept certificates which contain an empty Subject Name field, as specified in PKIX. Identity information in

such certificates will be contained entirely in the SubjectAltName extension.

4.1.2.2 Specifying Hosts and FQDN in Subject Name

Implementations which desire to place host names that are not intended to be processed by recipients as FQDNs (for instance "Gateway Router") in the Subject Name MUST use the commonName attribute.

While nothing prevents an FQDN, USER_FQDN, or IP address information from appearing somewhere in the Subject Name contents, such entries MUST NOT be interpreted as identity information for the purposes of matching with IKE_ID or for policy lookup.

If the FQDN is intended to be processed as identity for the purposes IKE_ID matching, it MUST be placed in the dNSName field of the SubjectAltName extension. Implementations MUST NOT populate the Subject Name in place of populating the dNSName field of the SubjectAltName extension.

4.1.2.3 EmailAddress

As specified in PKIX, implementations MUST NOT populate DistinguishedNames with the EmailAddress attribute.

4.1.3 X.509 Certificate Extensions

Conforming applications MUST recognize extensions which must or may be marked critical according to this specification. These extensions are: KeyUsage, SubjectAltName, and BasicConstraints.

Implementations SHOULD generate certificates such that the extension criticality bits are set in accordance with PKIX and this document. With respect to PKIX compliance, implementations processing certificates MAY ignore the value of the criticality bit for extensions that are supported by that implementation, but MUST support the criticality bit for extensions that are not supported by that implementation. That is, if an implementation supports (and thus is going to process) a given extension, then it isn't necessary to reject the certificate if the criticality bit is different from

what PKIX states it must be. However, if an implementation does not support an extension that PKIX mandates be critical, then the implementation must reject the certificate.

implements	bit in cert	PKIX mandate	behavior
yes	true	true	ok
yes	true	false	ok or reject
yes	false	true	ok or reject
yes	false	false	ok
no	true	true	reject
no	true	false	reject
no	false	true	reject
no	false	false	ok

4.1.3.1 AuthorityKeyIdentifier and SubjectKey ID

Implementations SHOULD NOT assume that other implementations support the AuthorityKeyIdentifier and SubjectKey ID extensions, and thus

SHOULD NOT generate certificate hierarchies which are overly complex to process in the absence of this extension, such as those that require possibly verifying a signature against a large number of similarly named CA certificates in order to find the CA certificate which contains the key that was used to generate the signature.

4.1.3.2 KeyUsage

IKE uses an end-entity certificate in the authentication process. The end-entity certificate may be used for multiple applications. As such, the CA can impose some constraints on the manner that a public key ought to be used. The key usage and extended key usage extensions apply in this situation.

Since we are talking about using the public key to validate a signature, if the key usage extension is present, then at least one of the digitalSignature (0) or the nonRepudiation (1) bit in the key usage extension MUST be set (both can be set as well). It is also fine if other key usage bits are set.

A summary of the logic flow for peer cert validation follows:

- If told (by configuration) to ignore KeyUsage (KU), accept cert regardless of its markings.
- If no KU extension, accept cert.
- If KU present and doesn't mention digitalSig or nonRepudiation, (both, in addition to other KUs, is also fine), reject cert.
- If none of the above, accept cert.

4.1.3.3 PrivateKeyUsagePeriod

PKIX recommends against the use of this extension. The PrivateKeyUsageExtension is intended to be used when signatures will need to be verified long past the time when signatures using the private keypair may be generated. Since IKE SAs are short-lived relative to the intended use of this extension in addition to the fact that each signature is validated only a single time, the usefulness of this extension in the context of IKE is unclear. Therefore, implementations MUST NOT generate certificates that contain the PrivateKeyUsagePeriod extension. If an implementation receives a certificate with this set, it SHOULD ignore it.

4.1.3.4 Certificate Policies

Many IPsec implementations do not currently provide support for the Certificate Policies extension. Therefore, implementations that generate certificates which contain this extension SHOULD NOT mark the extension as critical.

4.1.3.5 PolicyMappings

Many implementations do not support the PolicyMappings extension.

4.1.3.6 SubjectAltName

Deployments that intend to use an IKE_ID of either FQDN, USER_FQDN or IP*_ADDR MUST issue certificates with the corresponding SujectAltName fields populated with the same data. Implementations SHOULD generate only the following GeneralName choices in the subjectAltName extension, as these choices map to legal IKEv1/ISAKMP/IKEv2 Identification Payload types: rfc822Name, dNSName, or iPAddress.

Although it is possible to specify any GeneralName choice in the Identification Payload by using the ID_DER_ASN1_GN ID type, implementations SHOULD NOT assume that a peer supports such functionality, and SHOULD NOT generate certificates that do so.

4.1.3.6.1 dNSName

This field MUST contain a fully qualified domain name. If IKE ID type equals FQDN then the dNSName field MUST match its contents. Implementations MUST NOT generate names that contain wildcards. Implementations MAY treat certificates that contain wildcards in this field as syntactically invalid.

Although this field is in the form of an FQDN, implementations SHOULD NOT assume that this field contains an FQDN that will resolve via the DNS, unless this is known by way of some out-of-band mechanism. Such a mechanism is out of the scope of this document. Implementations SHOULD NOT treat the failure to resolve as an error.

4.1.3.6.2 iPAddress

If IKE ID type equals IP*_ADDR then the iPAddress field MUST match its contents. Note that although PKIX permits CIDR [10] notation in the "Name Constraints" extension, PKIX explicitly prohibits using CIDR notation for conveying identity information. In other words, the CIDR notation MUST NOT be used in the subjectAltName extension.

4.1.3.6.3 rfc822Name

If IKE ID type equals USER_FQDN then the rfc822Name field MUST match its contents. Although this field is in the form of an Internet mail address, implementations SHOULD NOT assume that this field contains a valid email address, unless this is known by way of some out-of-band mechanism. Such a mechanism is out of the scope of this document.

4.1.3.7 IssuerAltName

Implementations SHOULD NOT assume that other implementations support the IssuerAltName extension, and especially should not assume that information contained in this extension will be displayed to end

users.

4.1.3.8 SubjectDirectoryAttributes

The SubjectDirectoryAttributes extension is intended to contain privilege information, in a manner analogous to privileges carried in Attribute Certificates. Implementations MAY ignore this extension when it is marked non-critical, as PKIX mandates.

4.1.3.9 BasicConstraints

PKIX mandates that CA certificates contain this extension and that it be marked critical. Implementations SHOULD reject CA certificates that do not contain this extension. For backwards compatibility, implementations may accept such certificates if explicitly configured to do so, but the default for this setting MUST be to reject such certificates.

4.1.3.10 NameConstraints

Many implementations do not support the NameConstraints extension. Since PKIX mandates that this extension be marked critical when present, implementations which intend to be maximally interoperable SHOULD NOT generate certificates which contain this extension.

4.1.3.11 PolicyConstraints

Many implementations do not support the PolicyConstraints extension. Since PKIX mandates that this extension be marked critical when present, implementations which intend to be maximally interoperable SHOULD NOT generate certificates which contain this extension.

4.1.3.12 ExtendedKeyUsage

The CA SHOULD NOT include the ExtendedKeyUsage (EKU) extension in certificates for use with IKE. Note that there were three IPsec related object identifiers in EKU that were assigned in 1999. The semantics of these values were never clearly defined. The use of these three EKU values in IKE/IPsec is obsolete and explicitly deprecated by this specification. CAs SHOULD NOT issue certificates for use in IKE with them. (For historical reference only, those values were id-kp-ipsecEndSystem, id-kp-ipsecTunnel, and id-kp-ipsecUser.)

PKIX [7] section 4.2.1.13 states, "If a CA includes extended key usages to satisfy such applications, but does not wish to restrict usages of the key, the CA can include the special keyPurposeID anyExtendedKeyUsage. If the anyExtendedKeyUsage keyPurposeID is present, the extension SHOULD NOT be critical."

The CA SHOULD NOT mark the EKU extension in certificates for use with IKE and one or more other applications. If the CA administrator feels they must use an EKU for some other application, then such certificates MUST contain the keyPurposeID anyExtendedKeyUsage as well as the keyPurposeID values associated with the other applications for which the certificate is intended to be used. Recall however, EKU extensions in certificates meant for use in IKE are NOT RECOMMENDED.

A summary of the logic flow for peer certificate validation regarding the EKU extension follows:

- If told (by configuration) to ignore ExtendedKeyUsage (EKU), accept cert regardless of the presence or absence of the extension.
- If no EKU extension, accept cert.
- If EKU present AND anyExtendedKeyUsage is included, accept cert.
- Otherwise, reject cert.

4.1.3.13 CRLDistributionPoints

Because this document deprecates the sending of CRLs in band, the use of CRLDistributionPoints (CDP) becomes very important if CRLs are used for revocation checking (as opposed to say Online Certificate Status Protocol - OCSP [12]). The IPsec peer either needs to have a URL for a CRL written into its local configuration, or it needs to learn it from CDP. Therefore, implementations SHOULD issue certificates with a populated CDP.

Failure to validate the CRLDistributionPoints/ IssuingDistributionPoint pair can result in CRL substitution where an entity knowingly substitutes a known good CRL from a different distribution point for the CRL which is supposed to be used which would show the entity as revoked.

Implementations MUST support validating that the contents of CRLDistributionPoints match those of the IssuingDistributionPoint to prevent CRL substitution when the issuing CA is using them. At least one CA is known to default to this type of CRL use. See Section 4.2.2.5 for more information.

CDPs SHOULD be "resolvable". For example some very prominent implementations are well known for including CDPs like

http://localhost/path_to_CRL and http:///path_to_CRL which are as bad as not including the CDP.

See PKIX docs for CRLDistributionPoints intellectual property rights (IPR) information. Note that both the CRLDistributionPoints and IssuingDistributionPoint extensions are RECOMMENDED but not REQUIRED by PKIX, so there is no requirement to license any IPR.

4.1.3.14 InhibitAnyPolicy

Many implementations do not support the InhibitAnyPolicy extension. Since PKIX mandates that this extension be marked critical when present, implementations which intend to be maximally interoperable SHOULD NOT generate certificates which contain this extension.

4.1.3.15 FreshestCRL

Implementations MUST NOT assume that the FreshestCRL extension will exist in peer extensions. Note that most implementations do not support delta CRLs.

4.1.3.16 AuthorityInfoAccess

PKIX defines the AuthorityInfoAccess extension, which is used to indicate "how to access CA information and services for the issuer of the certificate in which the extension appears." Because this document deprecates the sending of CRLs in band, the use of AuthorityInfoAccess (AIA) becomes very important if OCSP [12] is to be used for revocation checking (as opposed to CRLs). The IPsec peer either needs to have a URI for the OCSP query written into its local configuration, or it needs to learn it from AIA. Therefore, implementations SHOULD support this extension, especially if OCSP will be used.

4.1.3.17 SubjectInfoAccess

PKIX defines the SubjectInfoAccess private certificate extension, which is used to indicate "how to access information and services for the subject of the certificate in which the extension appears." This extension has no known use in the context of IPsec. Conformant implementations SHOULD ignore this extension when present

4.2 X.509 Certificate Revocation Lists

When validating certificates, implementations MUST make use of certificate revocation information, and SHOULD support such revocation information in the form of CRLs, unless non-CRL revocation information is known to be the only method for transmitting this

information. Deployment that intend to use CRLs for revocation MUST populate the CRLDistributionPoint field. Therefore implementations MUST support issuing certificates with this field populated according to administrator's needs. Implementations MAY provide a configuration option to disable use of certain types of revocation information, but that option MUST be off by default. Such an option is often valuable in lab testing environments.

4.2.1 Multiple Sources of Certificate Revocation Information

Implementations which support multiple sources of obtaining certificate revocation information MUST act conservatively when the information provided by these sources is inconsistent: when a certificate is reported as revoked by one trusted source, the certificate MUST be considered revoked.

4.2.2 X.509 Certificate Revocation List Extensions

4.2.2.1 AuthorityKeyIdentifier

Implementations SHOULD NOT assume that other implementations support the AuthorityKeyIdentifier extension, and thus SHOULD NOT generate certificate hierarchies which are overly complex to process in the absence of this extension.

4.2.2.2 IssuerAltName

Implementations SHOULD NOT assume that other implementations support the IssuerAltName extension, and especially should not assume that information contained in this extension will be displayed to end users.

4.2.2.3 CRLNumber

As stated in PKIX, all issuers conforming to PKIX MUST include this

extension in all CRLs.

4.2.2.4 DeltaCRLIndicator

4.2.2.4.1 If Delta CRLs Are Unsupported

Implementations that do not support delta CRLs MUST reject CRLs which contain the DeltaCRLIndicator (which MUST be marked critical according to PKIX) and MUST make use of a base CRL if it is available. Such implementations MUST ensure that a delta CRL does not "overwrite" a base CRL, for instance in the keying material database.

4.2.2.4.2 Delta CRL Recommendations

Since some implementations that do not support delta CRLs may behave incorrectly or insecurely when presented with delta CRLs, administrators and deployers SHOULD consider whether issuing delta CRLs increases security before issuing such CRLs.

And, if all the elements in the VPN and PKI systems do not adequately support Delta CRLs, then their use should be questioned.

The authors are aware of several implementations which behave in an incorrect or insecure manner when presented with delta CRLs. See Appendix B for a description of the issue. Therefore, this specification RECOMMENDS NOT issuing delta CRLs at this time. On the other hand, failure to issue delta CRLs exposes a larger window of vulnerability. See the Security Considerations section of PKIX [7] for additional discussion. Implementors as well as administrators are encouraged to consider these issues.

4.2.2.5 IssuingDistributionPoint

A CA that is using CRLDistributionPoints may do so to provide many "small" CRLs, each only valid for a particular set of certificates issued by that CA. To associate a CRL with a certificate, the CA places the CRLDistributionPoints extension in the certificate, and places the IssuingDistributionPoint in the CRL. The distributionPointName field in the CRLDistributionPoints extension MUST be identical to the distributionPoint field in the

IssuingDistributionPoint extension. At least one CA is known to default to this type of CRL use. See Section 4.1.3.13 for more information.

4.2.2.6 FreshestCRL

Given the recommendations against implementations generating delta CRLs, this specification RECOMMENDS that implementations do not populate CRLs with the FreshestCRL extension, which is used to obtain delta CRLs.

5. Configuration Data Exchange Conventions

Below we present a common format for exchanging configuration data. Implementations MUST support these formats, MUST support arbitrary whitespace at the beginning and end of any line, MUST support arbitrary line lengths although they SHOULD generate lines less than 76 characters, and MUST support the following three line-termination disciplines: LF (US-ASCII 10), CR (US-ASCII 13), and CRLF.

5.1 Certificates

Certificates MUST be Base64 encoded and appear between the following delimiters:

-----BEGIN CERTIFICATE-----

-----END CERTIFICATE-----

5.2 Public Keys

Implementations MUST support two forms of public keys: certificates

and so-called "raw" keys. Certificates should be transferred in the same form as above. A raw key is only the SubjectPublicKeyInfo portion of the certificate, and MUST be Base64 encoded and appear between the following delimiters:

-----BEGIN PUBLIC KEY-----

-----END PUBLIC KEY-----

5.3 PKCS#10 Certificate Signing Requests

A PKCS#10 [6] Certificiate Signing Request MUST be Base64 encoded and appear between the following delimiters:

-----BEGIN CERTIFICATE REQUEST-----

-----END CERTIFICATE REQUEST-----

6. Security Considerations

6.1 Identification Payload

Depending on the exchange type, ID may be passed in the clear. Administrators in some environments may wish to use the empty Certification Authority option to prevent such information from leaking, at the possible cost of some performance, although such use is discouraged.

6.2 Certificate Request Payload

The Contents of CERTREQ are not encrypted in IKE. In some environments this may leak private information. Administrators in

some environments may wish to use the empty Certification Authority option to prevent such information from leaking, at the cost of performance.

6.3 Certificate Payload

Depending on the exchange type, CERTs may be passed in the clear and therefore may leak identity information.

6.4 IKEv1 Main Mode

Certificates may be included in any message, and therefore implementations may wish to respond with CERTs in a message that offers privacy protection, in Main Mode messages 5 and 6. Implementations may not wish to respond with CERTs in the second message, thereby violating the identity protection feature of Main Mode in IKEv1.

7. Intellectual Property Rights

No new intellectual property rights are introduced by this document.

8. IANA Considerations

There are no known numbers which IANA will need to manage.

9. References

9.1 Normative References

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

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

[3] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", draft-ietf-ipsec-ikev2-15 (work in progress), August 2004.

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

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

[6] Kaliski, B., "PKCS #10: Certification Request Syntax Version 1.5", RFC 2314, March 1998.

[7] Housley, R., Polk, W., Ford, W. and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

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

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

9.2 Informative References

[10] Fuller, V., Li, T., Yu, J. and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", RFC 1519, September 1993.

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

[12] Myers, M., Ankney, R., Malpani, A., Galperin, S. and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate

Status Protocol - OCSP", RFC 2560, June 1999.

[13] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 1883, December 1995.

[14] Arsenault, A. and S. Turner, "Internet X.509 Public Key Infrastructure:Roadmap", draft-ietf-pkix-roadmap-09 (work in progress), July 2002.

[15] Lynn, C., "X.509 Extensions for IP Addresses and AS Identifiers", draft-ietf-pkix-x509-ipaddr-as-extn-03 (work in progress), September 2003.

Author's Address

Brian Korver
Xythos Software, Inc.
One Bush Street, Suite 600
San Francisco, CA 94104
US

Phone: +1 415 248 3800
EMail: briank@xythos.com

Appendix A. Change History

* August 2004 (-02) (Edited by Gregory Lebovitz, with XML formatting and cross-referencing by Paul Knight)

3.1.1 the text between the **s was added to paragraph, per the question that arose in IETF60 WG session: Implementations MUST be capable of verifying that the address contained in the ID is the same as the peer source address **contained in the outer most IP header**.

3.2.7 - added HTTP_CERT_LOOKUP_SUPPORTED to this section and described its use - #38

3.3 - changed back sending of intermediate CA certificates from SHOULD NOT to MAY (for backward compatibility). Added text to explain further why we want to stay away from actually doing it though.

3.3.8 - changed text per Knowles/Korver 2004.07.28.

3.3.9.5 - Change discard of Irrelevant Certificates from may to SHOULD - #23(Kent 2004.04.26)

4.1.3.2 - KU - re-worked to reflect discussion on list and in IETF60 - #36

4.1.3.12 - EKU - re-worked to reflect discussion on list and in IETF60 - #36

[IKEv2] - update the reference to the -14 draft of May 29, 2004

* July 2004 (-01) (Edited by Gregory Lebovitz)

Changed ISAKMP references in Abstract and Intro to IKE.

Editorial changes to make the text conform with the summary table in 3.1, especially in the text following the table in 3.1. Particular note should be paid to changes in section 3.5.1.

Sect 3.1.1 - editorial changes to aid in clarification. Added text on when deployers might consider using IP addr, but strongly encouraged not to.

Sect 3.1.8 - removed IP address from list of practically used ID types.

3.1.9 overhauled (per Kivinen, July 18)

3.2 - added IKEv2's Hash and URL of x.509 to list of those profiled and gave it its own section, now 3.2.5

- added note in CRL/ARL section about revocation occurring OOB of IKE

- deleted ARL as its own section and collapsed it into Revocation Lists (CRL and ARL) for consciseness. Renumbered accordingly.

Sect 3.2.7.2 - Changed from MUST not send empty certreqs to SHOULD send CERTREQs which contain CA fields with direction on how, but MAY send empty CERTREQs in certain case. Use case added, and specifics of both initiator and responder behavior listed.

APPENDIX C added to fill out the explanation (mostly discussion from list).

3.3 - clarified that sending CRLs and chaining certs is deprecated.

- added IKEv2's Hash and URL of x.509 to list of those profiled and gave it its own section. Condensed ARL into CRL and renumbered

accordingly.

- duplicate section was removed, renumbered accordingly

3.3.10.2 - title changed. sending chaining becomes SHOULD NOT.

4.1.2 added text to explicity call out support for CN, C, O, OU

collapsed 4.1.2.3 into 4.1.2.2 and renumbered accordingly.

Collapsed 4.1.3.2 into 4.1.3.1 and renumbered accordingly

Edited 4.1.3.2 Key Usage and 4.1.3.12 ExtKey Usage according to Hoffman, July18

4.1.3.3 if receive cert w/ PKUP, ignore it.

4.1.3.13 - CDP changed text to represent SHOULD issue, and how important CDP becomes when we do not send CRLs in-band. Added SHOULD for CDPs actually being resolvable (reilly email).

Reordered 6.4 for better clarity.

Added Rescorla to Acknowledgements section, as he is no longer listed as an editor, since -00.

* May 2004 (renamed draft-ietf-pki4ipsec-ikecert-profile-00.txt) (edited by Brian Korver)

Made it clearer that the format of the ID_IPV4_ADDR payload comes from RFC791 and is nothing new. (Tero Kivinen Feb 29)

Permit implementations to skip verifying that the peer source address matches the contents of ID_IPV{4,6}_ADDR. (Tero Kivinen Feb 29, Gregory Lebovitz Feb 29)

Removed paragraph suggesting that implementations favor unauthenticated peer source addresses over an unauthenticated ID for initial policy lookup. (Tero Kivinen Feb 29, Gregory Lebovitz Feb 29)

Removed some text implying RSA encryption mode was in scope. (Tero Kivinen Feb 29)

Relaxed deprecation of PKCS#7 CERT payloads. (Tero Kivinen Feb 29)

Made it clearer that out-of-scope local heuristics should be used for picking an EE cert to use when generating CERTREQ, not when receiving CERTREQ. (Tero Kivinen Feb 29)

Made it clearer that CERT processing can be skipped when the contents of a CERT are already known. (Tero Kivinen Feb 29)

Implementations SHOULD generate BASE64 lines less than 76 characters. (Tero Kivinen Feb 29)

Added "Except where specifically stated in this document, implementations MUST conform to the requirements of PKIX" (Steve Hanna Oct 7, 2003)

RECOMMENDS against populating the ID payload with IP addresses due to interoperability issues such as problem with NAT traversal. (Gregory Lebovitz May 14)

Changed "as revoked by one source" to "as revoked by one trusted source". (Michael Myers, May 15)

Specifying Certificate Authorities section needed to be regularized with Gregory Lebovitz's CERT proposal from -04. (Tylor Allison, May 15)

Added text specifying how receipients SHOULD NOT be expected to iterate over multiple end-entity certs. (Tylor Allison, May 15)

Modified text to refer to IKEv2 as well as IKEv1/ISAKMP where

relevant.

IKEv2: Explained that IDr sent by responder doesn't have to match the [IDr] sent initiator in second exchange.

IKEv2: Noted that "The identity ... does not necessarily have to match anything in the CERT payload" (S3.5) is not contradicted by SHOULD in this document.

IKEv2: Noted that ID_USER_FQDN renamed to ID_RFC822_ADDR, and

ID_USER_FQDN would be used exclusively in this document.

IKEv2: Declared that 3 new CERTREQ and CERT types are not profiled in this document (well, at least not yet, pending WG discussion of what to do -- note that they are only SHOULDs in IKEv2).

IKEv2: Noted that CERTREQ payload changed from DN to SHA-1 of SubjectPublicKeyInfo.

IKEv2: Noted new requirement that specifies that the first certificate sent MUST be the EE cert (section 3.6).

* February 2004 (-04)

Minor editorial changes to clean up language

Deprecate in-band exchange of CRLs

Incorporated Gregory Lebovitz's proposal for CERT payloads: "should deal with all the CRL, Intermediat Certs, Trust Anchors, etc OOB of IKE; MUST be able to send and receive EE cert payload; only real exception is Intermediate Cets which MAY be sent and SHOULD be able to be receivable (but in reality there are very few hierarchies in operation, so really it's a corner case); SHOULD NOT send the other stuff (CRL, Trust Anchors, etc) in cert payloads in IKE; SHOULD be able to accept the other stuff if by chance it gets sent, though we hope they don't get sent"

Incorporated comments contained in Oct 7, 2003 email from steve.hanna@sun.com to ipsec@lists.tislabs.com

Moved text from "Profile of ISAKMP" Background section to each payload section (removing duplication of these sections)

Removed "Certificate-Related Playloads in ISAKMP" section since it was not specific to IKE.

Incorporated Gregory Lebovitz's table in the "Identification Payload" section

Moved text from "binding identity to policy" sections to each payload section

Moved text from "IKE" section into now-combined "IKE/ISAKMP" section

ID_USER_FQDN and ID_FQDN promoted to MUST from MAY

Promoted sending ID_DER_ASN1_DN to MAY from SHOULD NOT, and receiving from MUST from MAY

Demoted ID_DER_ASN1_GN to MUST NOT

Demoted populating Subject Name in place of populating the dNSName from SHOULD NOT to MUST NOT and removed the text regarding domainComponent

Revocation information checking MAY now be disabled, although not by default

Aggressive Mode removed from this profile

* June 2003 (-03)

Minor editorial changes to clean up language

Minor additional clarifying text

Removed hyphenation

Added requirement that implementations support configuration data exchange having arbitrary line lengths

* February 2003 (-02)

Word choice: move from use of "root" to "trust anchor", in accordance with PKIX

SBGP note and reference for placing address subnet and range information into certificates

Clarification of text regarding placing names of hosts into the Name

commonName attribute of SubjectName

Added table to clarify text regarding processing of the certificate extension criticality bit

Added text underscoring processing requirements for CRLDistributionPoints and IssuingDistributionPoint

* October 2002, Reorganization (-01)

* June 2002, Initial Draft (-00)

Appendix B. The Possible Dangers of Delta CRLs

The problem is that the CRL processing algorithm is sometimes written incorrectly with the assumption that all CRLs are base CRLs and it is assumed that CRLs will pass content validity tests. Specifically, such implementations fail to check the certificate against all possible CRLs: if the first CRL that is obtained from the keying material database fails to decode, no further revocation checks are performed for the relevant certificate. This problem is compounded by the fact that implementations which do not understand delta CRLs may fail to decode such CRLs due to the critical DeltaCRLIndicator extension. The algorithm that is implemented in this case is approximately:

- fetch newest CRL
- check validity of CRL signature
- if CRL signature is valid then
- if CRL does not contain unrecognized critical extensions
- and certificate is on CRL then
- set certificate status to revoked

The authors note that a number of PKI toolkits do not even provide a method for obtaining anything but the newest CRL, which in the presence of delta CRLs may in fact be a delta CRL, not a base CRL.

Note that the above algorithm is dangerous in many ways. See PKIX [7] for the correct algorithm.

Appendix C. More on Empty CERTREQs

Sending empty certificate requests is commonly used in implementations, and in the IPsec interop meetings, vendors have generally agreed that it means that send all/any certificates you have (if multiple certificates are sent, they must have same public key, as otherwise the other end does not know which key was used). For 99% of cases the client have exactly one certificate and public key, so it really doesn't matter, but the server might have multiple, thus it simply needs to say to the client, use any certificate you have. If we are talking about corporate vpns etc, even if the client have multiple certificates or keys, all of them would be usable when authenticating to the server, so client can simply pick one.

If there is some real difference on which cert to use (like ones giving different permissions), then the client MUST be configured anyways, or it might even ask the user which one to use (the user is the only one who knows whether he needs admin privileges, thus needs to use admin cert, or is the normal email privileges ok, thus using email only cert).

99% of the cases the client have exactly one certificate, so it will send it. In 90% of the rest of the cases, any of the certificates is ok, as they are simply different certificates from same CA, or different CAs for the same corporate VPN, thus any of them is ok.

Sending empty certificate requests has been agreed there to mean "give me a cert; any cert".

Justification:

- Responder first does all it can to send a certreq with a CA, check for IP match in SPD, have a default set of CAs to use in ambiguous cases, etc.

- o sending empty certreq's is fairly common in implementations today, and is generally accepted to mean "send me a cert, any cert that works for you"
- o saves responder sending potentially 100's of certs, the fragmentation problems that follow, etc.
- o in +90% of use cases, Initiators have exactly 1 cert
- o in +90% of the remaining use cases, the multiple certs it has are issued by the same CA
- o in the remaining use case(s) -- if not all the others above -- the Initiator will be configured explicitly with which cert to send, so responding to an empty certreq is easy.

The following example shows why initiators need to have sufficient policy definition to know which certificate to use for a given connection it initiates.

EXAMPLE: Your client (initiator) is configured with VPN policies for gateways A and B (representing perhaps corporate partners).

The policies for the two gateways look something like:

```
      Acme Company policy (gateway A)
         Engineering can access 10.1.1.0
              Trusted CA: CA-A, Trusted Users: OU=Engineering
      Partners can access 20.1.1.0
                Trusted CA: CA-B, Trusted Users: OU=AcmePartners

      Bizco Company policy (gateway B)
        sales can access 30.1.1.0
                Trusted CA: CA-C, Trusted Users: OU=Sales
        Partners can access 40.1.1.0
                Trusted CA: CA-B, Trusted Users: OU=BizcoPartners
```

You are an employee of Acme and you are issued the following certificates:

- o From CA-A: CN=JoeUser,OU=Engineering
- o From CA-B: CN=JoePartner,OU=BizcoPartners

The client MUST be configured locally to know which CA to use when connecting to either gateway. If your client is not configured to

know the local credential to use for the remote gateway, this scenario will not work either. If you attempt to connect to Bizco, everything will work... as you are presented with responding with a certificate signed by CA-B or CA-C... as you only have a certificate from CA-B you are OK. If you attempt to connect to Acme, you have an issue because you are presented with an ambiguous policy selection. As the initiator, you will be presented with certificate requests from both CA A and CA B. You have certificates issued by both CAs, but only one of the certificates will be usable. How does the client know which certificate it should present? It must have sufficiently clear local policy specifying which one credential to present for the connection it initiates.

Appendix D. Acknowledgements

The authors would like to acknowledge the expired draft-ietf-ipsec-pki-req-05.txt for providing valuable materials for this document, especially Eric Rescorla, one of its original authors.

The authors would like to especially thank Greg Carter, Russ Housley, Steve Hanna, and Gregory Lebovitz for their valuable comments, some of which have been incorporated unchanged into this document.

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any

assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

Copyright Statement

Acknowledgment

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