IPSEC Working Group
INTERNET-DRAFT
draft-ietf-ipsec-isakmp-oakley-01.txt
Expire in six months

D. Harkins, D. Carrel, Editors
cisco Systems
November 1996

The resolution of ISAKMP with Oakley
<draft-ietf-ipsec-isakmp-oakley-01.txt>

Status of this Memo

1. Abstract

[MSST96] (ISAKMP) provides a framework for authentication and key exchange but does not define them. ISAKMP is designed to be key exchange independant; that is, it is designed to support many different key exchanges.

[Orm96] (Oakley) describes a series of key exchanges-- called "modes"-- and details the services provided by each (e.g. perfect forward secrecy for keys, identity protection, and authentication).

This document describes a proposal for using the Oakley Key Exchange Protocol in conjunction with ISAKMP to obtain authenticated keying material for use with ISAKMP, and for other security associations such as AH and ESP for the IETF IPsec DOI.

2. Discussion

This draft combines ISAKMP and Oakley. The purpose is to negotiate, and provide authenticated keying material for, security associations

in a protected manner.

Processes which implement this draft can be used for negotiating virtual private networks (VPNs) and also for providing a remote user from a remote site (whose IP address need not be known beforehand) access to a secure host or network.

Proxy negotiation is supported. Proxy mode is where the negotiating parties are not the endpoints for which security association negotiation is taking place. When used in proxy mode, the identities of the end parties remain hidden.

This proposal does not implement the entire Oakley protocol, but only a subset necessary to satisfy its goals. It does not claim conformance or compliance with the entire Oakley protocol. For greater understanding of the Oakley protocol and the mathematics behind it, please refer to [Orm96].

3. Terms and Definitions

3.1 Requirements Terminology

In this document, the words that are used to define the significance of each particular requirement are usually capitalised. These words are:

- MUST

  This word or the adjective "REQUIRED" means that the item is an absolute requirement of the specification.

- SHOULD

  This word or the adjective "RECOMMENDED" means that there might exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before taking a different course.

- MAY

  This word or the adjective "OPTIONAL" means that this item is truly optional. One vendor might choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

3.2 Notation

The following notation is used throughout this draft.

HDR is an ISAKMP header whose exchange type is the mode. When writen as HDR* it indicates payload encryption.

SA is an SA negotiation payload with one or more proposals. An initiator MAY provide multiple proposals for negotiation; a responder MUST reply with only one.

SAp is the entire body of the SA payload (minus the SA header)-- i.e. all proposals and transforms offered by the Initiator.

KE is the key exchange payload.

Nx is the nonce payload; x can be: i or r for the ISAKMP initiator and responder respectively.

IDx is the identity payload for "x". x can be: "ii" or "ir" for the ISAKMP initiator and responder respectively during phase one negotiation; or "ui" or "ur" for the user initiator and responder respectively during phase two. The ID payload format for the Internet DOI is defined in [Pip96].

SIG is the signature payload. The data to sign is exchange-specific.

CERT is the certificate payload.

HASH is the hash payload. The contents of the hash are exchange specific.

prf() is the keyed hash function negotiated as part of the ISAKMP SA.

SKEYID_e is the keying material used by the ISAKMP SA to protect it's messages.

SKEYID_a is the keying material used by the ISAKMP SA to authenticate it's messages.

<x>y indicates that "x" is encrypted with the key "y".

--> signifies "initiator to responder" communication (requests).

<-- signifies "responder to initiator" communication (replies).

| signifies concatenation of information-- e.g. X | Y is the concatentation of X with Y.

[x] indicates that x is optional.

Payload encryption (when noted by a '*' after the ISAKMP header) MUST begin immediately after the ISAKMP header. When communication is protected, all payloads following the ISAKMP header MUST be encrypted. Encryption keys are generated from SKEYID_e in a manner that is defined for each algorithm.

3.3 Perfect Forward Secrecy

When used in the draft Perfect Forward Secrecy (PFS) refers to the notion that compromise of a single key will permit access to only data protected by a single key. For PFS to exist the key used to protect transmission of data MUST NOT be used to derive any additional keys, and if the key used to protect transmission of data was derived from some other keying material, that material MUST NOT be used to derive any more keys.

Perfect Forward Secrecy for both keys and identities is provided in this protocol. (Sections 5.7 and 7).

3.4 Security Association

A security association (SA) is a set of policy and key used to protect information. The ISAKMP SA is the shared policy and key used by the negotiating peers in this protocol to protect their communication.

4. Introducttion

Oakley defines a method to establish an authenticated key exchange. This includes how payloads are constructed, the information they carry, the order in which they are processed and how they are used.

While Oakley defines "modes", ISAKMP defines "phases". The relationship between the two is very straightforward. ISAKMP's phase 1 is where the two ISAKMP peers establish a secure, authenticated channel with which to communicate. This is called the ISAKMP Security Association (SA). Oakley's "Main Mode" and "Aggressive Mode" each accomplish a phase 1 exchange. "Main Mode" and "Aggressive Mode" MUST ONLY be used in phase 1.

Phase 2 is where Security Associations are negotiated on behalf of services such as AH, ESP or any other service which needs key material and/or parameter negotiation. Oakley's "Quick Mode" accomplishes a phase 2 exchange. "Quick Mode" MUST ONLY be used in phase 2.

Oakley's "New Group Mode" is not really a phase 1 or phase 2. It follows phase 1, but serves to establish a new group which can be used in future negotiations. "New Group Mode" MUST ONLY be used in phase 2.

With the use of ISAKMP phases, an implementation can accomplish very fast keying when necessary. A single phase 1 negotiation may be used for more than one phase 2 negotiation. Additionally a single phase 2 negotiation can request multiple Security Associations. With these optimizations, an implementation can see less than one round trip per SA as well as less than one DH exponentiation per SA. "Main Mode" for phase 1 provides identity protection. When identity protection is not needed, "Aggressive Mode" can be used to reduce round trips even further. Developer hints for doing these optimizations are included below.

The following attributes are required by Oakley and MUST be negotiated as part of the ISAKMP Security Association. (These attributes pertain only to the ISAKMP Security Association and not to any Security Associations that ISAKMP may be negotiating on behalf of other services.)

- encryption algorithm (e.g. DES, IDEA, Blowfish).

- hash algorithm (e.g. MD5, SHA)

- authentication method (e.g. DSS sig, RSA sig, RSA encryption, pre-shared key)

- information about a group over which to do Diffie-Hellman.

The selected hash algorithm MUST support both keyed and unkeyed modes.

Oakley implementations MUST support the following default attributes:

- DES-CBC with a weak, and semi-weak, key check (weak and semi-weak keys are referenced in [Sch94] and listed in Appendix A). The key is derived according to Appendix B.

- MD5 and SHA in native and HMAC mode [KBC96].

- Authentication via pre-shared keys. The Digital Signature Standard SHOULD be supported; RSA SHOULD also be supported.

- MODP over the default Oakley group (see below). ECP and EC2N groups MAY be supported.

The Oakley modes described here MUST be implemented whenever the IETF IPsec DOI [Pip96] is implemented. Other DOIs MAY use the Oakley modes described here.

5. Exchanges

There are two basic methods used to establish an authenticated key exchange: Oakley Main Mode and Oakley Aggressive Mode. Each generates authenticated keying material from an ephemeral Diffie-Hellman exchange. Oakley in Main Mode MUST be implemented; Oakley Aggressive Mode SHOULD be implemented. In addition, Oakley Quick Mode MUST be implemented as a mechanism to generate fresh keying material and negotiate non-ISAKMP security services. In additon, Oakley New Group Mode SHOULD be implemented as a mechanism to define private groups for Diffie-Hellman exchanges. Implementations MUST NOT switch exchange types in the middle of an exchange.

Exchanges conform to standard ISAKMP [MSST96] payload syntax. attribute encoding, timeouts and retransmits of messages, and informational messages-- e.g a notify response is sent when, for example, a proposal is unacceptable, or a signature verification or decryption was unsuccessful, etc.

Oakley Main Mode is an instantiation of the ISAKMP Identity Protect Exchange: The first two messages negotiate policy; the next two exchange Diffie-Hellman public values and ancillary data (e.g. nonces) necessary for the exchange; and the last two messages authenticate the Diffie-Hellman Exchange. The authentication method negotiated as part of the initial ISAKMP exchange influences the

composition of the payloads but not their purpose. The XCHG for Oakley Main Mode is ISAKMP Identity Protect.

Similarly, Oakley Aggressive Mode is an instantiation of the ISAKMP Base Exchange. The first two messages negotiate policy, exchange Diffie-Hellman public values and ancillary data necessary for the exchange, and identities. In addition the second message authenticates the responder. The third message authenticates the initiator and provides a proof of participation in the exchange. The XCHG for Oakley Aggressive Mode is ISAKMP Base Exchange. The final message is not sent under protection of the ISAKMP SA allowing each party to postpone exponentiation, if desired, until negotiation of this exchange is complete.

The result of either exchange is two groups of authenticated keying material:

```
SKEYID_e = prf(g^xy, Ni | Nr | CKY-I | CKY-R | IDii | IDir | 0)
SKEYID_a = prf(g^xy, Ni | Nr | CKY-I | CKY-R | IDii | IDir | 1)
```

and agreed upon policy to protect further communications. The values of 0 and 1 above are represented by a single octet. The key used for encryption is derived from SKEYID_e in an algorithm-specific manner (see appendix B).

Quick Mode and New Group Mode have no analog in ISAKMP. The XCHG values for Quick Mode and New Group Mode are defined in Appendix A.

As mentioned above, the negotiated authentication method influences the content and use of messages for Phase 1 Oakley Modes, but not their intent. When using public keys for authentication, the Phase 1 Oakley can be accomplished either by using signatures or by using public key encryption (if the algorithm supports it). Following are Main Mode Exchanges with different authentication options.

5.1 ISAKMP/Oakley Phase 1 Authenticated With Signatures

Using signatures, the ancillary information exchanged during the second roundtrip are nonces; the exchange is authenticated by signing a mutually obtainable hash. Oakley Main Mode with signature authentication is described as follows:

```
     Initiator                          Responder
    ----------                         -----------
     HDR, SA                    -->
                                <--     HDR, SA
     HDR, KE, Ni                -->
                                <--     HDR, KE, Nr
     HDR*, IDii, [ CERT, ] SIG  -->
                                <--     HDR*, IDir, [ CERT, ] SIG
```

Oakley Aggressive mode with signatures in conjunction with ISAKMP is described as follows:

```
     Initiator                          Responder
    -----------                        -----------
     HDR, SA, KE, Ni, IDii      -->
                                <--     HDR, SA, KE, Nr, IDir,
                                             [ CERT, ] SIG
     HDR, [ CERT, ] SIG         -->
```

In both modes, the signed data in SIG is a signature of a keyed hash of the concatenation of the nonces, cookies, the entire SA offer-- everything following the SA header-- that was sent from Initiator to Responder, and the sender's ID, with g^xy as the key to the hash function. The order of the nonces, and cookies are specific to the direction. In other words, the sender signs, HASH_I, and the responder signs HASH_R where:

```
HASH_I = prf(g^xy, Ni | Nr | CKY-I | CKY-R | SAp | IDii))
HASH_R = prf(g^xy, Nr | Ni | CKY-R | CKY-I | SAp | IDir))
```

In general the keyed hash will be the HMAC version of the negotiated hash function. This can be overridden for construction of the signature if the signature algorithm is tied to a particular hash algorithm. In this case, the negotiated hash function would continue to be used for all other proscribed hashing functions.

One or more certificate payloads MAY be optionally passed.

5.2 Oakley Phase 1 Authenticated With Public Key Encryption

Using public key encryption to authenticate the exchange, the ancillary information exchanged is encrypted nonces. Each party's ability to reconstruct a hash (proving that the other party decrypted the nonce) authenticates the exchange.

In order to perform the public key encryption, the initiator must already have the responder's public key. In the case where a party has multiple public keys, a hash of the certificate the initiator is using to encrypt the ancillary information is passed as part of the third message. In this way the responder can determine which corresponding private key to use to decrypt the nonce and identity protection is retained.

In addition, the identities of the parties (IDii and IDir) are also encrypted with the other parties public key. If the authentication method is public key encryption, the nonce and identity payloads MUST be encrypted with the public key of the other party.

When using encrytion for authentication with Oakley, Main Mode is defined as follows.

```
     Initiator                          Responder
    -----------                        -----------
     HDR, SA                    -->
                                <--     HDR, SA
     HDR, KE, [ HASH(1), ]
       <IDii>PubKey_r,
        <Ni>PubKey_r            -->
                                        HDR, KE, <IDir>PubKey_r,
                                <--           <Nr>PubKey_i
     HDR*, HASH_I               -->
                                <--     HDR*, HASH_R
```

Oakley Aggressive Mode authenticated with encryption is described as follows:

```
     Initiator                          Responder
    -----------                        -----------
     HDR, SA, [ HASH(1),] KE,
       <IDii>Pubkey_r,
       <Ni>PubKey_r             -->
                                        HDR, SA, KE, <Nr>PubKey_i,
                                <--          IDir, HASH_R
     HDR, HASH_I                -->
```

Where HASH(1) is a hash (using the negotiated hash function) of the

certificate which the initiator is using to encrypt the nonce and identity. The contents of the other hashes (HASH_I and HASH_R) are the results of the HMAC version of the hash algorithm negotiated in the first roundtrip, with a concatenation of the nonces as the key and a concatenation of the shared secret, the cookies, the entire SA offer-- everything following the SA header-- that was sent from Initiator to Responder, and the sender's ID as the hashed data. The order of the nonces and cookies are specific to the direction. In other words:

```
HASH_I = prf(Ni | Nr, g^xy | CKY-I | CKY-R | SAp | IDii))
HASH_R = prf(Nr | Ni, g^xy | CKY-R | CKY-I | SAp | IDir))
```

Using encryption for authentication provides for a plausably deniable exchange. There is no proof (as with a digital signature) that the conversation ever took place since each party can completely reconstruct both sides of the exchange.

Note that, unlike other authentication methods, authentication with public key encryption allows for identity protection with Aggressive Mode.

5.3 Oakley Phase 1 Authenticated With a Pre-Shared Key

A key derived by some out-of-band mechanism may also be used to authenticate the exchange. The actual establishment of this key is out of the scope of this document.

When doing a pre-shared key authentication with Oakley, Main Mode is defined as follows

```
   Initiator                          Responder
  ----------                         -----------
   HDR, SA              -->
                        <--    HDR, SA
   HDR, KE, Ni          -->
                        <--    HDR, KE, Nr
   HDR*, IDii, HASH_I   -->
                        <--    HDR*, IDir, HASH_R
```

Oakley Aggressive mode with a pre-shared key is described as follows:

```
   Initiator                          Responder
  -----------                        -----------
   HDR, SA, KE, Ni, IDii -->
                        <--    HDR, SA, KE, Nr, IDir, HASH_R
   HDR, HASH_I          -->
```

The hash is the result of the HMAC version of the hash algorithm negotiated in the first roundtrip with the pre-shared key as the key to the HMAC, and a concatenation of the shared secret, the nonces, the cookies, the complete SA offer-- everything following the SA header-- sent from Initiator to Responder, and the sender's ID. The order of the cookies and nonces are specific to the direction. In other words,

```
   HASH_I = prf(pre-shared-key, g^xy | Ni | Nr | CKY-I | CKY-R | SAp
| IDii)
   HASH_R = prf(pre-shared-key, g^xy | Nr | Ni | CKY-R | CKY-I | SAp
| IDir)
```

5.4 Oakley Phase 2 - Quick Mode

Oakley Quick Mode is not a complete exchange itself, but is used as part of the ISAKMP SA negotiation process (phase 2) to derive keying material and negotiate shared policy for non-ISAKMP SAs. The information exchanged along with Oakley Quick Mode MUST be protected by the ISAKMP SA-- i.e. all payloads except the ISAKMP header are encrypted.

Quick Mode is essentially an exchange of nonces that provides replay protection. The nonces are used to generate fresh key material and prevent replay attacks from generating bogus security associations. An optional Key Exchange payload can be exchanged to allow for an additional Diffie-Hellman exchange and exponentiation per Quick Mode. While use of the key exchange payload with Quick Mode is optional it MUST be supported.

Base Quick Mode (without the KE payload) refreshens the keying material derived from the exponentiation in phase 1. This does not provide PFS. Using the optional KE payload, an additional exponentiation is performed and PFS is provided for the keying material. If a KE payload is sent, a Diffie-Hellman group (see section 5.5.1 and Appendix A) MUST be sent as attributes of the SA being negotiated.

Quick Mode is defined as follows:

```
     Initiator                          Responder
    -----------                        -----------
     HDR*, HASH(1), SA, Ni
       [, KE ] [, IDui, IDur ] -->
                                <--     HDR*, HASH(2), SA, Nr
                                              [, KE ] [, IDui, IDur ]
     HDR*, HASH(3)              -->
```

Where:
HASH(1) and HASH(2) are keyed hashes of the entire message that follows the hash including payload headers, and HASH(3)-- for liveliness-- is a keyed hash of the value zero represented as a single octet, followed by a concatenation of the two nonces. For example, the hashes for the above exchange would be:

```
HASH(1) = prf( SKEYID_a, SA | Ni [ | KE ] [ | IDui | IDur ] )
HASH(2) = prf( SKEYID_a, SA | Nr [ | KE ] [ | IDui | IDur ] )
HASH(3) = prf( SKEYID_a, 0 | Ni | Nr )
```

If PFS is not needed, and KE payloads are not exchanged, the new keying material is defined as KEYMAT = prf(SKEYID_e, Ni | Nr | 0).

If PFS is desired and KE payloads were exchanged, the new keying material is defined as KEYMAT = prf(g(qm)^xy, Ni | Nr | 0), where g(qm)^xy is the shared secret from the ephemeral Diffie-Hellman exchange of this Quick Mode.

In either case, 0 is represented by a single octet.

For situations where the amount of keying material desired is greater than that supplied by the prf, KEYMAT is expanded by concatenation and rehashing with a monotonically increasing number represented by a single octet, i.e.

```
KEYMAT = prf(SKEYID_e, Ni | Nr | 0) | prf(SKEYID_e, Ni | Nr | 1)
...
```

repeated until the required keying material has been reached.

This keying material (whether with PFS or without, and whether derived directly or through concatenation) MUST be used with the negotiated SA. It is up to the service to define how keys are derived from the keying material (see Appendix B).

In the case of an ephemeral Diffie-Hellman exchange in Quick Mode, the exponential (g(qm)^xy) is irretreivably removed from the current state and SKEYID_e and SKEYID_a (derived from phase 1 negotiation) continue to protect and authenticate the ISAKMP SA.

If ISAKMP is acting as a proxy negotiator on behalf of another party the identities of the parties MUST be passed as IDui and IDur. Local policy will dictate whether the proposals are acceptible for the identities specified. If IDs are not exchanged, the negotiation is assumed to be done on behalf of each ISAKMP peer. If an ID range (see Appendix A of [Pip96]) is not acceptable (for example, the specified subnet is too large) a BAD_ID_RANGE notify message followed

by an acceptible ID range, in an ID payload, MUST be sent.

Using Quick Mode, multiple SA's and keys can be negotiated with one exchange as follows:

```
     Initiator                          Responder
    -----------                        -----------
     HDR*, HASH(1), SA0, SA1, Ni,
       [, KE ] [, IDui, IDur ] -->
                                <--     HDR*, HASH(2), SA0, SA1, Nr,
                                           [, KE ] [, IDui, IDur ]
     HDR*, HASH(3)              -->
```

and the keying material for SAx is prf(SKEYID_e, Ni | Nr | x) where x is the number of the SA negotiated (starting with zero). (In the case where one, or all, of the SAs required keys longer than that supplied by the prf, the number merely monotonically increases for this entire exchange-- e.g. SA0 uses 0 and 1; SA1 uses 2 and 3; etc). For ease of processing the HASH payloads MUST immediately follow the ISAKMP header and precede all other payloads.

5.4 Oakley New Group Mode

Oakley New Group Mode MUST NOT be used prior to establishment of an ISAKMP SA. The description of a new group MUST only follow phase 1 negotiation. (It is not a phase 2 exchange, though).

```
     Initiator                          Responder
    -----------                        -----------
     HDR*, HASH(1), SA          -->
                                <--     HDR*, HASH(2), SA
```

where HASH(1) is a keyed hash, using SKEYID_a as the key, and the entire SA proposal, body and header, as the data; HASH(2) is a keyed hash, using SKEYID_a as the key, and the reply as the data.

The proposal will be an Oakley proposal which specifies the characteristics of the group (see appendix A, "Oakley Attribute Assigned Numbers"). Group descriptions for private Oakley Groups MUST be greater than or equal to 2^15. If the group is not acceptable, the responder MUST reply with a Notify payload with the message type set to GROUP_NOT_ACCEPTABLE (13).

ISAKMP implementations MAY require private groups to expire with the SA under which they were established.

Groups may be directly negotiated in the SA proposal with Oakley Main Mode. To do this the Prime, Generator, and Group Type are passed as SA attributes (see Appendix A in [MSST96]). Alternately, the nature of the group can be hidden using Oakley New Group Mode and only the

group identifier is passed in the clear during Main Mode.

5.5 Oakley Groups

[Orm96] defines several groups. The value 0 indicates no group. The value 1 indicates the default group described below. The attriute class for "Group" is defined in Appendix A. Other values are also defined in [Orm96]. All values 2^15 and higher are used for private group identifiers.

5.5.1 Oakley Default Group

Oakley implementations MUST support a MODP group with the following prime and generator. This group is assigned id 1 (one).

The prime is: 2^768 - 2 ^704 - 1 + 2^64 * { [2^638 pi] + 149686 }
Its hexadecimal value is

FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1
29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD
EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245
E485B576 625E7EC6 F44C42E9 A63A3620 FFFFFFFF FFFFFFFF

The generator is: 2.

other groups can be defined using Oakley New Group Mode. This default group was generated by Richard Schroeppel at the University of Arizona. Properties of this prime are described by the following exerpt from [Orm96]:

> The prime for this group was selected to have certain properties. The high order 64 bits are forced to 1. This helps the classical remainder algorithm, because the trial quotient digit can always be taken as the high order word of the dividend, possibly +1. The low order 64 bits are forced to 1. This helps the Montgomery-style remainder algorithms, because the multiplier digit can always be taken to be the low order word of the dividend. The middle bits are taken from the binary expansion of pi. This guarantees that they are effectively random, while avoiding any suspicion that the primes have secretly been selected to be weak.
>
> The prime is chosen to be a Sophie-Germain prime (i.e., (P-1)/2 is also prime), to have the maximum strength against the square-root attack. The starting trial numbers were repeatedly incremented by 2^64 until suitable primes were located.
>
> Because this prime is congruent to 7 (mod 8), 2 is a quadratic

residue. All powers of 2 will also be quadratic residues. This prevents an opponent from learning the low order bit of the Diffie-Hellman exponent. Using 2 as a generator is efficient for some modular exponentiation algorithms. [Note that 2 is technically not a generator in the number theory sense, because it omits half of the possible residues mod P. From a cryptographic viewpoint, this is a virtue.]

A further discussion of the properties of this group, the motivation behind its creation, as well as the definition of several more groups can be found in [Orm96].

5.6 Payload Explosion for Complete ISAKMP-Oakley Exchange

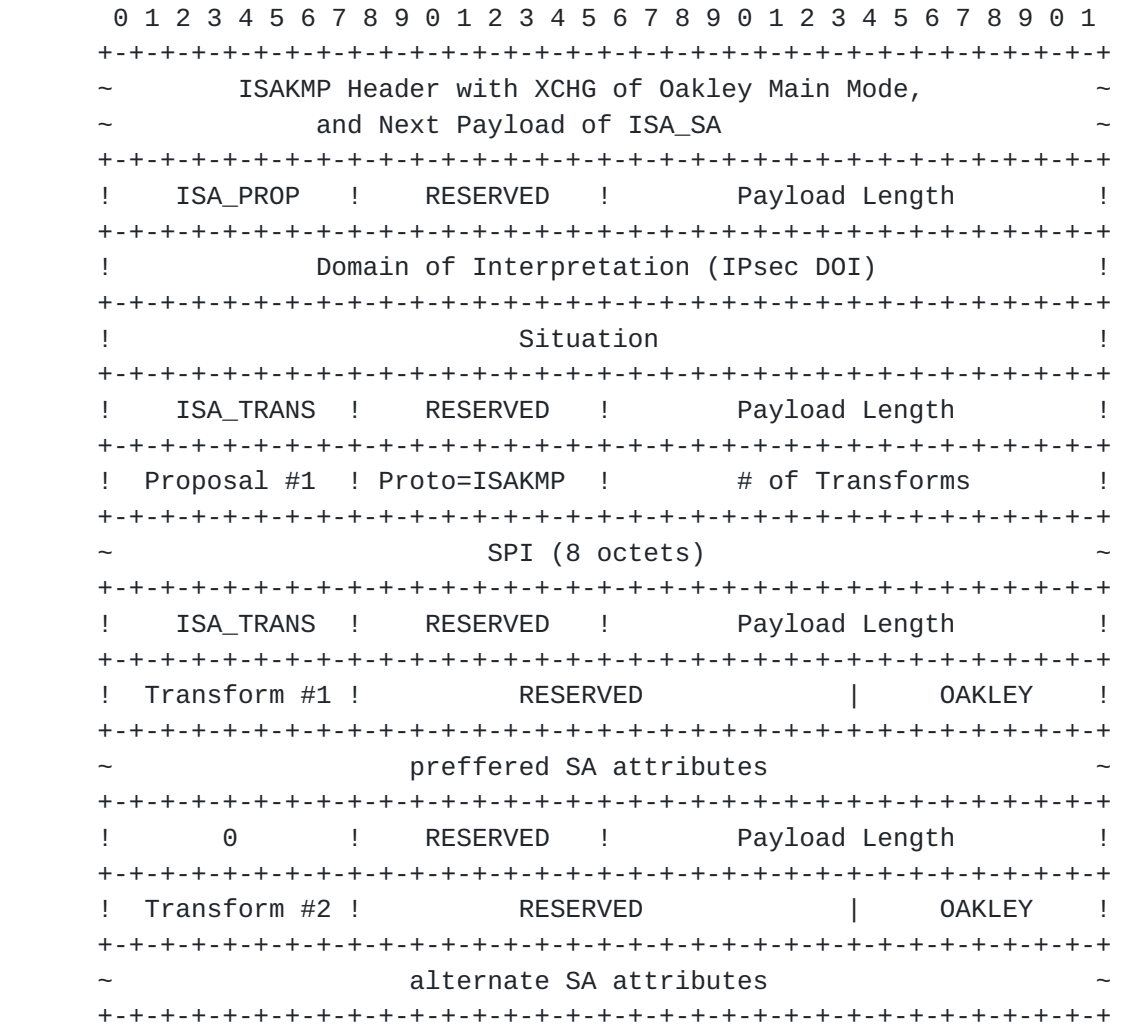
This section illustrates how ISAKMP payloads are used with Oakley to:

- establish a secure and authenticated channel between ISAKMP processes (phase 1); and

- generate key material for, and negotiate, an IPsec SA (phase 2).

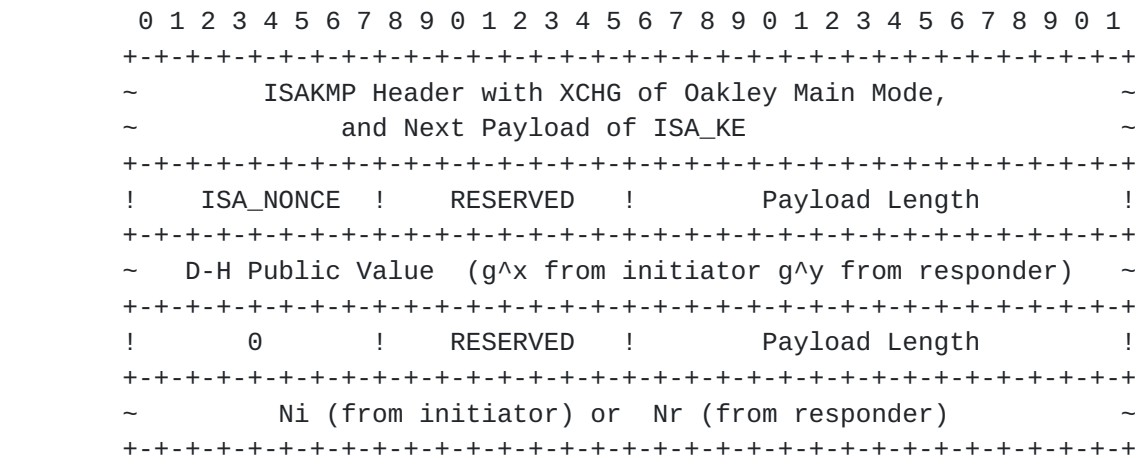
5.6.1 Phase 1 using Oakley Main Mode

The following diagram illustrates the payloads exchanged between the two parties in the first round trip exchange. The initiator MAY propose several proposals; the responder MUST reply with one.

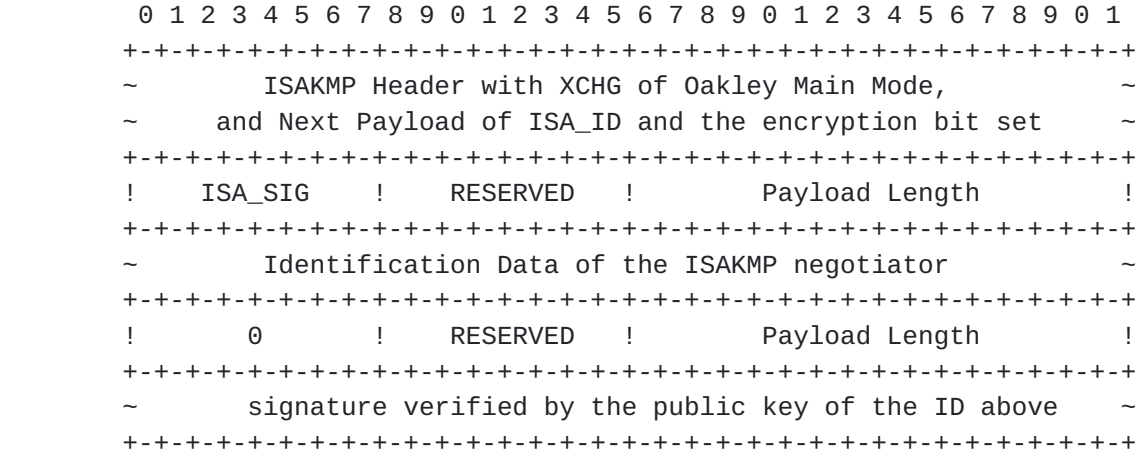


The responder replies in kind but selects, and returns, one transform proposal (the ISAKMP SA attributes).

The second exchange consists of the following payloads:



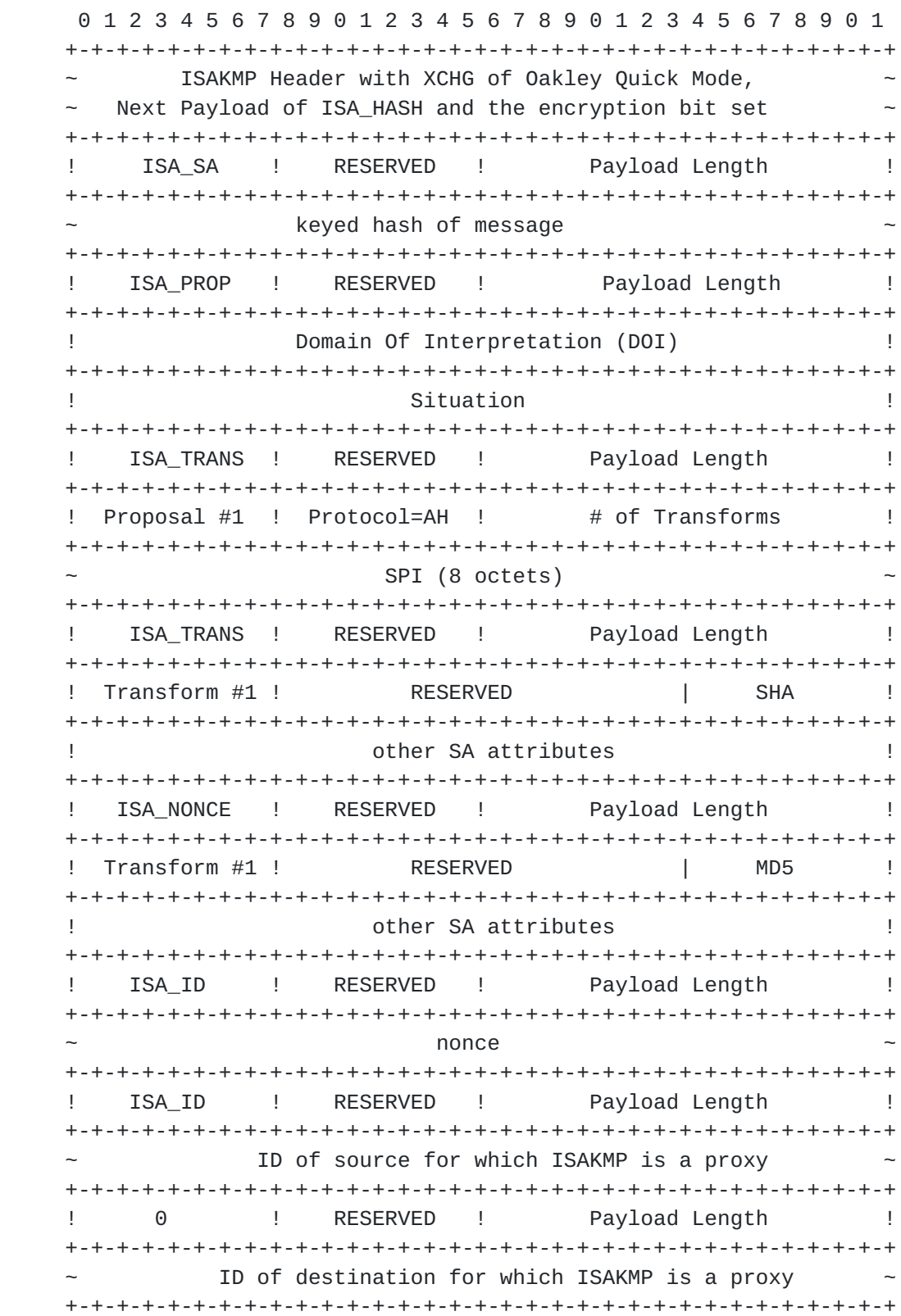
The shared keys, SKEYID_e and SKEYID_a, are now used to protect and authenticate all further communication. Note that both SKEYID_e and SKEYID_a are unauthenticated.



The key exchange is authenticated over a signed hash as described in section 5.1. Once the signature has been verified using the authentication algorithm negotiated as part of the ISAKMP SA, the shared keys, SKEYID_e and SKEYID_a can be marked as authenticated. (For brevity, certificate payloads were not exchanged).

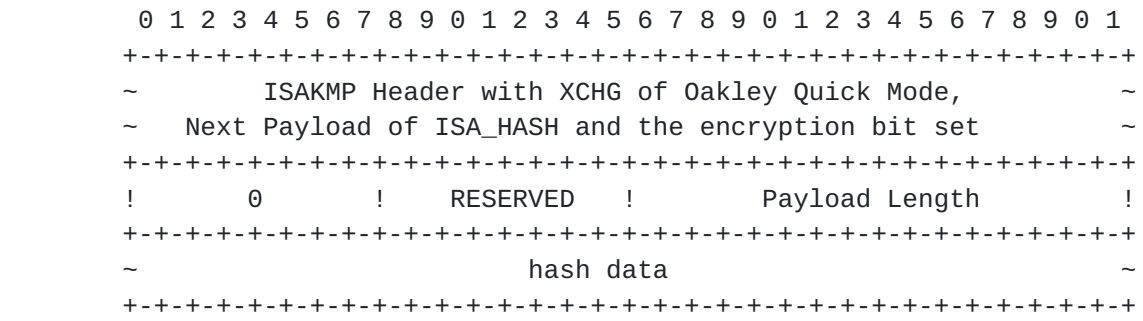
5.6.2 Phase 2 using Oakley Quick Mode

The following payloads are exchanged in the first round of Oakley Quick Mode with ISAKMP SA negotiation. In this hypothetical exchange, the ISAKMP negotiators are proxies for other parties which have requested authentication.



where the contents of the hash are described in 5.4 above. The responder replies with a similar message which only contains one

transform-- the selected AH transform. Upon receipt, the initiator can provide the key engine with the negotiated security association and the keying material. As a check against replay attacks, the responder waits until receipt of the next message.



where the contents of the hash are described in 5.4 above.

5.7 Perfect Forward Secrecy Example

This protocol can provide PFS of both keys and identities. The identies of both the ISAKMP negotiating peer and, if applicable, the proxy for whom the peers are negotiating can be protected with PFS.

To provide Perfect Forward Secrecy of both keys and all identities, two parties would perform the following:

- o A Main Mode Exchange to protect the identities of the ISAKMP peers.
  This establishes an ISAKMP SA.
- o A Quick Mode Exchange to negotiate other security protocol protection.
  This establishes a SA on each end for this protocol.
- o Delete the ISAKMP SA and its associated state.

Since the key for use in the non-ISAKMP SA was derived from the single ephemeral Diffie-Hellman exchange PFS is preserved.

To provide Perfect Forward Secrecy of merely the keys of a non-ISAKMP security association, it in not necessary to do a phase 1 exchange if an ISAKMP SA exists between the two peers. A single Quick Mode in which the optional KE payload is passed, and an additional Diffie-Hellman exchange is performed, is all that is required. At this point the state derived from this Quick Mode must be deleted from the ISAKMP SA as described in section 5.4.

6. Implementation Hints

Using a single ISAKMP Phase 1 negotiation makes subsequent Phase 2 negotiations extremely quick. As long as the Phase 1 state remains

cached, and PFS is not needed, Phase 2 can proceed without any exponentiation. How many Phase 2 negotiations can be performed for a single Phase 1 is a local policy issue. The decision will depend on the strength of the algorithms being used and level of trust in the peer system.

An implementation may wish to negotiate a range of SAs when performing Quick Mode. By doing this they can speed up the "re-keying". Quick Mode defines how KEYMAT is defined for a range of SAs. When one peer feels it is time to change SAs they simple use the next one within the stated range. A range of SAs can be established by negotiating multiple SAs (identical attributes, different SPIs) with one Quick Mode.

An optimization that is often useful is to establish Security Associations with peers before they are needed so that when they become needed they are already in place. This ensures there would be no delays due to key management before initial data transmission. This optimization is easily implemented by setting up more than one Security Association with a peer for each requested Security Association and caching those not immediately used.

Also, if ISAKMP is implementation is alerted that a SA will soon be needed (e.g. to replace an existing SA that will expire in the near future), then it can establish the new SA before that new SA is needed.

7. Security Considerations

This entire draft discusses a hybrid protocol, combining Oakley with ISAKMP, to negotiate, and derive keying material for, security associations in a secure and authenticated manner.

Confidentiality is assured by the use of a negotiated encryption algorithm. Authentication is assured by the use of a negotiated method: a digital signature algorithm; a public key algorithm which supports encryption; or, a pre-shared key. The confidentiality and authentication of this exchange is only as good as the attributes negotiated as part of the ISAKMP security association.

Repeated re-keying using Quick Mode can consume the entropy of the Diffie- Hellman shared secret. Implementors should take note of this fact and set a limit on Quick Mode Exchanges between exponentiations. This draft does not proscribe such a limit.

Perfect Forward Secrecy (PFS) of both keying material and identities is possible with this protocol. By specifying a Diffie-Hellman group, and passing public values in KE payloads, ISAKMP peers can establish

PFS of keys-- the identities would be protected by SKEYID_e from the ISAKMP SA and would therefore not be protected by PFS. If PFS of both keying material and identities is desired, an ISAKMP peer MUST establish only one non-ISAKMP security association (e.g. IPsec Security Association) per ISAKMP SA. PFS for keys and identities is accomplished by deleting the ISAKMP SA (and optionally issuing a DELETE message) upon establishment of the single non-ISAKMP SA. In this way a phase one negotiation is uniquely tied to a single phase two negotiation, and the ISAKMP SA established during phase one negotiation is never used again.

8. Acknowledgements

This document is the result of close consultation with Hilarie Orman, Douglas Maughan, Mark Schertler, Mark Schneider, and Jeff Turner. It relies completely on protocols which were written by them. Without their interest and dedication, this would not have been written.

We would also like to thank Cheryl Madson, Harry Varnis, Elfed Weaver, and Hugo Krawcyzk for technical input.

9. References

[KBC96] Krawczyk, H., Bellare, M., Canetti, R., "HMAC: Keyed-Hashing for Message Authentication", draft-ietf-ipsec-hmac-md5-01.txt

[Kra96] Krawcyzk, H., "SKEME: A Versatile Secure Key Exchange Mechanism for Internet", from IEEE Proceedings of the 1996 Symposium on Network and Distributed Systems Security.

[MSST96] Maughhan, D., Schertler, M., Schneider, M., and Turner, J., "Internet Security Association and Key Management Protocol (ISAKMP)", version 6, draft-ietf-ipsec-isakmp-06.{ps,txt}.

[Orm96] Orman, H., "The Oakley Key Determination Protocol", version 1, draft-ietf-ipsec-oakley-01.txt.

[Pip96] Piper, D., "The Internet IP Security Domain Of Interpretation for ISAKMP", version 1, draft-ietf-ipsec-ipsec-doi-01.txt.

[Sch94] Schneier, B., "Applied Cryptography, Protocols, Algorithms, and Source Code in C", 1st edition.

Appendix A

This is a list of DES Weak and Semi-Weak keys. The keys come from [Sch94]. All keys are listed in hexidecimal.

DES Weak Keys
0101 0101 0101 0101
1F1F 1F1F E0E0 E0E0
E0E0 E0E0 1F1F 1F1F
FEFE FEFE FEFE FEFE

DES Semi-Weak Keys
01FE 01FE 01FE 01FE
1FE0 1FE0 0EF1 0EF1
01E0 01E0 01F1 01F1
1FFE 1FFE 0EFE 0EFE
011F 011F 010E 010E
E0FE E0FE F1FE F1FE

FE01 FE01 FE01 FE01
E01F E01F F10E F10E
E001 E001 F101 F101
FE1F FE1F FE0E FE0E
1F01 1F01 0E01 0E01
FEE0 FEE0 FEF1 FEF1

Attribute Assigned Numbers

Attributes negotiated during phase one use the following definitions. Phase two attributes are defined in the applicable DOI specification (for example, IPsec attributes are defined in the IPsec DOI), with the exception of a group description when Quick Mode includes an ephemeral Diffie-Hellman exchange. Attribute types can be either Basic (B) or Variable-length (V). Encoding of these attributes is defined in the base ISAKMP specification.

Attribute Classes

class	value	type
Encryption Algorithm	1	B
Hash Algorithm	2	B
Authentication Method	3	B
Group Description	4	B
Group Type	5	B
Group Prime	6	V
Group Generator One	7	V

Group Curve A	9	V
Group Curve B	10	V
Life Type	11	B
Life Duration	12	B/V

Class Values

Encryption Algorithm

DEC-CBC	1
IDEA-CBC	2
Blowfish-CBC	3

values 4-65000 are reserved. Values 65001-65535 are for private use among mutually consenting parties.

Hash Algorithm

MD5	1
SHA	2
Tiger	3

values 4-65000 are reserved. Values 65001-65535 are for private use among mutually consenting parties.

Authentication Method

pre-shared key	1
DSS signatures	2
RSA signatures	3
RSA encryption	4

values 5-65000 are reserved. Values 65001-65535 are for private use among mutually consenting parties.

Group Description

default group (section 5.5.1)	1

values 2-32767 are reserved. Values 32768-65535 are for private use among mutually consenting parties.

Group Type

MODP (modular exponentiation group)	1
ECP (elliptic curve group)	2

Life Type

seconds	1
kilobytes	2

values 3-65000 are reserved. Values 65001-65535 are for private use among mutually consenting parties. For a given "Life Type"

the value of the "Life Duration" attribute defines the actual length of the SA life-- either a number of seconds, or a number of kbytes protected.

Additional Exchanges Defined-- XCHG values

Quick Mode	32
New Group Mode	33

Appendix B

Encryption keys used to protect the ISAKMP SA are derived from SKEYID_e in an algorithm-specific manner. When SKEYID_e is not long enough to supply all the necessary keying material an algorithm requires, the key is derived from a concatenation of SKEYID_e and successive keyed hashes of a single character which contains a monotonically increasing counter beginning at one (1), and SKEYID_e as the key, using the negotiated hash function.

For example, if (ficticious) algorithm AKULA requires 320-bits of key (and has no weak key check) and the prf used to generate SKEYID_e only generates 120 bits of material, the key for AKULA, would be the first 320-bits of Ka, where:

```
Ka = SKEYID_e | prf(SKEYID_e | 1) | prf(SKEYID_e | 2)
```

where prf is the HMAC version of the negotiated hash function. SKEYID_e provides 120-bits, and each of the two additional hashes provide 120-bits, for a total of 360 bits.

Material for the initialization vector (IV material) for CBC mode encryption algorithms is derived from a hash of a concatenation of the initiator's public Diffie-Hellman value and the responder's public Diffie-Hellman value using the negotiated hash algorithm.

The key for DES-CBC is derived from the first eight (8) non-weak and semi-weak (see Appendix A) bytes of SKEYID_e. The IV is the first 8 bytes of the IV material derived above.

The key for IDEA-CBC is derived from the first sixteen (16) bytes of SKEYID_e.
The IV is the first 8 bytes of the IV material derived above.

The key for Blowfish-CBC is derived from the first fifty-six (56) bytes of a key derived in the method described above, by concatenating successive hashes onto SKEYID_e until the requesite number of bytes has been achieved. The IV is the first 8 bytes of the IV material derived above.

Editors' Addresses:

Dan Harkins <dharkins@cisco.com>
Dave Carrel <carrel@cisco.com>
cisco Systems
170 W. Tasman Dr.
San Jose, California, 95134-1706
United States of America
+1 408 526 4000

Editors' Note:

The editors encourage independent implementation, and interoperability testing, of this hybrid exchange.