Final draft ETSI TS 103 523-5 V0.0.7 (2020-01)

CYBER;
Middlebox Security Protocol;
Part 5: Enterprise Network Security

Reference
DTS/CYBER-0027-5

Keywords
cyber security

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Cyber Security (CYBER).

The present document is part 5 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.1].

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Executive summary

Requirements - such as legal mandates and service agreements - exist for enterprise network and data centre operators and service providers, organizations, and small businesses to be able to observe and audit the content and metadata of encrypted sessions transported across their infrastructures. The IPsec protocol standards [1] [i.2] [i.3] can be managed in a way that enables this enterprise visibility.

The present document is one of a series of MSP implementation profiles that, to achieve these required visibility capabilities, puts the enterprise operators and users in control of the access to their data. It sets forth an "Enterprise Network Security" profile for use in enterprise networks and data centres that meets mandatory capabilities for the Middlebox Security Protocol (MSP) [i.1].

Introduction

The present document specifies the MSP profile for Enterprise Network Security, which is an implementation variant of Internet Key Exchange Protocol Version 2 (IKEv2) [1]. Hereafter this profile is referred to as "Enterprise Network Security" or ENS for brevity. This profile maintains interoperability with IPsec peers that are unaware of this profile; however, when a certificate is used, an extension provides notice that this profile is being used.

There are operational circumstances where passive decryption of IPsec-encrypted packets by authorized entities is a requirement. This decryption can be performed in real-time, or the packets can be captured, stored, and then decrypted later.

Situations requiring passive decryption of IPsec-encrypted packets generally occur in environments where both the communicating peers, and by inference the data being exchanged, are under the control of the same organization. IPsec encryption is often stipulated by internal or external security policies. Authorized access to the plaintext packet data is required for operational reasons, including:

- Application health monitoring and troubleshooting;
- Intrusion detection;
- Detection of malware activity, including lateral movement, command and control, and data exfiltration traffic;
- Detection of advanced distributed denial of service (DDOS) attacks; and
- Compliance audits.

One possible approach to decrypting IPsec-encrypted packets passively is to export the keying material generated for each Security Association (SA) to middleboxes. However, this approach has significant limitations: two in particular are as follows. Firstly, it is very difficult to ensure that the exported SA keying material will arrive at the middlebox in sufficient time to allow decryption in real-time. Secondly, the keying material needs to be correlated with every stored packet session in anticipation of post-capture decryption. For these reasons and more, this approach does not scale to the needs of a data centre.

The ENS profile therefore uses longer-lived static Diffie-Hellman public/private key pairs. These Diffie-Hellman keys are used to establish many SAs. This ensures that the static Diffie-Hellman public/private key pairs can be distributed to real-time decryption middleboxes prior to the arrival of network packets, and it greatly reduces the amount of keying material to be stored and correlated with packet storage systems. Enterprises can implement automated key rotation to frequently change the Diffie-Hellman private keys.

IKEv2 supports three forms of authentication: certificates and digital signatures, pre-shared keys, and the Extensible Authentication Protocol (EAP) [i.6]. In each case, the ENS profile requires notice to be provided that plaintext packets can be inspected. When the certificate [333] is used, the ENS profile includes the capability for such notice to be included in certificates, which indicates to the other IPsec peer that the corresponding static Diffie-Hellman private key is being used to enable visibility. This visibility information describes the set of entities or roles or domains, or any combination of these, for which the policy of the party signing the certificate allows sharing of the corresponding Diffie-Hellman private key.

Annex A describes an exception to the certificate visibility requirement. In these situations, notice of packet inspection will be provided in another way. One use case is where pre-shared keys or Extensible Authentication Protocol are used for authentication and there is no certificate. A second use case of this variant is when the IPsec peers are wholly within a private enterprise network and the operator of the peers has already been notified by alternative means, such as a condition of access to the network, that plaintext packets can be inspected.

The ENS profile is compatible with any IPsec peer that implements IKEv2 for the establishment of SAs. Another variant is described in Annex B - an "Enterprise Network Security aware IPsec peer" – whereby the IPsec endpoint provides additional visibility and control over the use of the ENS profile for the IPsec operator.

1 Scope

The present document specifies a protocol implementation profile to enable secure communication between IPsec-protected network endpoints while enabling network operations. The Enterprise Network Security profile depends on two protocols in the IPsec family of protocols. First, Internet Key Exchange Protocol Version 2 (IKEv2) [1] is used to establish Security Associations (SAs). In this profile, when certificates are used to provide authentication in IKEv2, those certificates include an extension to provide notice that this profile is being used. Second, the IP Encapsulating Security Payload (ESP) [i.2] is used to encrypt packets.

This profile describes two deployment scenarios. In the first one, one of the IPsec peers is inside the enterprise and the other one is outside the enterprise. In the other scenario, both IPsec peers are inside the enterprise. This profile describes the Diffie-Hellman key exchange, and it specifies the certificate extension that provides visibility information to indicate that the ENS profile is being used.

The actions the IPsec peers take upon receiving the visibility information in the certificate extension and structure of the policy included in the visibility information are not normatively defined; however, capabilities for an optional "Enterprise Network Security aware IPsec peer" are defined. The means by which the IPsec endpoints obtain the static Diffie-Hellman public/private key pairs is specified, and some examples are provided.

A variant of the ENS profile is also provided to enable visibility in circumstances where the operator of an IPsec peer has been informed by other means that packets can be inspected.

The present document also includes the security guarantees made by the ENS profile, based on the security guarantees of the IPsec family of protocols. Details are also provided for MSP profile capabilities that are applicable to the ENS profile, taken from the draft specification of ETSI TS 103 523-1 [i.1], which allows this profile to be a standalone document.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

The following referenced documents are necessary for the application of the present document.

[1] IETF RFC 7296: "Internet Key Exchange Protocol Version 2 (IKEv2)".

[2] FIPS 180-4: "Secure Hash Standard".

[3] ITU-T Recommendation X.509 (10/2016) | ISO/IEC 9594-8: "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks".

[4] IETF RFC 5958: "Asymmetric Key Packages".

[5] IETF RFC 7906: "NSA's Cryptographic Message Syntax (CMS) Key Management Attributes".

[6] IETF RFC 5480: "Elliptic Curve Cryptography Subject Public Key Information".

[7] IETF RFC 5915: "Elliptic Curve Private Key Structure".

[8] IETF RFC 3279: "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile".

[9] IETF RFC 2818: "HTTP Over TLS".

[10] IETF RFC 8446: "The Transport Layer Security (TLS) Protocol Version 1.3".

[11] IETF RFC 8551: "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification".

[12] IETF RFC 7231: "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content".

[13] IETF RFC 8410: "Algorithm Identifiers for Ed25519, Ed448, X25519, and X448 for Use in the Internet X.509 Public Key Infrastructure".

[14] IANA: “Internet Key Exchange Version 2 (IKEv2) Parameters”.

NOTE: available at https://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xhtml

[15] IETF RFC 6989: “Additional Diffie-Hellman Tests for the Internet Key Exchange Protocol Version 2 (IKEv2)”.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

The following referenced documents are not necessary for the application of the present document, but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 103 523-1: "CYBER; Middlebox Security Protocol; Part 1: Capability Requirements".

[i.2] IETF RFC 4303: "IP Encapsulating Security Payload (ESP)".

[i.3] IETF RFC 4301: "Security Architecture for the Internet Protocol".

[i.4] IETF RFC 5652: "Cryptographic Message Syntax (CMS) ".

[i.5] IETF RFC 5083: "Cryptographic Message Syntax (CMS) Authenticated-Enveloped-Data Content Type".

[i.6] IETF RFC 3748: "Extensible Authentication Protocol (EAP)".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

1-sided profile: an MSP profile where middlebox traffic visibility is enabled unilaterally by one endpoint

NOTE: The other endpoint is not able to reject or negotiate the traffic visibility other than by ceasing the communication.

2-sided: an MSP profile where middlebox traffic visibility is enabled only when both endpoints agree to it

Child SA: SA established by IKEv2 for use with the Encapsulating Security Payload (ESP) protocol or Authentication Header (AH) protocol

DH elements: static Diffie-Hellman public/private key pair contained in the Asymmetric Key Package

IKE SA: shared secret information that can be used to efficiently establish Child SAs

initiator: IKEv2 endpoint that begins negotiation of a Security Association

responder: peer IKEv2 endpoint to the initiator

Security Association: secret information shared by IPsec peers

SIG elements: private signing key and a certificate with the corresponding public key contained in the Asymmetric Key Package

single-context: an MSP class where access is granted, or not granted, only to the entire data stream, not to portions of the data stream.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ASN	Abstract Syntax Notation
B2B	Business to Business
CMS	Cryptographic Message Syntax
DDOS	Distributed Denial Of Service
DER	Distinguished Encoding Rules
DH	Diffie-Hellman
ENS	Enterprise Network Security
ESP	Encapsulating Security Payload
HSM	Hardware Security Module
HTTP	HyperText Transfer Protocol
IKEv2	Internet Key Exchange Protocol Version 2
IP	Internet Protocol
IPsec	IP Security
MSP	Middlebox Security Protocol
SA	Security Association
SIG	Signature
VPN	Virtual Private Network

4 Enterprise Network Security for the Middlebox Security Protocol (MSP) framework

4.1 MSP requirements mapping

MSP Part 1 [i.1] defines several Capability Requirements that are demanded of this profile, and any other profile that wishes to comply with the MSP framework. The full and complete mapping of MSP Part 1 [i.1] requirements to this profile is left to a future revision when MSP Part 1 [i.1] is final. However, draft capabilities of MSP profiles are mapped to relevant properties of this profile in Annex C.

For any mapping, a profile needs to be categorized as either a 1-sided or a 2-sided profile, and it needs to provide either single or fine-grained context, as defined in the planned MSP Part 1 [i.1]. Such categorization determines the mandatory and optional requirements for the MSP profile.

The ENS profile defined in the present document is a 1-sided MSP profile, with one endpoint using a static Diffie-Hellman key, and so that endpoint unilaterally enables traffic visibility. The ENS profile defined in the present document is also single-context, as access is granted, or not granted, to the entire data stream associated with the Security Associations (SAs) that are established under the static Diffie-Hellman public/private key pair.

4.2 Deployment of Enterprise Network Security (informative)

4.2.1 Introduction

IPsec offers two modes of use: transport mode and tunnel mode as defined in IETF RFC 4301 [i.3]. In transport mode, ESP provides protection primarily for next layer protocols, and a transport mode SA is typically employed between a pair of hosts to provide end-to-end security services. In tunnel mode, ESP is applied to tunnelled IP packets, and a tunnel mode SA is typically employed between two security gateways or between a host and a security gateway.

EXAMPLE: An encrypting firewall is a type of a security gateway.

4.2.2 Transport Mode Security Associations

Figure 4.1 depicts the deployment of ENS in an enterprise firewall. In this deployment environment, all IPsec SAs use transport mode. As a result, one set of end-to-end SAs are established between the IPsec peer on the private B2B connection and the IPsec peer on the enterprise network. The SAs are established under the static Diffie-Hellman public/private key pair, and this enables the middlebox to decrypt the packets that are encrypted under those SAs.

EXAMPLE: The firewall at the edge of the enterprise is not authorized to inspect the packets in real-time, and it does not have a copy of the static Diffie-Hellman public/private key pair used with IKEv2 to establish the SAs. In this situation, the firewall cannot decrypt the packets, but it can apply rules based on the plaintext IP header.

Figure 4.2 depicts the same configuration as Figure 4.1, except the firewall is IPsec-enabled. In this situation, the firewall at the edge of the enterprise is authorized to inspect the packets in real-time, and the firewall receives a copy of the static Diffie-Hellman public/private key pair used with IKEv2 to establish the SAs. This enables the firewall to decrypt the packets in real-time, which enables a rich set of rules to be applied to all packets.

The middlebox receives a passive copy of these packets along with a copy of the static Diffie-Hellman public/private key pair used with IKEv2 to establish the SAs. This enables the middlebox to decrypt the packets in real-time to perform its functions, such as real-time intrusion detection and malware detection. The middlebox can also store the packets for later decryption and back-in-time analysis.

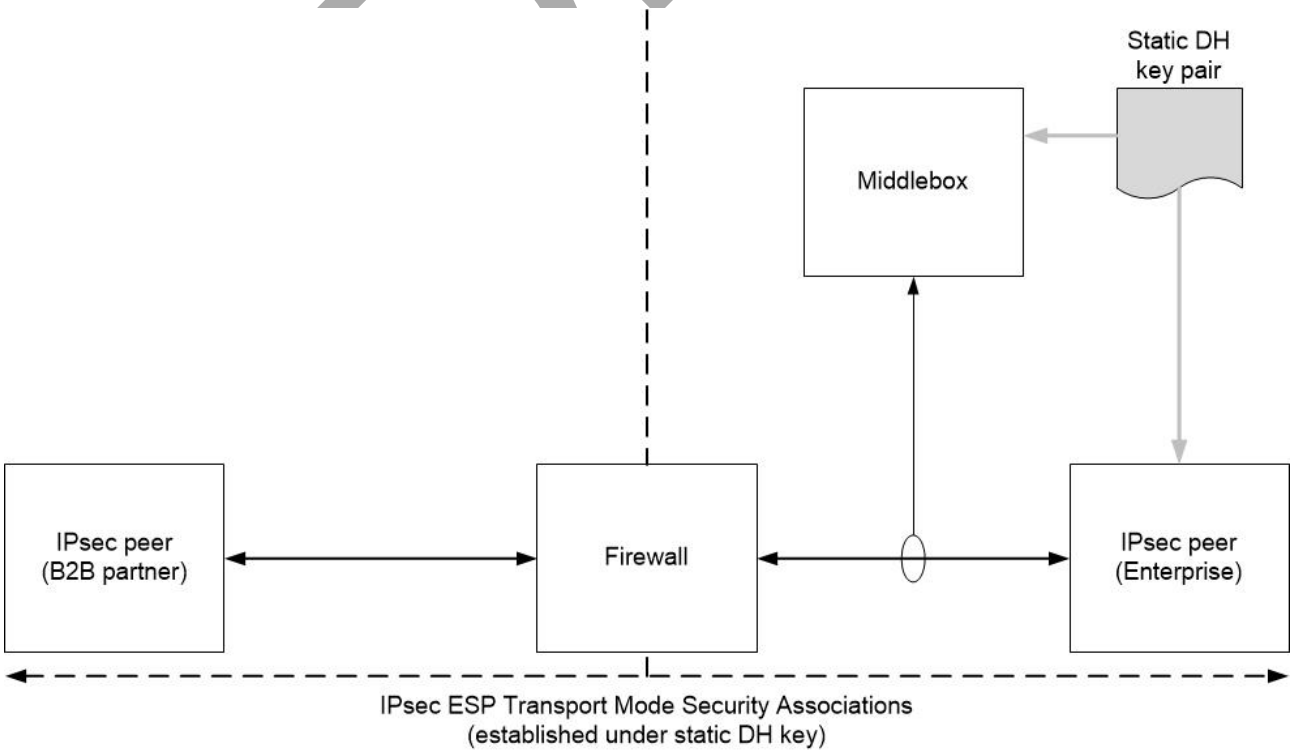

Figure 4.1: Enterprise Firewall and IPsec Transport Mode

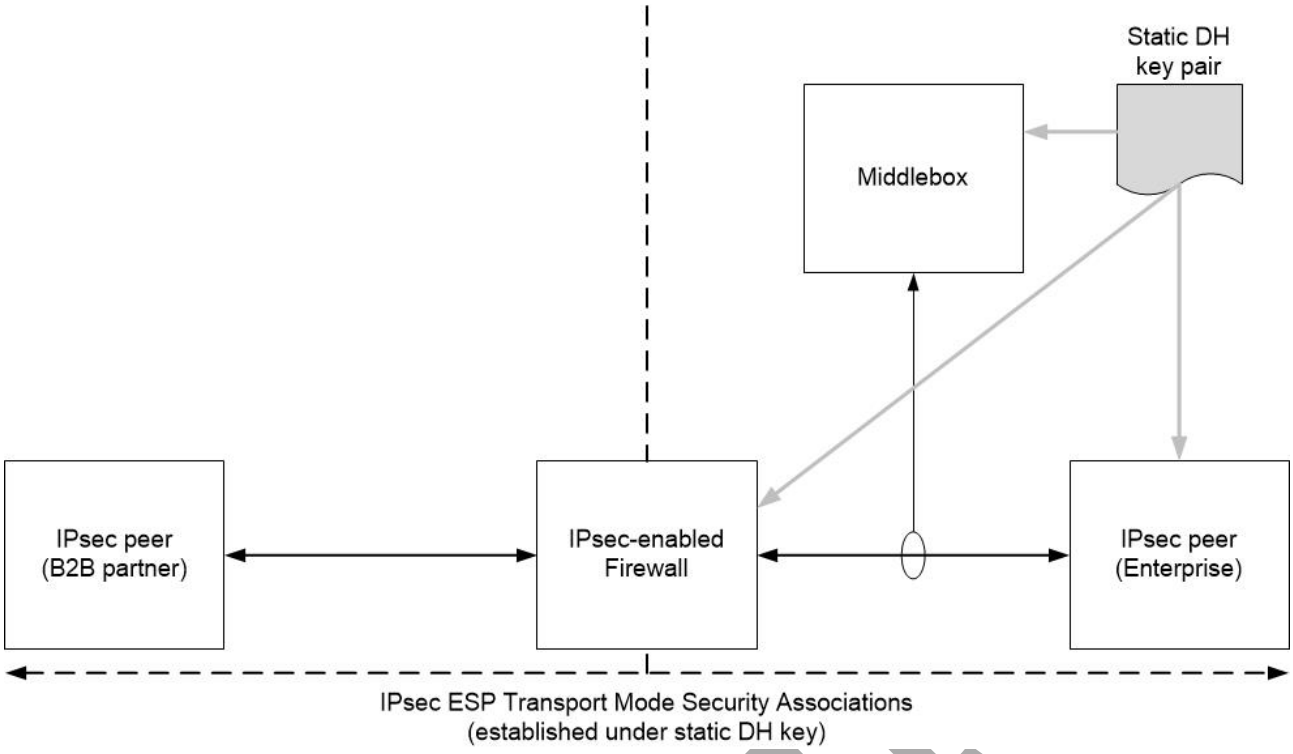

Figure 4.2: Enterprise IPsec-enabled Firewall and IPsec Transport Mode

4.2.3 Tunnel Mode Security Associations

Figure 4.3 depicts the deployment of ENS with an enterprise security gateway. In this deployment environment, all IPsec SAs use tunnel mode. As a result, one set of SAs are established between the IPsec peer on the private B2B connection and the enterprise security gateway, and a separate set of SAs are established between the IPsec peer on the enterprise network and the enterprise security gateway. The SAs inside the enterprise are established under the static Diffie-Hellman public/private key pair, and this enables the middlebox to decrypt the packets that are encrypted under those SAs. The configuration in Figure 4.3 is essentially two back-to-back IPsec Virtual Private Networks (VPNs), where the VPN inside the enterprise enables visibility, but the one on the private B2B connection does not.

The firewall functions in the enterprise security gateway have full access to the plaintext packets, which enables a rich set of rules to be applied to all packets.

As in the case of Transport Mode, the middlebox receives a passive copy of these packets along with a copy of the static Diffie-Hellman public/private key pair used with IKEv2 to establish the SAs. This enables the middlebox to decrypt the packets in real-time or to store the packets for decryption and analysis at a later time.

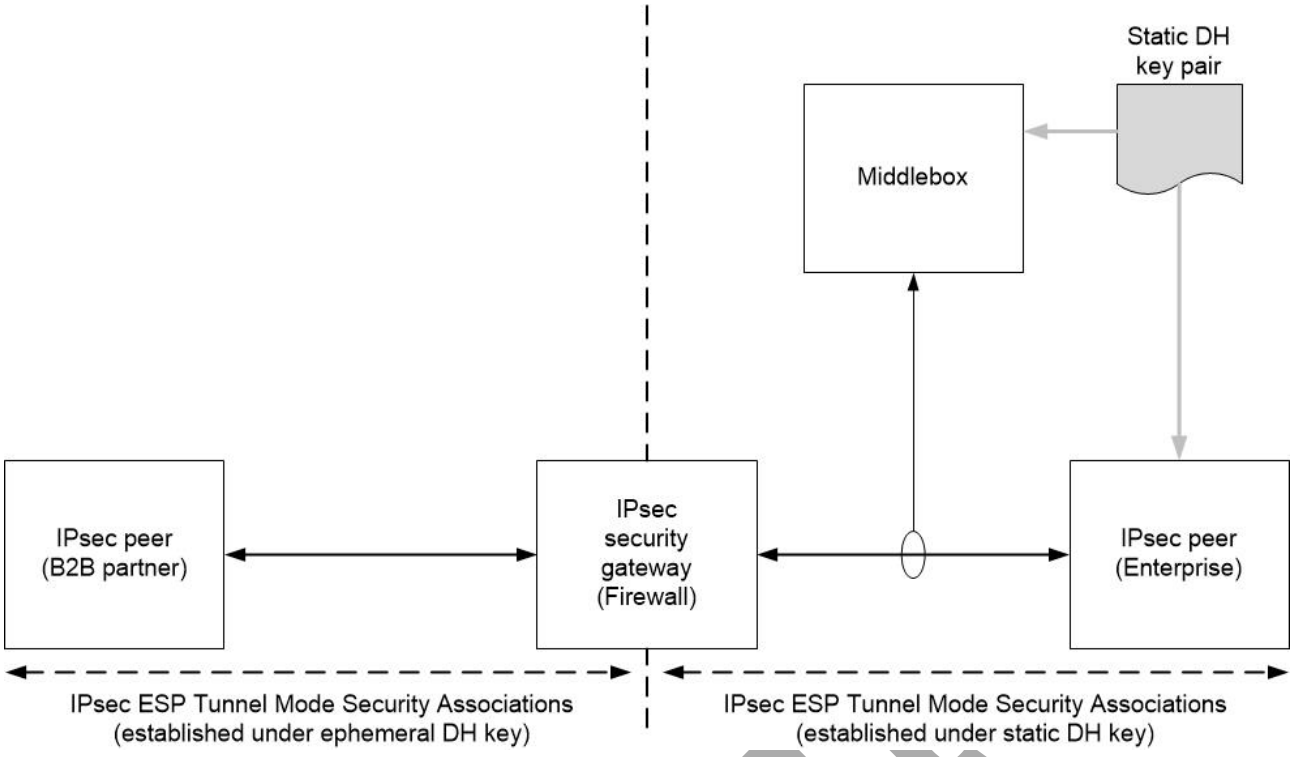

Figure 4.3: Enterprise Security Gateway and IPsec Tunnel Mode

Figure 4.4 depicts the deployment of Enterprise Network Security wholly within an enterprise. IPsec tunnels can be used internally for security and/or compliance reasons. The SAs inside the enterprise are established under the static Diffie-Hellman public/private key pair, and this enables the middlebox to decrypt the packets that are encrypted under those SAs.

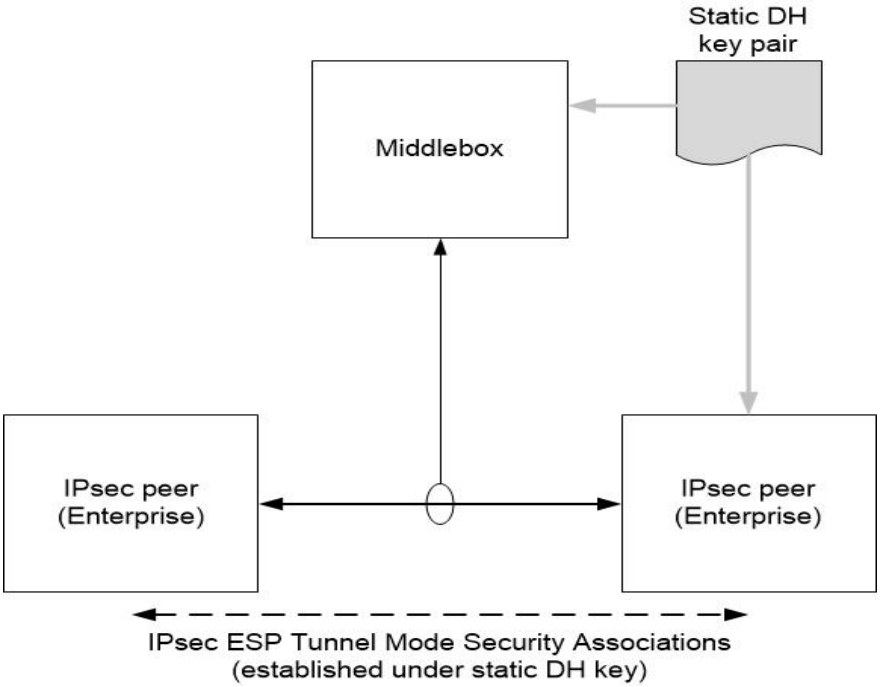

Figure 4.4: IPsec Tunnel Mode within the Enterprise

4.3 Enterprise Network Security

4.3.1 IKEv2 Diffie-Hellman Key Exchange (informative)

IKEv2 [1] is a component of the IPsec family of protocols that performs mutual authentication and establishes Security Associations (SAs). IKEv2 establishes an IKE SA that includes shared secret information that can be used to efficiently establish SAs for the Encapsulating Security Payload (ESP) [i.2]. All IKEv2 communications consist of pairs of messages: a request and a response.

The first exchange of an IKEv2 session, IKE_SA_INIT, negotiates security parameters for the IKE SA, sends nonces, and sends Diffie-Hellman public keys.

The second exchange, IKE_AUTH, transmits identities, proves knowledge of the secrets corresponding to the two identities, and sets up an SA for the first ESP SA, which is referred to as a Child SA. The messages in this exchange can include a certificate or certificate chain. The certificate provides evidence that the key used to compute a digital signature belongs to the name in the ID payload. When certificates are not used, either pre-shared keys or the Extensible Authentication Protocol (EAP) [i.6] are used for authentication.

Subsequent CREATE_CHILD_SA exchanges can be used to establish additional Child SAs for use by ESP. In addition, new keys for the IKE SA and existing ESP SAs can be established with the CREATE_CHILD_SA exchange.

For reference, a brief description of the normal IKEv2 exchange is provided below. Complete details can be found in the IKEv2 specification [1]. In many scenarios, only steps 1 through 7 are used. However, complete visibility requires insight into the creation of multiple ESP SAs, rekeying the IKE SA, and rekeying existing ESP SAs. For this reason, steps 8 through 16 are also provided.

1) The initiator sends the IKE_SA_INIT request, which includes the initiator’s Diffie-Hellman public key (KEi) and the initiator’s nonce (Ni).

2) The responder sends the IKE_SA_INIT response, which completes the Diffie-Hellman exchange by including the responder's Diffie-Hellman public key (KEr) and the responder's nonce (Nr). When certificates are used for authentication, a list of trust anchors is provided (CERTREQ).

3) The initiator and responder each use their own Diffie-Hellman private key, the Diffie-Hellman public key received from the peer, and the exchanged nonces to generate a shared secret, called SKEYSEED. The details of the computation are described in Section 2.14 of the IKEv2 specification [1].

4) The initiator and responder then use the SKEYSEED, along with other information from the IKE_SA_INIT exchanges to generate the cryptographic keys for the IKE SA, which includes an encryption key (SK_e) for each direction, and an authentication key (SK_a) for each direction. In addition, another secret quantity is computed (SK_d) for the derivation of further keying material for Child SAs as well as for the rekeying of the IKE SA.

5) The initiator sends the IKE_AUTH request, which asserts an identity (IDi) and authenticates that identity. When certificates are used for authentication, the initiator's certificate is sent (CERT), a list of trust anchors is provided (CERTREQ), and the identity is authenticated by a digital signature and certificate.

6) The responder sends the IKE_AUTH response, which asserts an identity (IDr) and completes negotiation of a Child SA for use with ESP. When certificates are used for authentication, the responder's certificate is sent (CERT), and the identity is authenticated by a digital signature and certificate.

7) The initiator and responder then use SK_d and the exchanged nonces to compute the keying material, called KEYMAT, for the SA used for ESP.

8) If additional Child SAs are needed, the initiator sends the CREATE_CHILD_SA request, which includes a nonce (Ni) and optionally a Diffie-Hellman public key (KEi).

9) The responder replies with the CREATE_CHILD_SA response, which includes a nonce (Nr) and a Diffie-Hellman public key (KEr) if KEi was included in the request.

10) The initiator and responder then use SK_d, the exchanged nonces, and optionally their own Diffie-Hellman private key along with the Diffie-Hellman public key received from the peer to compute the keying material, called KEYMAT, for the SA used for ESP.

11) If there is a desire to rekey the IKE SA, the initiator sends the CREATE_CHILD_SA request, which includes a nonce (Ni) and a Diffie-Hellman public key (KEi).

12) The responder replies with the CREATE_CHILD_SA response, which includes a nonce (Nr) and a Diffie-Hellman public key (KEr).

13) The initiator and responder then use SK_d, the exchanged nonces, and their own Diffie-Hellman private key along with the Diffie-Hellman public key received from the peer to compute a new SKEYSEED and the keying material for the IKE SA.

14) If there is a desire to rekey an existing Child SA, the initiator sends the CREATE_CHILD_SA request, which includes a nonce (Ni) and optionally a Diffie-Hellman public key (KEi).

15) The responder replies with the CREATE_CHILD_SA response, which includes a nonce (Nr) and a Diffie-Hellman public key (KEr) if KEi was included in the request.

16) The initiator and responder then use SK_d, the exchanged nonces, and can use their own Diffie-Hellman private key along with the Diffie-Hellman public key received from the peer to compute the KEYMAT for the SA used for ESP.

4.3.2 Diffie-Hellman key exchange with visibility

The ENS profile shall use exactly the same messages and procedures as defined in IETF RFC 7296 [1] to establish the initial IKE SA, to rekey the IKE SA, to establish ESP SAs, and to rekey ESP SAs. However, the party located in the enterprise network or data centre (whether initiator or responder), shall do all of the following:

1) use static Diffie-Hellman public/private key pairs instead of generating ephemeral keys;

2) when certificates are used for authentication, use a certificate for authentication that includes an extension that contains visibility information as defined in clause 4.3.3 to indicate that this profile is being used;

   (When pre-shared keys or the Extensible Authentication Protocol (EAP) [i.6] are used for authentication, notice that packets may be inspected is provided in another way as specified in Annex A).

NOTE 1: Neither the static Diffie-Hellman public key nor the certificate extension containing visibility information affect the operation of IKEv2, so an IPsec peer that follows the Enterprise Network Security profile is fully interoperable with other IPsec peers.

IPsec peers that follow the Enterprise Network Security profile shall be provisioned with static Diffie-Hellman public/private key pairs for each supported Diffie-Hellman group.

NOTE 2: Supported Diffie-Hellman groups can be elliptic curves or finite fields, defined in the IANA Internet Key Exchange Version 2 (IKEv2) parameters registry[14].

The static Diffie-Hellman public/private key pairs may be shared with middleboxes that are authorized to decrypt packets from the IPsec peers.

The Diffie-Hellman public/private keys shall be either:

1) installed directly on the IPsec peer, or generated on an associated hardware security module (HSM), and rotated according to organizational policy, described in clause 4.3.4.2; or

2) downloaded from a central key manager and updated according to organizational policy, described in clause 4.3.4.3.

4.3.3 Visibility information

The ENS profile shall support authentication based on a certificate, as defined in ITU-T Recommendation X.509 [3]. The certificate shall include an extension that contains visibility information to indicate that the ENS profile is being used.

1) The visibility information in the certificate may be bound to the static Diffie-Hellman public/private key pairs used in the IKEv2 protocol exchanges by including the fingerprint of the static Diffie-Hellman public key.

2) The static Diffie-Hellman public key used for key exchange shall not be the same as the certificate subject public key in the certificate, which is used for the digital signature for IKEv2 authentication.

3) The visibility information in the certificate shall identify, either generally or specifically, the controlling or authorizing entities or roles or domains, or any combination of these, of any middleboxes that can access the static Diffie-Hellman public/private key pairs described in clause 4.3.2 of the present document.

The above points describe the Visibility Information structure, which shall be defined by the following ASN.1 type:

```
VisibilityInformation ::= SEQUENCE {
   fingerprint         OCTET STRING (SIZE(10)) OPTIONAL,
   accessBy            UTF8String }
```

where:

- When the `fingerprint` is included, it shall be set to the truncated SHA-256 digest of the static Diffie-Hellman public key used in the IKEv2 exchanges, specifically the input to the SHA-256 digest shall be the contents of the Key Exchange Data field of the KE payload defined in section 3.4 of IETF RFC 7296 [1]. The SHA-256 digest shall be represented as the vector of 32-bit words (H_0,H_1,…,H_7) as defined in FIPS 180-4 [2], and then truncated to $H_0||H_1||(H_2>>16)$, which is the high-order 80 bits of the vector in big-endian format.
- The `accessBy` field shall be a human-readable text string that identifies, either generally or specifically, the controlling or authorizing entities or roles or domains, or any combination of these, of any middleboxes that can be granted access to the static Diffie-Hellman private key. The `accessBy` field shall be accurate. A structure for the description is not defined in the present document.

The `VisibilityInformation` structure shall be sent as an `AttributesSyntax` field entry of the `associatedInformation` certificate extension as defined in clause 9.3.2.4 of ITU-T Recommendation X.509 [3]. When the `associatedInformation` extension is being used for ENS `VisibilityInformation`, the extension shall be flagged as non-critical by the issuing CA. The attribute type shall be set to the Object Identifier 0.4.0.3523.5.1 {itu-t(0) identified-organization(4) etsi(0) msp(3523) ens(5) visibility(1)} and the attribute value shall be set to the DER encoding of `VisibilityInformation`. This is described by the following ASN.1:

```
associatedInformation EXTENSION ::= {
  SYNTAX         AttributesSyntax
  IDENTIFIED BY  id-ce-associatedInformation }


visibilityInformation ATTRIBUTE {
  WITH SYNTAX     VisibilityInformation
  SINGLE VALUE    FALSE
  ID              id-msp-ENS-visibility }

id-msp-ENS-visibility OBJECT IDENTIFIER ::= { 0 4 0 3523 5 1 }
```

Thus `fingerprint`, in conjunction with the certificate's validity period, binds the certificate to a particular static Diffie-Hellman public/private key pair. The server has several options for including the `visibilityInformation` attribute in the `associatedInformation` extension.

1) The server may include the `visibilityInformation` attribute with no `fingerprint`. This would serve as an indicator that ENS is being used, but certificate issuance is decoupled from the Diffie-Hellman public/private key pairs used with ENS.

2) The server may bind a single static Diffie-Hellman public/private key pair to a certificate by including a single `VisibilityInformation` value with a `fingerprint` in the `visibilityInformation` attribute. This binds the certificate to the indicated static Diffie-Hellman public/private key pair and would require issuing a new certificate when the static Diffie-Hellman public/private key pair is rotated.

3) The server may bind multiple static Diffie-Hellman public/private key pairs to a single certificate by including multiple `VisibilityInformation` values, one for each key pair, in the `visibilityInformation` attribute. This binds the certificate to the indicated set of static Diffie-Hellman public/private key pairs, and would allow for rotation of this set of static Diffie-Hellman public/private key pairs without issuing a new certificate.

There shall be zero or one `visibilityInformation` attribute per `associatedInformation` extension.

An IPsec peer receiving the visibility information should follow the actions that can be found in Annex B, which is optional.

An optional variant of ENS, where visibility information is not sent, is defined in Annex A. This variant may be used when visibility information is not suitable. This variant shall not be used unless the operator of the IPsec peer has been informed by some means that the packets can be inspected. The means by which the operator is informed is out of scope of the present document.

EXAMPLE: A client could be informed of the visibility information through an acceptable use policy for a client on a private enterprise network.

If operation according to Annex A is supported, it shall be explicitly enabled via a configuration option; otherwise, the IPsec peer shall not establish Enterprise Network Security using certificates without visibility information.

4.3.4 Static Diffie-Hellman public/private key pairs

4.3.4.1 General

The present document does not mandate how the static Diffie-Hellman public/private key pairs defined in clause 4.3.2 are shared between IPsec peers and a middlebox. However, an ENS implementation may use the mechanisms specified in clauses 4.3.4.2-4.3.4.3.4.

4.3.4.2 Directly installed keys

This clause describes one of the simplest means. When the directly installed keys mechanism is used, the static Diffie-Hellman public/private key pair shall be directly installed in both the IPsec peer and permitted middleboxes.

4.3.4.3 Centrally managed keys

4.3.4.3.1 Introduction

Figure 4.5 shows the architecture for using a central key manager to deploy the Enterprise Network Security profile static Diffie-Hellman keys to a key consumer, which include the IPsec peer and passive decryption middleboxes.

When the centrally managed keys mechanism is used, the key manager shall generate the static Diffie-Hellman public/private key pairs. The key manager may actively push keys to the key consumer or the key consumer may pull keys. The Asymmetric Key Package is specified in clause 4.3.4.3.2, the protection of the key package is specified in clause 4.3.4.3.3, and the transfer mechanism is specified in clause 4.3.4.3.4.

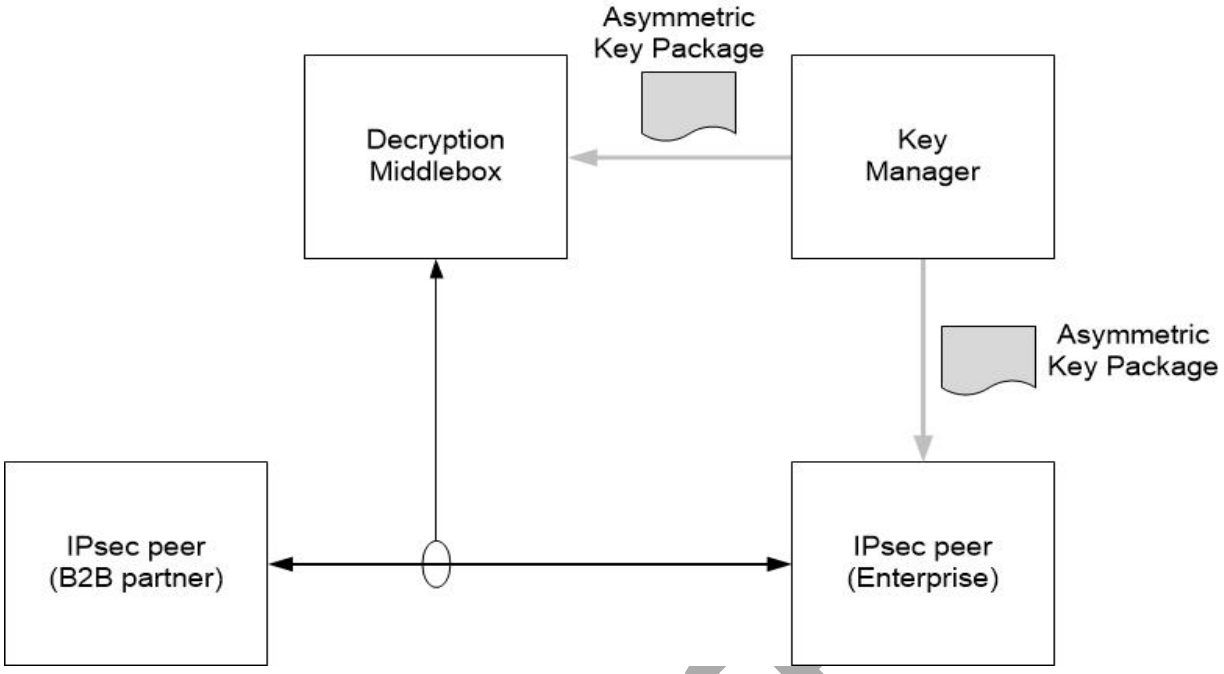

Figure 4.5: Centralized management of static Diffie-Hellman keys

NOTE: In an enterprise environment where a key manager generates the static Diffie-Hellman public/private key pairs for installation on the IPsec peer and decryption middleboxes, the key manager could generate the IKEv2 certificate at the same time.

4.3.4.3.2 Asymmetric key package

When Enterprise Network Security profile static Diffie-Hellman public/private key pairs are sent from the key manager to a key consumer, they shall be packaged using the Asymmetric Key Package defined in IETF RFC 5958 [4]. Each Asymmetric Key Package shall contain one or more `OneAsymmetricKey` elements. Such an element shall be either:

a) static Diffie-Hellman public/private key pairs, hereafter referred to as DH elements; or

b) a private signing key and a certificate with the corresponding public key, hereafter referred to as SIG elements.

Although certificates are not sent in the same `OneAsymmetricKey` element as static Diffie-Hellman keys, each Asymmetric Key Package may contain one or more SIG elements in the Asymmetric Key Package that are bound to the static Diffie-Hellman public keys in the Asymmetric Key Package. The use of multiple certificates is intended for the situation where it is necessary to provide certificates with different signature algorithms. Within the Asymmetric Key Package, the DH elements shall appear before the SIG elements.

For DH elements, the `OneAsymmetricKey` structure that is used to store the static Diffie-Hellman public/private key pairs in the Asymmetric Key Package shall be as defined in IETF RFC 5958 [4]. The `OneAsymmetricKey` fields are set as follows:

1) The `version` field shall be set to version 2 (integer value of 1).

2) The `privateKeyAlgorithm` field shall be set to the appropriate Diffie-Hellman algorithm identifier for the Diffie-Hellman group (defined as `PrivateKeyAlgorithmIdentifier` in IETF RFC 5958 [4]), whether the group uses an elliptic curve or a finite field.

3) The `privateKey` field shall be set to the Diffie-Hellman private key, encoded as an `OCTET STRING`.

4) The `publicKey` field shall be present, and the field shall be set to the Diffie-Hellman public key, encoded as a `BIT STRING`.

5) The `attributes` field shall include a validity period for the Diffie-Hellman keys using the `KeyValidityPeriod` attribute defined in Section 15 of IETF RFC 7906 [5].

For elliptic curve groups, the `OneAsymmetricKey` fields shall follow the conventions specified in IETF RFC 5480 [6], IETF RFC 5915 [7], and IETF RFC 8410 [13].

EXAMPLE 1:

```
object identifier:  { 1 3 132 1 12 }
parameter encoding: ECParameters
private key encoding:  ECPrivateKey
public key encoding:   ECPoint
```

For finite field groups, the `OneAsymmetricKey` fields shall follow the conventions specified in IETF RFC 3279 [8].

EXAMPLE 2:

```
object identifier:  { 1 2 840 10046 2 1 }
parameter encoding: DomainParameters
private key encoding:  INTEGER
public key encoding:   INTEGER
```

When the package contains one or more certificates and corresponding signature private keys, the certificates shall include `VisibilityInformation` as defined in clause 4.3.3.

For SIG elements, the element that is used to store the public and private keys in the Asymmetric Key Package shall be as defined in IETF RFC 5958 [4], and the `OneAsymmetricKey` fields are set as follows:

1) The `version` field shall be set to version 1 (integer value of 0).
2) The `privateKeyAlgorithm` field shall be set to the same `AlgorithmIdentifier` that appears in the `algorithm` field of the `SubjectPublicKeyInfo` element of the certificate as defined in ITU-T X.509 [3].
3) The `privateKey` field shall be set to the signature private key, encoded as an `OCTET STRING`.
4) The `publicKey` field shall be absent.
5) The `attributes` field shall include the certificate, which includes the signature public key, as a `userCertificate` attribute as defined in Section 8 of IETF RFC 7906 [5].

4.3.4.4.3 Protecting the key package

When the centrally managed keys mechanism is used, the Cryptographic Message Syntax (CMS) defined in IETF RFC 5652 [i.4] and IETF RFC 5083 [i.5] may be used to provide authentication, integrity and/or confidentiality to the Asymmetric Key Package transported between the key manager and the key consumer.

4.3.4.4.4 Transferring keys

4.3.4.4.4.1 Protocol overview

When the centrally managed keys mechanism is used, Enterprise Network Security Asymmetric Key Packages shall be transferred between the key manager and key consumer using HTTP Over TLS [9], commonly called HTTPS.

The HTTP messages containing key packages shall comprise an HTTP header followed by the Asymmetric Key Package, encoded using Distinguished Encoding Rules (DER). The HTTP Content-Type header shall be set to `application/pkcs8` as defined in IETF RFC 5958 [4] for plaintext packages and set to `application/pkcs7-mime` as defined in IETF RFC 8551 [11] if the package is encapsulated using CMS.

4.3.4.4.4.2 Transfer initiated by the key manager

Key consumers may support key transfers initiated by the key manager via an HTTPS key installation service, whereby the key manager initiates an HTTPS connection to the key consumer, which acts as an HTTP server.

When the key consumer supports an HTTPS key installation service, the key consumer shall support receiving a key package via an HTTP PUT request to a request-target, given here in origin-form, of `/.well-known/enterprise-`

`network-security/keys`. When key transfers initiated by the key manager via an HTTPS key installation service are not supported by the key consumer, how the key consumer receives key packages is out of scope of the present document.

It is the responsibility of the key consumer to determine that the key manager is authorized to provide keys, and it is the responsibility of the key manager to determine that the key consumer is authorized to receive these keys.

4.3.4.3.4.3 Transfer initiated by the key consumer

Key managers may support key transfers initiated by the key consumer via an HTTPS key retrieval service, whereby the key consumer initiates an HTTPS connection to the key manager, which acts as an HTTP server.

When key transfers initiated by the key consumer via an HTTPS key retrieval service are not supported by the key manager, how the key consumer retrieves the key package is out of scope of the present document.

When key transfers initiated by the key consumer via an HTTPS key retrieval service are supported by the key manager, the key manager shall support retrieval of a key package using both of the following HTTP GET request formats:

1) `GET /.well-known/enterprise-network-security/keys?fingerprints=[fingerprints]`

where:

a) `fingerprints` shall be present, and its value, `[fingerprints]`, shall be either empty or a comma-separated list of the hexadecimal strings, where each of the strings in the list is the fingerprint of the static Diffie-Hellman public key, as defined in clause 4.3.3, of the keying material being requested.

b) The key manager shall return a key package that contains the corresponding static Diffie-Hellman public/private key pair for each fingerprint for which it has a record. In the unlikely case that the key manager has more than one static Diffie-Hellman public/private key pair corresponding to a requested fingerprint, it shall return all matching static Diffie-Hellman public/private key pairs in the key package. If `[fingerprints]` is empty, the actions of the implementation are out of scope of the present document.

c) The key manager shall return an appropriate HTTP error code if there is not at least one matching static Diffie-Hellman public/private key pair as defined in Section 6 of IETF RFC 7231 [12].

2) `GET /.well-known/enterprise-network-security/keys?groups=[groups]&certs=[sigalgs]&context=[contextstr]`

where:

a) `groups` shall be included, and the value, `[groups]`, shall be a comma-separated list where each entry in the list is a Diffie-Hellman Group Transform ID defined in the IANA Internet Key Exchange Version 2 (IKEv2) Parameters registry [14], represented as a hexadecimal string, for which the associated static Diffie-Hellman public/private key pairs are being requested.

b) `certs` may be included. If `certs` is included, the value, `[sigalgs]`, shall be a comma-separated list where each entry is a colon-separated pair of compatible `SignatureScheme` values defined in Section B.3.1.3 of IETF RFC 8446 [10], represented as a hexadecimal string, where the compatible values are those with names beginning with 'rsa_pkcs1_', 'ecdsa_', or 'dsa_sha1_'. The first value in the pair shall indicate the requested signature algorithm to be used by the certificate issuer to sign the certificate. The second value in the pair shall indicate the subject public key algorithm in the certificate. If `certs` is included, then for each entry in the list, the key consumer shall request one additional server certificate using that scheme, which is bound to all returned key pairs.

c) `context` may be included. If `context` is included, the value, `contextstr`, is a free string that the key manager shall use to determine the static Diffie-Hellman public/private key pairs and certificate contents to return. The structure of `contextstr` is not specified.

d) The key manager shall return a key package containing a static Diffie-Hellman public/private key pair for each group listed in `[groups]`that the key manager supports. For each static Diffie-Hellman

public/private key pair in the key package, the key manager shall also return a corresponding certificate for each given signature algorithm pair listed in `[sigalgs]` that it supports.

e) If no group in `[groups]` is supported by the key manager, the key manager shall return an appropriate HTTP error code as defined in Section 6 of IETF RFC 7231 [12]. If the key manager is unable to use `contextstr`, the key manager may return an appropriate HTTP error code, which shall be as defined in clause 6 of IETF RFC 7231 [12]. Other ways to handle the error are outside the scope of the present document.

f) If `groups` is included but `certs` and `context` are not included, then the key manager will not provide any certificates.

EXAMPLE 1: For a key consumer requesting two keys and supporting responses in either format, the HTTP GET request would be:

```
GET /.well-known/enterprise-network-security/
keys?fingerprints=0001020304050607080 9,0908070605040302 0100
Accept: application/pkcs8,application/pkcs7-mime
```

EXAMPLE 2: A key consumer requests two keys, one that uses the 384-bit random ECP group and one that uses Curve `25519`. The key consumer also requests two corresponding certificates bound to both keys. One of the certificates is to be signed with `rsa_pkcs1_sha256` and to be associated with an `ecdsa_secp256r1_sha256` subject signing key. The other certificate is to be signed with `ecdsa_secp384r1_sha384` and to be associated with an `ecdsa_secp384r1_sha384` subject signing key. The key consumer requests the resulting key package in only the `pkcs8` format. The HTTP GET request would be:

```
GET /.well-known/enterprise-network-security/
keys?groups=0x0014,0x001f&certs=0x0401:0x403,0x0503:0x0503
Accept: application/pkcs8
```

It is the responsibility of the key consumer to determine that the key manager is authorized to provide keys, and it is the responsibility of the key manager to determine that the key consumer is authorized to receive these keys.

5 Security

The Enterprise Network Security profile does not provide all the security guarantees provided by IKEv2 [1] and ESP [i.2].

The ENS profile does not provide the same forward secrecy for IPsec Security Associations (SAs). Knowledge of a given static Diffie-Hellman private key can be used to decrypt all of the SAs with keying material that was established with that static Diffie-Hellman private key; however, forward secrecy for all of the protected packets begins when all copies of that static Diffie-Hellman private key have been destroyed.

It is possible that both sides of the IPsec tunnel could decide to use static Diffie-Hellman private keys. In this scenario, recovery of the static Diffie-Hellman private key or shared secret (g^ir as per IETF RFC 7296 [1]) from either IPsec peer would allow decryption of all traffic encrypted under the SAs that were established with the recovered private key or shared secret. Because of this possibility, the administrator(s) of the IPsec peers should rotate static Diffie-Hellman private keys frequently. If both IPsec peers are using their own static Diffie-Hellman private key, forward secrecy for all of the packets protected under a given SA begins when all copies of both static Diffie-Hellman private keys used to derive that SA have been destroyed.

The relaxation of forward secrecy is deliberate. With proper management of the static Diffie-Hellman private key, the ENS profile maintains a high-degree of authentication, integrity and confidentiality while also allowing security breaches to be rapidly detected. The ENS profile also maximizes service availability by enabling rapid detection of services and application problems. It is the responsibility of each organization deploying the ENS profile to evaluate the trade-off between the IPsec security guarantees and the operational visibility provided by this profile.

As well as meeting IPsec security guarantees, with the exception of the forward secrecy described above, Annex C provides a map of the mandatory capabilities for the Middlebox Security Protocol (MSP) against the ENS profile.

The variant of this profile described in Annex A meets the same IPsec security guarantees as the full ENS profile, with the same exception regarding forward security. In the Annex A variant, MSP profile capabilities relating to notice about the visibility of middleboxes are not met; however, this is compensated for by the operator of the IPsec peer having been informed by other means that the packets can be inspected.

When an organization uses IPsec to protect the traffic between an external IPsec peer and the enterprise network or data centre, and then uses the Enterprise Network Security profile to protect traffic inside the enterprise network or data centre (as in Figure 4.3), use of this profile shall be explicitly enabled via a configuration option.

When this profile is enabled, the applicable additional key material tests required by IETF RFC 6989 [15] for IKEv2 implementations that reuse Diffie-Hellman keys shall be implemented.

Annex A (normative): Middlebox visibility information variant

This annex is optional. It establishes a variant of the Enterprise Network Security profile that was defined in clause 4, which is the same in all respects except that the IPsec peer sends no visibility information in the certificate; thus clause 4.3.3 is not carried out.

This variant of the Enterprise Network Security profile shall not be used unless the operator of the IPsec peer has been informed by some means that the packets can be inspected. The means by which the operator is informed is out of scope of this document.

EXAMPLE 1: The IPsec peers are wholly within a private enterprise network and the operator of the IPsec peer has already been notified by a contract denoted condition of access to the network that packets can be inspected.

EXAMPLE 2: The operator of the IPsec peer agrees to an acceptable use policy to access a private enterprise network. This acceptable use policy states that packets can be inspected.

Annex B (normative): Requirements for an Enterprise Network Security aware IPsec peer

This annex is optional. An IPsec peer that satisfies the requirements specified in this annex can be designated as "Enterprise Network Security aware".

a) The IPsec peer may provide configuration means to accept all Enterprise Network Security connections or deny all Enterprise Network Security connections as indicated by the certificate extension defined in clause 4.3.3.

b) If the IPsec implementation provides a means of viewing an IPsec peer's certificate, it should correctly parse and display the contents of the certificate extension defined in clause 4.3.3.

c) The IPsec peer may provide a means to establish only Enterprise Network Security profile Security Associations (SAs) that match a whitelist.

d) The IPsec peer may provide a means to refuse to establish any Enterprise Network Security profile SAs that match a blacklist.

e) The IPsec peer may offer a user prompt to establish an SA in accordance with the Enterprise Network Security profile.

Annex C (informative): Mapping MSP desired capabilities to the Enterprise Network Security profile

References to client and server throughout this annex correspond to initiator and responder in the context of IKEv2.

The reader is advised that some of the capabilities described here are not necessarily guaranteed by the protocol; some are guaranteed to the extent that entities with access to the Enterprise Network Security profile static Diffie-Hellman keys are trusted. This trust in these Enterprise Network Security entities - for the IPsec peer to provide accurate visibility information, and for all entities to share the Enterprise Network Security Diffie-Hellman private key only according to that policy - assures some of the capabilities described in this annex. The remaining capabilities are assured by the Enterprise Network Security profile.

The Enterprise Network Security profile is defined as a 1-sided, single-context MSP profile. These definitions are in the planned MSP Part 1 [i.1] as well as in this profile, as described in clause 4.1.

The Enterprise Network Security profile meets the mandatory capabilities for a 1-sided, single-context MSP profile. MSP capabilities, defined for the MSP standards, are split into three groupings:

- **Audit:** capabilities that relate to the ability for a middlebox to be audited using MSP.
- **Access:** capabilities that relate to the access granted to a middlebox that is using MSP.
- **Visibility:** capabilities that relate to the ability for a middlebox to be discovered using MSP.

First, mapping to audit capabilities, the Enterprise Network Security profile meets the following:

a) Destination endpoint able to detect if an unauthorized change to the data has occurred.

b) Middleboxes with read-only access able to detect if an unauthorized change to the data has occurred.

c) Middleboxes with permission to modify content able to detect if an unauthorized change to the data has occurred.

d) Middleboxes modifying content able to validate that no unauthorized changes have occurred prior to receipt by middleboxes or third parties.

These audit capabilities are satisfied by the Enterprise Network Security profile, as they are inherited from the security properties of IPsec as discussed in clause 5.

Second, the following access capability is met:

e) Client or server, alone, able to grant middlebox access permissions.

This is satisfied, as the Enterprise Network Security peer alone can grant access to a device by sharing its Diffie-Hellman private key with that device.

Third, these visibility capabilities are met:

f) Client able to learn the owner of all middleboxes.

g) Client able to learn the identity and function of all third-party middleboxes or groups of third-party middleboxes (i.e. middleboxes not under ownership of client or server endpoints).

h) Client able to learn the identity and function of all middleboxes or groups of middleboxes.

i) Server able to learn the owner of all middleboxes.

j) Server able to learn the identity and function of all third-party middleboxes or groups of third-party middleboxes.

k) Server able to learn the identity and function of all middleboxes or groups of middleboxes.

l) The client and server able to receive validation of identity of all middleboxes or groups of middleboxes.

m) Client able to learn the long-term identity of the server.

n) The client able to receive validation of the server long-term identity.

o) Server able to learn an identity (which may include "anonymous" or similar) for the client.

p) Server able to reject anonymous clients (if anonymous clients are supported).

The visibility capabilities f), g), h), i), j), k) and l) are satisfied by the inclusion of visibility information in the Enterprise Network Security profile certificate.

The visibility requirements m), n), o) and p) are satisfied by inheritance of the security guarantees from IPsec.

The `accessBy` field, part of the Visibility Information field defined in clause 4.3.3, identifies, either generally or specifically, the controlling or authorizing entities or roles or domains, or any combination of these, of any middleboxes that can be given access to the static Diffie-Hellman private key used to create the Security Association. For example, an `accessBy` can restrict access of the static Diffie-Hellman private key to the administrative domain that controls all middleboxes within an organization.

The certificate issuer and the Enterprise Network Security peer are trusted to enforce the visibility restriction on the middlebox entities that are given access to the static Diffie-Hellman private key. Though any dishonest holder of key material in an ENS Security Association (SA) can give it to any party, this is also the case in IPsec - and the protection of the key material is assured to the extent that the Enterprise Network Security endpoint is trusted by the other SA endpoint.

The Annex A variant of the Enterprise Network Security profile meets the same MSP profile capabilities except for f), g), h) and l) which are the capabilities relating to visibility of middleboxes. This is because the IPsec peer does not receive the visibility information. This is compensated for by the operator of the IPsec peer having been informed by other means that the packets can be inspected.

History

Document history		
V0.0.0	May 2019	Skeleton
V0.0.1	June 2019	Early Draft
V0.0.2	June 2019	Early Draft
V0.0.3	August 2019	Updates to Section 5, Security
V0.0.4	November 2019	Base version based on V0.0.3, numerous updates
V0.0.5	November 2020	Updates to section 4.3.3, Visibility Information
V0.0.6	January 2020	Updated Section 4.3.3, Visibility Information, to use the X.509 associatedInformation certificate extension; also miscellaneous updates
V0.0.7	January 2020	Updates to section 4.3.3