INTERNET-DRAFT
Expires April 2001
Document: <draft-dukes-ike-mode-cfg-00.txt>
D. Dukes
R. Pereira
Cisco Systems
October 2000

The ISAKMP Configuration Method
<draft-dukes-ike-mode-cfg-00.txt>

Status of this Memo

Abstract

This document describes a new ISAKMP method that allows configuration items to be exchanged securely by using both push/acknowledge or request/reply paradigms.

The authors currently intend this document to be published as an Informational RFC, not a standards-track document, so that the many IPsec implementations that have implemented to earlier drafts of this protocol can have a single stable reference.

Comments regarding this draft should be sent to ietf-mode-cfg@vpnc.org or to the authors.

Conventions used 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 [2].

Table of Contents

1. Introduction

The ISAKMP protocol provides a framework to negotiate and generate Security Associations. While negotiating SAs, it is sometimes quite useful to retrieve certain information from the other peer before the non-ISAKMP SA can be established. Luckily, ISAKMP is also flexible enough to provide configuration information and do it securely. This document will present a mechanism to extend ISAKMP to provide such functionality.

1.1. Changes since last revision.

The last revision of this document was published as "draft-ietf-ipsec-isakmp-mode-cfg-06.txt"

o Prepared this document for submission as informational and renamed it.

o Added INTERNAL_IP6_SUBNET attribute.

1.2. Reader Prerequisites

It is assumed that the reader is familiar with the terms and concepts described in the "Security Architecture for the Internet Protocol" [ArchSec] and "IP Security Document Roadmap" [Thayer97] documents.

Readers are advised to be familiar with both [IKE] and [ISAKMP] because of the terminology used within this document and the fact that this document is an extension of both of those documents.

2. Configuration Transaction

A "Configuration Transaction" is defined as two configuration exchanges, the first being either a Set or a Request and the second being either an Acknowledge or a Reply, respectively. A common identifier is used to identify the transaction between exchanges.

There are two paradigms to follow for this method.

o "Request/Reply" allows a host to request information from an informed hosts (a configuration manager). If the attributes in the Request message are not empty, then these attributes are taken as suggestions for that attribute. The Reply message MAY wish to choose those values, or return new values. It MAY also add new attributes and not include some requested attributes.

A Reply MUST always be sent when a Request is received, even if it is an empty Reply or if there are missing attributes in the Request.

This merely means that the requested attributes were not available or unknown.

```
    Initiator                Responder
    ---------------          --------------
    REQUEST           -->
                      <--    REPLY
```

o "Set/Acknowledge" works on the push principle that allows a configuration manager (a host that wishes to send information to another host) to start the configuration transaction. This code sends attributes that it wants the peer to alter. The Acknowledge code MUST return the zero length attributes that it accepted. Those attributes that it did not accept will NOT be sent back in the acknowledgement.

```
    Initiator                Responder
    ---------------          -------------------
    SET               -->
                      <--    ACKNOWLEDGE
```

Transactions are completed once the Reply or Acknowledge code is received. If one is not received, the implementation MAY wish to retransmit the original exchange as detailed in a later section.

The initiator and responder are not necessarily the same as the initiator and responder of the ISAKMP exchange.

3. Configuration Method Exchange and Payload

3.1. Transaction Exchanges

A new exchange mode is required for the configuration method. This exchange is called the "Transaction Exchange" and has a value of 6. This exchange is quite similar to the Information exchange described in [ISAKMP] and [IKE], but allows for multi-exchange transactions instead of being a one-way transmittal of information.

This specification protects ISAKMP Transaction Exchanges when possible.

3.1.1. Protected Exchanges

Once an ISAKMP security association has been established (and SKEYID_e and SKEYID_a have been generated), the ISAKMP Transaction Exchange is as follows:

```
     Initiator                         Responder
    -----------                       -----------
     HDR*, HASH, ATTR      -->
                           <--         HDR*, HASH, ATTR
```


Where the HASH payload contains the prf output, using SKEYID_a as the key, and the M-ID (ISAKMP header Message ID) unique to this exchange concatenated with all of the payloads after the HASH payload. In other words, the hash for the above exchange is:

```
    HASH = prf( SKEYID_a, M-ID | ATTR )
```

Multiple ATTR payloads MAY NOT be present in the Transaction Exchange.

As noted, the message ID in the ISAKMP header-- as used in the prf computation-- is unique to this exchange and MUST NOT be the same as the message ID of another exchange. The derivation of the initialization vector (IV) for the first message, used with SKEYID_e to encrypt the message, is described in Appendix B of [IKE]. Subsequent IVs are taken from the last ciphertext block of the previous message as described in [IKE].

3.1.2. Unprotected Exchanges

If the ISAKMP security association has not yet been established at the time of the Transaction Exchange and the information being exchanged is not sensitive, the exchange MAY be done in the clear without an accompanying HASH payload.

```
     Initiator                        Responder
    -----------                      -----------
     HDR, ATTR          -->
                        <--          HDR, ATTR
```

Multiple ATTR payloads MAY NOT be present in the Transaction Exchange.

3.2. Attribute Payload

A new payload is defined to carry attributes as well as the type of transaction message.

```
                       1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Next Payload  !   RESERVED    !         Payload Length        !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !     Type      !   RESERVED    !           Identifier          !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !                                                               !
   ~                           Attributes                          ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```


The Attributes Payload fields are defined as follows:

o Next Payload (1 octet) - Identifier for the payload type of the next payload in the message. If the current payload is the last in the message, then this field will be 0.

o RESERVED (1 octet) - Unused, set to 0.

o Payload Length (2 octets) - Length in octets of the current payload, including the generic payload header, the transaction-specific header and all attributes. If the length does not match the length of the payload headers plus the attributes, (i.e. an attribute is half contained within this payload) then entire payload MUST be discarded.

o Attribute Message Type (1 octet) - Specifies the type of message represented by the attributes. These are defined in the next section.

o RESERVED (1 octet) - Unused, set to 0.

o Identifier (2 octets) - An identifier used to reference a configuration transaction within the individual messages.

o Attributes (variable length) - Zero or more ISAKMP Data Attributes as defined in [ISAKMP]. The attribute types are defined in a later section.

The payload type for the Attributes Payload is 14.

3.3. Configuration Message Types

These values are to be used within the Type field of an Attribute ISAKMP payload.

Types	Value
RESERVED	0
ISAKMP_CFG_REQUEST	1
ISAKMP_CFG_REPLY	2
ISAKMP_CFG_SET	3
ISAKMP_CFG_ACK	4
Reserved for Future Use	5-127
Reserved for Private Use	128-255

Messages with unknown types SHOULD be silently discarded.

3.4. Configuration Attributes

Zero or more ISAKMP attributes [ISAKMP] are contained within an Attributes Payload. Zero length attribute values are usually sent in a Request and MUST NOT be sent in a Response.

All IPv6 specific attributes are mandatory only if the implementation supports IPv6 and vice versa for IPv4. Mandatory attributes are stated below.

Unknown private attributes SHOULD be silently discarded.

The following attributes are currently defined:

Attribute	Value	Type	Length
RESERVED	0		
INTERNAL_IP4_ADDRESS	1	Variable	0 or 4 octets
INTERNAL_IP4_NETMASK	2	Variable	0 or 4 octets
INTERNAL_IP4_DNS	3	Variable	0 or 4 octets
INTERNAL_IP4_NBNS	4	Variable	0 or 4 octets
INTERNAL_ADDRESS_EXPIRY	5	Variable	0 or 4 octets
INTERNAL_IP4_DHCP	6	Variable	0 or 4 octets
APPLICATION_VERSION	7	Variable	0 or more
INTERNAL_IP6_ADDRESS	8	Variable	0 or 16 octets
INTERNAL_IP6_NETMASK	9	Variable	0 or 16 octets
INTERNAL_IP6_DNS	10	Variable	0 or 16 octets
INTERNAL_IP6_NBNS	11	Variable	0 or 16 octets
INTERNAL_IP6_DHCP	12	Variable	0 or 16 octets
INTERNAL_IP4_SUBNET	13	Variable	0 or 8 octets
SUPPORTED_ATTRIBUTES	14	Variable	0 or multiples of 2
INTERNAL_IP6_SUBNET	15	Variable	0 or 17 octets
Reserved for future use	16-16383		
Reserved for private use	16384-32767		

o INTERNAL_IP4_ADDRESS, INTERNAL_IP6_ADDRESS - Specifies an address within the internal network. This address is sometimes called a red node address or a private address and MAY be a private address on the Internet. Multiple internal addresses MAY be requested by requesting multiple internal address attributes. The responder MAY only send up to the number of addresses requested.

The requested address is valid until the expiry time defined with the INTERNAL_ADDRESS EXPIRY attribute or until the ISAKMP SA that was used to secure the request expires. The address MAY also expire when the IPSec (phase 2) SA expires, if the request is associated with a phase 2 negotiation. If no ISAKMP SA was used to secure the request, then the response MUST include an
expiry or the host MUST expire the SA after an implementation-defined time.

An implementation MUST support this attribute.

o INTERNAL_IP4_NETMASK, INTERNAL_IP6_NETMASK - The internal network's netmask. Only one netmask is allowed in the request and

reply messages (e.g. 255.255.255.0) and it MUST be used only with an INTERNAL_ADDRESS attribute.

An implementation MUST support this attribute.

o INTERNAL_IP4_DNS, INTERNAL_IP6_DNS - Specifies an address of a DNS server within the network. Multiple DNS servers MAY be requested. The responder MAY respond with zero or more DNS server attributes.

o INTERNAL_IP4_NBNS, INTERNAL_IP6_NBNS - Specifies an address of a NetBios Name Server (WINS) within the network. Multiple NBNS servers MAY be requested. The responder MAY respond with zero or more NBNS server attributes.

o INTERNAL_ADDRESS_EXPIRY - Specifies the number of seconds that the host can use the internal IP address. The host MUST renew the IP address before this expiry time. Only one attribute MAY be present in the reply.

An implementation MUST support this attribute.

o INTERNAL_IP4_DHCP, INTERNAL_IP6_DHCP - Instructs the host to send any internal DHCP requests to the address contained within the attribute. Multiple DHCP servers MAY be requested. The responder MAY respond with zero or more DHCP server attributes.

o APPLICATION_VERSION - The version or application information of the IPSec host. This is a string of printable ASCII characters that is NOT null terminated.

This attribute does not need to be secured.

An implementation MUST support this attribute.

o INTERNAL_IP4_SUBNET - The protected sub-networks that this edge-device protects. This attribute is made up of two fields; the first being an IP address and the second being a netmask. Multiple sub-networks MAY be requested. The responder MAY respond with zero or more sub-network attributes.

An implementation MUST support this attribute.

o SUPPORTED_ATTRIBUTES - When used within a Request, this attribute must be zero length and specifies a query to the responder to reply back with all of the attributes that it supports. The response contains an attribute that contains a set of attribute identifiers each in 2 octets. The length divided by 2 (bytes) would state the number of supported attributes contained in the response.

An implementation MUST support this attribute.

o INTERNAL_IP6_SUBNET - The protected sub-networks that this edge-device protects. This attribute is made up of two fields; the first being a 16 octet IPv6 address the second being a one octet prefix-mask as defined in [ADDRIPV6]. Multiple sub-networks MAY be requested. The responder MAY respond with zero or more sub-network attributes.

An implementation MUST support this attribute.

Note that no recommendations are made in this document how an implementation actually figures out what information to send in a reply. i.e. we do not recommend any specific method of (an edge device) determining which DNS server should be returned to a requesting host.

3.5. Retransmission

Retransmission SHOULD follow the same retransmission rules used with standard ISAKMP messages.

4. Exchange Positioning

The exchange and messages defined within this document MAY appear at any time. Because of security considerations with most attributes, the exchange SHOULD be secured with an ISAKMP phase 1 SA.

Depending on the type of transaction and the information being exchanged, the exchange MAY be dependant on an ISAKMP phase 1 SA negotiation, a phase 2 SA negotiation, or none of the above.

The next section details specific functions and their position within an ISAKMP negotiation.

5. Specific Uses

The following descriptions detail how to perform specific functions using this protocol. Other functions are possible and thus this list is not a complete list of all of the possibilities. While other functions are possible, the functions listed below MUST be performed as detailed in this document to preserve interoperability among different vendor's implementations.

5.1. Requesting an Internal Address

This function provides address allocation to a remote host trying to tunnel into a network protected by an edge device. The remote host requests an address and optionally other information concerning the internal network from the edge device. The edge device procures an internal address for the remote host from any number of sources such as a DHCP/BOOTP server or an its own address pool.


```
 Initiator                           Responder
----------------------------       ------------------------------
 HDR*, HASH, ATTR1(REQUEST)    -->
                               <--   HDR*, HASH, ATTR2(REPLY)
```

ATTR1(REQUEST) MUST contain at least an INTERNAL_ADDRESS attribute (either IPv4 or IPv6) but MAY also contain any number of additional attributes that it wants returned in the response.

For example:

```
ATTR1(REQUEST) =
  INTERNAL_ADDRESS(0.0.0.0)
  INTERNAL_NETMASK(0.0.0.0)
  INTERNAL_DNS(0.0.0.0)

ATTR2(REPLY) =
  INTERNAL_ADDRESS(192.168.219.202)
  INTERNAL_NETMASK(255.255.255.0)
  INTERNAL_SUBNET(291.168.219.0/255.255.255.0)
```

All returned values will be implementation dependent. As can be seen in the above example, the edge device MAY also send other attributes that were not included in the REQUEST and MAY ignore the non-mandatory attributes that it does not support.

This Transaction Exchange MUST occur after an ISAKMP phase 1 SA is already established and before an ISAKMP phase 2 negotiation has started, since that negotiation requires the internal address.

```
Initial Negotiation:
  MainMode or AggressiveMode
  TransactionMode (IP Address request)
  QuickMode(s)
```

Subsequent address requests would be done without the phase 1 negotiation when there already exists a phase 1 SA.

```
Subsequent Negotiations:
  TransactionMode (IP Address request)
  QuickMode(s)
```

5.2. Requesting the PeerÆs Version

An IPSec host wishing to inquire about the other peer's version information (with or without security) MUST use this method.

```
 Initiator                            Responder
-----------------------------       --------------------------
 HDR, ATTR1(REQUEST)          -->
                              <--   HDR, ATTR2(REPLY)
```

ATTR1(REQUEST) =
  APPLICATION_VERSION("")

ATTR2(REPLY) =
  APPLICATION_VERSION("foobar v1.3beta, (c) Foo Bar Inc.")

The return text string will be implementation dependent. This transaction MAY be done at any time and with or without any other ISAKMP exchange and because the version information MAY be deemed not sensitive, security is optional.

6. Enterprise Management Considerations

The method defined in this document SHOULD NOT be used for wide scale management. Its main intent is to provide a bootstrap mechanism to exchange information within IPSec. While it MAY be useful to use such a method of exchange information to some outlying IPSec hosts or small networks, existing management protocols such as DHCP [DHCP], RADIUS [RADIUS], SNMP or LDAP [LDAP] should be considered for enterprise management as well as subsequent information exchanges.

7. Security Considerations

This entire draft discusses a new ISAKMP configuration method to allow IPSec-enabled entities to acquire and share configuration information.

The draft mandates that this exchange should normally occur after the Phase I Security Association has been set up and that the entire exchange be protected by that Phase I SA. Thus the exchange is as secure as any Phase II SA negotiation.

This exchange MAY be secured (encrypted and authenticated) by other means as well, such as pre-configured ESP or data-link security.

8. References

[1] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

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

[ArchSec] S. Kent, R. Atkinson, "Security Architecture for the Internet Protocol", RFC2401

[ISAKMP] D. Maughan, M. Schertler, M. Schneider, J. Turner, "Internet Security Association and Key Management Protocol", RFC2408

[IKE] D. Harkins, D. Carrel, "The Internet Key Exchange

(IKE)", RFC2409

[DHCP] R. Droms, "Dynamic Host Configuration Protocol", RFC2131

[RADIUS] C. Rigney, A. Rubens, W. Simpson, S. Willens, "Remote Authentication Dial In User Service (RADIUS)", RFC2138

[LDAP] M. Wahl, T. Howes, S. Kille., "Lightweight Directory Access Protocol (v3)", RFC2251

[ESP] S. Kent, "IP Encapsulating Security Payload (ESP)", RFC2406

[ADDRIPV6] Hinden, R., "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

9. Acknowledgments

The editors would like to thank Sanjay Anand, Baiju V. Patel, Stephane Beaulieu, Tim Jenkins, Peter Ford, Bob Moskowitz and Shawn Mamros.

10. Author's Addresses

Darren Dukes
Cisco Systems Co.
365 March Road

Kanata, ON, Canada
Phone: 1-613-271-3679
Email: ddukes@cisco.com

Roy Pereira
Cisco Systems, Inc.
170 Tasman Drive
San Jose, CA, USA
Phone: 1-408-526-6793
Email: royp@cisco.com

Full Copyright Statement

This Internet Draft expires April 2001