Internet Engineering Task Force (IETF)
Request for Comments: 5840
Category: Standards Track
ISSN: 2070-1721

K. Grewal
Intel Corporation
G. Montenegro
Microsoft Corporation
M. Bhatia
Alcatel-Lucent

April 2010

Wrapped Encapsulating Security Payload (ESP) for Traffic Visibility

Abstract

This document describes the Wrapped Encapsulating Security Payload (WESP) protocol, which builds on the Encapsulating Security Payload (ESP) RFC 4303 and is designed to allow intermediate devices to (1) ascertain if data confidentiality is being employed within ESP, and if not, (2) inspect the IPsec packets for network monitoring and access control functions. Currently, in the IPsec ESP standard, there is no deterministic way to differentiate between encrypted and unencrypted payloads by simply examining a packet. This poses certain challenges to the intermediate devices that need to deep inspect the packet before making a decision on what should be done with that packet (Inspect and/or Allow/Drop). The mechanism described in this document can be used to easily disambiguate integrity-only ESP from ESP-encrypted packets, without compromising on the security provided by ESP.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc5840.

Table of Contents

1. Introduction

Use of ESP within IPsec [RFC4303] specifies how ESP packet encapsulation is performed. It also specifies that ESP can provide data confidentiality and data integrity services. Data integrity without data confidentiality ("integrity-only ESP") is possible via the ESP-NULL encryption algorithm [RFC2410] or via combined-mode algorithms such as AES-GMAC [RFC4543]. The exact encapsulation and algorithms employed are negotiated out of band using, for example, Internet Key Exchange Protocol version 2 (IKEv2) [RFC4306] and based on policy.

Enterprise environments typically employ numerous security policies (and tools for enforcing them), as related to access control, content screening, firewalls, network monitoring functions, deep packet inspection, Intrusion Detection and Prevention Systems (IDS and IPS), scanning and detection of viruses and worms, etc. In order to enforce these policies, network tools and intermediate devices require visibility into packets, ranging from simple packet header inspection to deeper payload examination. Network security protocols that encrypt the data in transit prevent these network tools from performing the aforementioned functions.

When employing IPsec within an enterprise environment, it is desirable to employ ESP instead of Authentication Header (AH) [RFC4302], as AH does not work in NAT environments. Furthermore, in order to preserve the above network monitoring functions, it is desirable to use integrity-only ESP. In a mixed-mode environment, some packets containing sensitive data employ a given encryption cipher suite, while other packets employ integrity-only ESP. For an intermediate device to unambiguously distinguish which packets are using integrity-only ESP requires knowledge of all the policies being employed for each protected session. This is clearly not practical. Heuristics-based methods can be employed to parse the packets, but these can be very expensive, requiring numerous rules based on each different protocol and payload. Even then, the parsing may not be robust in cases where fields within a given encrypted packet happen to resemble the fields for a given protocol or heuristic rule. In cases where the packets may be encrypted, it is also wasteful to check against heuristics-based rules, when a simple exception policy (e.g., allow, drop, or redirect) can be employed to handle the encrypted packets. Because of the non-deterministic nature of heuristics-based rules for disambiguating between encrypted and non-encrypted data, an alternative method for enabling intermediate devices to function in encrypted data environments needs to be defined. Additionally, there are many types and classes of network devices employed within a given network and a deterministic approach provides a simple solution for all of them. Enterprise environments

typically use both stateful and stateless packet inspection mechanisms. The previous considerations weigh particularly heavy on stateless mechanisms such as router Access Control Lists (ACLs) and NetFlow exporters. Nevertheless, a deterministic approach provides a simple solution for the myriad types of devices employed within a network, regardless of their stateful or stateless nature.

This document defines a mechanism to provide additional information in relevant IPsec packets so intermediate devices can efficiently differentiate between encrypted and integrity-only packets. Additionally, and in the interest of consistency, this extended format can also be used to carry encrypted packets without loss in disambiguation.

This document is consistent with the operation of ESP in NAT environments [RFC3947].

The design principles for this protocol are the following:

- Allow easy identification and parsing of integrity-only IPsec traffic
- Leverage the existing hardware IPsec parsing engines as much as possible to minimize additional hardware design costs
- Minimize the packet overhead in the common case

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Applicability Statement

The document is applicable only to the wrapped ESP header defined below, and does not describe any changes to either ESP [RFC4303] or the IP Authentication Header (AH) [RFC4302].

There are two well-accepted ways to enable intermediate security devices to distinguish between encrypted and unencrypted ESP traffic:

- The heuristics approach [Heuristics] has the intermediate node inspect the unchanged ESP traffic, to determine with extremely high probability whether or not the traffic stream is encrypted.

- The Wrapped ESP (WESP) approach, described in this document, in contrast, requires the ESP endpoints to be modified to support the new protocol. WESP allows the intermediate node to distinguish encrypted and unencrypted traffic deterministically, using a simpler implementation for the intermediate node.

Both approaches are being documented simultaneously by the IP Security Maintenance and Extensions (IPsecME) Working Group, with WESP (this document) as a Standards Track RFC while the heuristics approach is expected to be published as an Informational RFC. While endpoints are being modified to adopt WESP, we expect both approaches to coexist for years because the heuristic approach is needed to inspect traffic where at least one of the endpoints has not been modified. In other words, intermediate nodes are expected to support both approaches in order to achieve good security and performance during the transition period.

2. Wrapped ESP (WESP) Header Format

Wrapped ESP (WESP) encapsulation uses protocol number 141. Accordingly, the (outer) protocol header (IPv4, IPv6, or Extension) that immediately precedes the WESP header SHALL contain the value (141) in its Protocol (IPv4) or Next Header (IPv6, Extension) field. WESP provides additional attributes in each packet to assist in differentiating between encrypted and non-encrypted data, and to aid in parsing of the packet. WESP follows RFC 4303 for all IPv6 and IPv4 considerations (e.g., alignment considerations).

This extension essentially acts as a wrapper to the existing ESP protocol and provides an additional 4 octets at the front of the existing ESP packet for IPv4. For IPv6, additional padding may be required and this is described below.

The packet format may be depicted as follows:

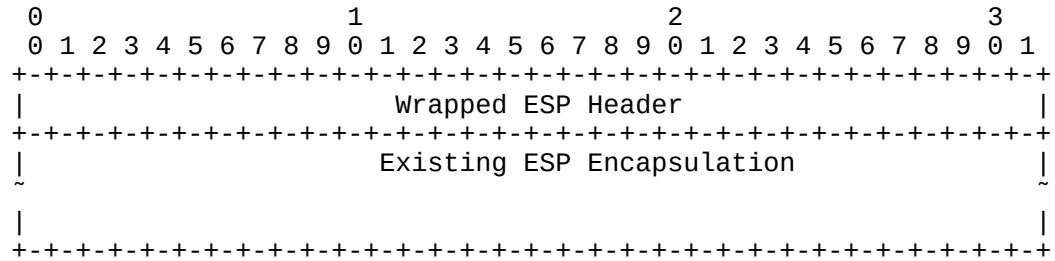


Figure 1: WESP Packet Format

By preserving the body of the existing ESP packet format, a compliant implementation can simply add in the new header, without needing to change the body of the packet. The value of the new protocol used to identify this new header is 141. Further details are shown below:

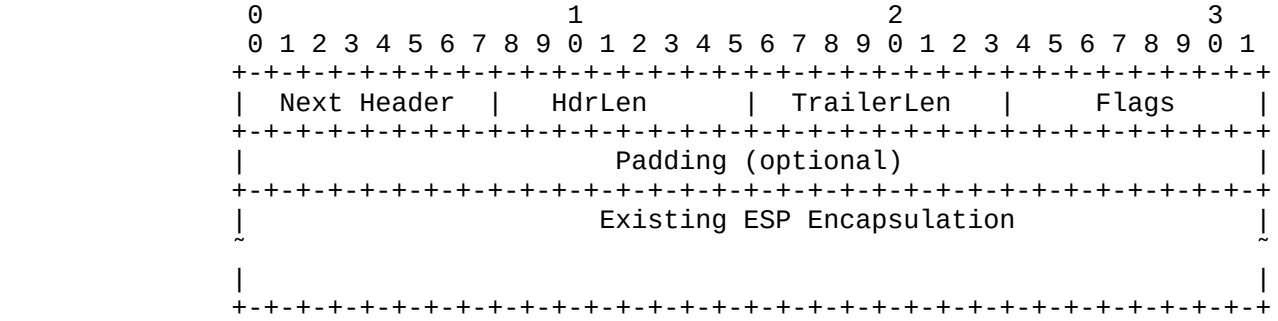


Figure 2: Detailed WESP Packet Format

Where:

Next Header, 8 bits: This field MUST be the same as the Next Header field in the ESP trailer when using ESP in the Integrity-only mode. When using ESP with encryption, the "Next Header" field looses this name and semantics and becomes an empty field that MUST be initialized to all zeros. The receiver MUST do some sanity checks before the WESP packet is accepted. The receiver MUST ensure that the Next Header field in the WESP header and the Next Header field in the ESP trailer match when using ESP in the Integrity-only mode. The packet MUST be dropped if the two do not match. Similarly, the receiver MUST ensure that the Next Header field in the WESP header is an empty field initialized to zero if using WESP with encryption. The WESP flags dictate if the packet is encrypted.

HdrLen, 8 bits: Offset from the beginning of the WESP header to the beginning of the Rest of Payload Data (i.e., past the IV, if present and any other WESP options defined in the future) within the encapsulated ESP header, in octets. HdrLen MUST be set to zero when using ESP with encryption. When using integrity-only ESP, the following HdrLen values are invalid: any value less than 12; any value that is not a multiple of 4; any value that is not a multiple of 8 when using IPv6. The receiver MUST ensure that this field matches with the header offset computed from using the negotiated Security Association (SA) and MUST drop the packet in case it does not match.

TrailerLen, 8 bits: TrailerLen contains the size of the Integrity Check Value (ICV) being used by the negotiated algorithms within the IPsec SA, in octets. TrailerLen MUST be set to zero when using ESP with encryption. The receiver MUST only accept the packet if this field matches with the value computed from using the negotiated SA. This ensures that sender is not deliberately setting this value to obfuscate a part of the payload from examination by a trusted intermediary device.

Flags, 8 bits: The bits are defined most-significant-bit (MSB) first, so bit 0 is the most significant bit of the flags octet.

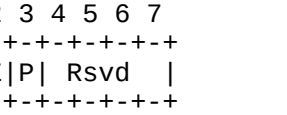


Figure 3: Flags Format

Version (V), 2 bits: MUST be sent as 0 and checked by the receiver. If the version is different than an expected version number (e.g., negotiated via the control channel), then the packet MUST be dropped by the receiver. Future modifications to the WESP header require a new version number. In particular, the version of WESP defined in this document does not allow for any extensions. However, old implementations will still be able to find the encapsulated cleartext packet using the HdrLen field from the WESP header, when the 'E' bit is not set. Intermediate nodes dealing with unknown versions are not necessarily able to parse the packet correctly. Intermediate treatment of such packets is policy dependent (e.g., it may dictate dropping such packets).

Encrypted Payload (E), 1 bit: Setting the Encrypted Payload bit to 1 indicates that the WESP (and therefore ESP) payload is protected with encryption. If this bit is set to 0, then the payload is using integrity-only ESP. Setting or clearing this bit also impacts the value in the WESP Next Header field, as described above. The recipient MUST ensure consistency of this flag with the negotiated policy and MUST drop the incoming packet otherwise.

Padding header (P), 1 bit: If set (value 1), the 4-octet padding is present. If not set (value 0), the 4-octet padding is absent. This padding MUST be used with IPv6 in order to preserve IPv6 8-octet alignment. If WESP is being used with UDP encapsulation (see Section 2.1 below) and IPv6, the Protocol Identifier (0x00000002) occupies 4 octets so the IPv6 padding is not needed, as the header is already on an 8-octet boundary. This padding MUST NOT be used with IPv4, as it is not needed to guarantee 4-octet IPv4 alignment.

Rsvd, 4 bits: Reserved for future use. The reserved bits MUST be sent as 0, and ignored by the receiver. Future documents defining any of these bits MUST NOT affect the distinction between encrypted and unencrypted packets or the semantics of HdrLen. In other words, even if new bits are defined, old implementations will be able to find the encapsulated packet correctly. Intermediate nodes dealing with unknown reserved bits are not necessarily able to parse the packet correctly. Intermediate treatment of such packets is policy dependent (e.g., it may dictate dropping such packets).

Future versions of this protocol may change the version number and/or the reserved bits sent, possibly by negotiating them over the control channel.

As can be seen, the WESP format extends the standard ESP header by the first 4 octets for IPv4 and optionally (see above) by 8 octets for IPv6.

2.1. UDP Encapsulation

This section describes a mechanism for running the new packet format over the existing UDP encapsulation of ESP as defined in RFC 3948. This allows leveraging the existing IKE negotiation of the UDP port for Network Address Translation Traversal (NAT-T) discovery and usage [RFC3947] [RFC4306], as well as preserving the existing UDP ports for ESP (port 4500). With UDP encapsulation, the packet format can be depicted as follows.

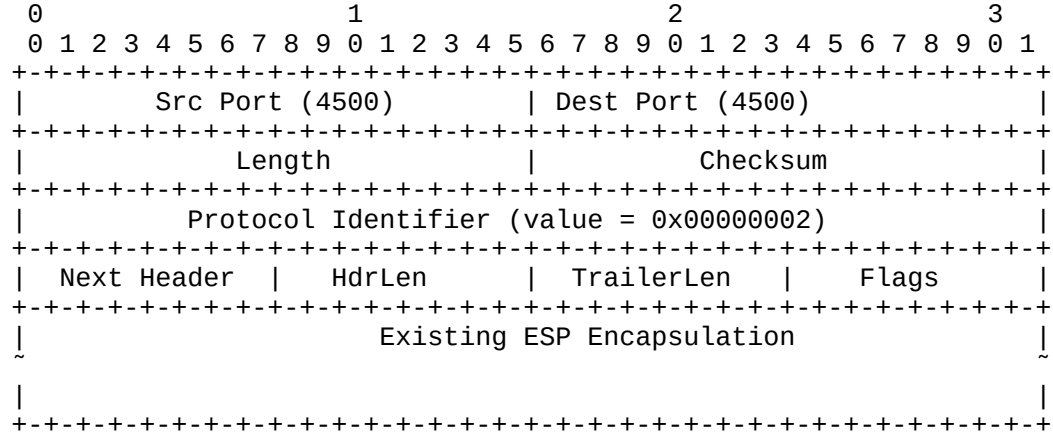


Figure 4: UDP-Encapsulated WESP Header

Where:

Source/Destination port (4500) and checksum: describes the UDP encapsulation header, per RFC 3948.

Protocol Identifier: new field to demultiplex between UDP encapsulation of IKE, UDP encapsulation of ESP per RFC 3948, and the UDP encapsulation in this specification.

According to RFC 3948, Section 2.2, a 4-octet value of zero (0) immediately following the UDP header indicates a Non-ESP marker, which can be used to assume that the data following that value is an IKE packet. Similarly, a value greater then 255 indicates that the packet is an ESP packet and the 4-octet value can be treated as the ESP Security Parameter Index (SPI). However, RFC 4303, Section 2.1 indicates that the values 1-255 are reserved and cannot be used as the SPI. We leverage that knowledge and use one of these reserved values to indicate that the UDP encapsulated ESP header contains this new packet format for ESP encapsulation.

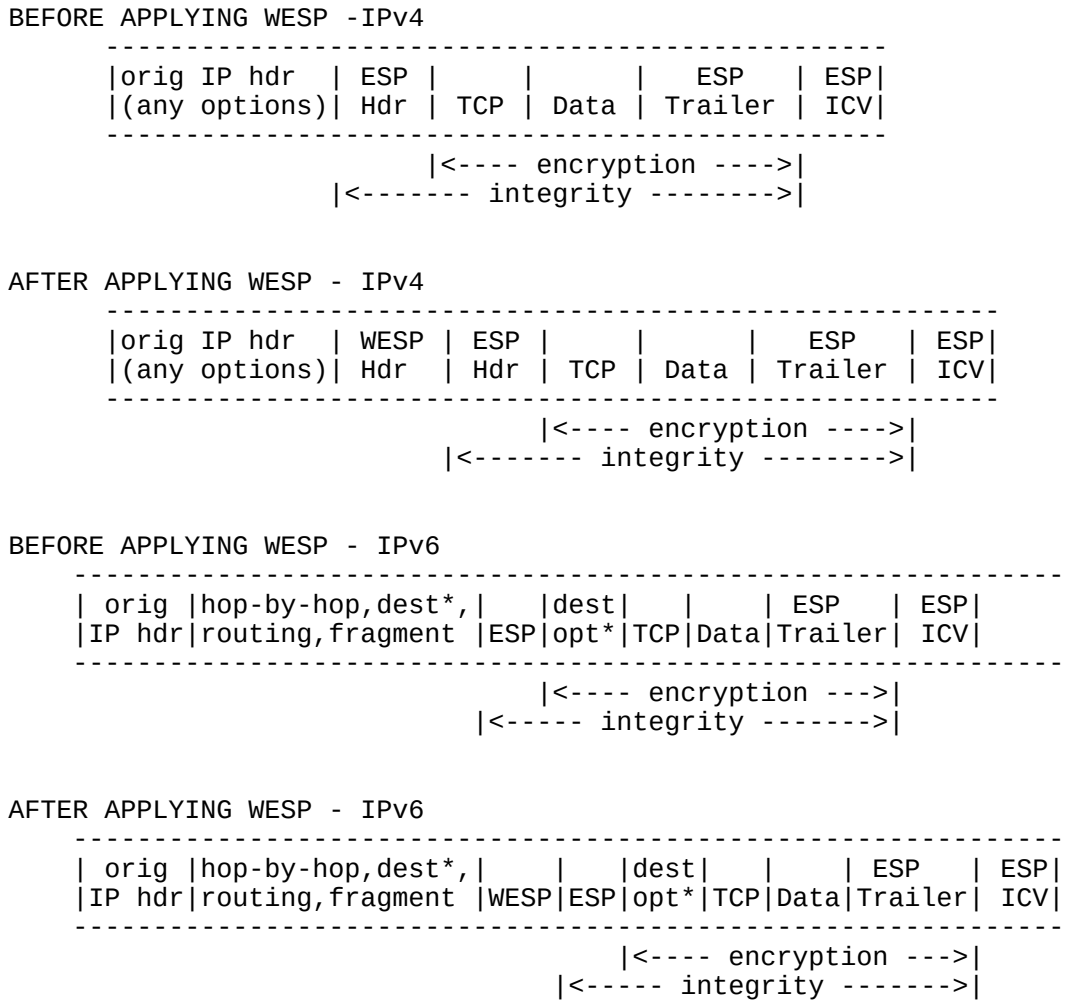
The remaining fields in the packet have the same meaning as per Section 2 above.

2.2. Transport and Tunnel Mode Considerations

This extension is equally applicable to transport and tunnel mode where the ESP Next Header field is used to differentiate between these modes, as per the existing IPsec specifications.

2.2.1. Transport Mode Processing

In transport mode, ESP is inserted after the IP header and before a next layer protocol, e.g., TCP, UDP, ICMP, etc. The following diagrams illustrate how WESP is applied to the ESP transport mode for a typical packet, on a "before and after" basis.

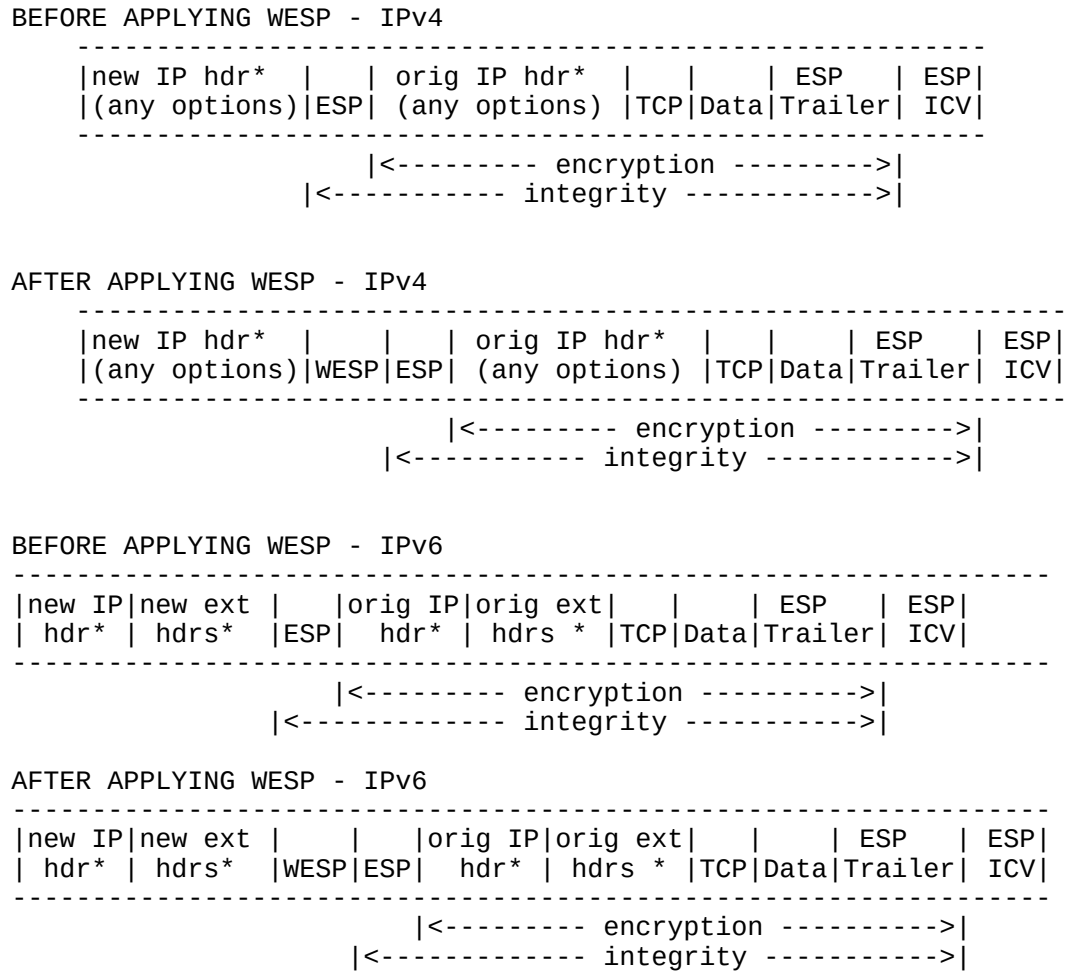


* = if present, could be before WESP, after ESP, or both

All other considerations are as per RFC 4303.

2.2.2. Tunnel Mode Processing

In tunnel mode, ESP is inserted after the new IP header and before the original IP header, as per RFC 4303. The following diagram illustrates how WESP is applied to the ESP tunnel mode for a typical packet, on a "before-and-after" basis.



* = if present, construction of outer IP hdr/extensions and modification of inner IP hdr/extensions is discussed in the Security Architecture document.

All other considerations are as per RFC 4303.

2.3. IKE Considerations

This document assumes that WESP negotiation is performed using IKEv2. In order to negotiate the new format of ESP encapsulation via IKEv2 [RFC4306], both parties need to agree to use the new packet format. This can be achieved using a notification method similar to USE_TRANSPORT_MODE, defined in RFC 4306.

The notification, USE_WESP_MODE (value 16415) MUST be included in a request message that also includes an SA payload requesting a CHILD_SA using ESP. It signals that the sender supports the WESP version defined in the current document and requests that the CHILD_SA use WESP mode rather than ESP for the SA created. If the request is accepted, the response MUST also include a notification of type USE_WESP_MODE. If the responder declines the request, the CHILD_SA will be established using ESP, as per RFC 4303. If this is unacceptable to the initiator, the initiator MUST delete the SA.

Note: Except when using this option to negotiate WESP mode, all CHILD_SAs will use standard ESP.

Negotiation of WESP in this manner preserves all other negotiation parameters, including NAT-T [RFC3948]. NAT-T is wholly compatible with this wrapped format and can be used as-is, without any modifications, in environments where NAT is present and needs to be taken into account.

WESP version negotiation is not introduced as part of this specification. If the WESP version is updated in a future specification, then that document MUST specify how the WESP version is negotiated.

3. Security Considerations

As this document augments the existing ESP encapsulation format, UDP encapsulation definitions specified in RFC 3948 and IKE negotiation of the new encapsulation, the security observations made in those documents also apply here. In addition, as this document allows intermediate device visibility into IPsec ESP encapsulated frames for the purposes of network monitoring functions, care should be taken not to send sensitive data over connections using definitions from this document, based on network domain/administrative policy. A strong key agreement protocol, such as IKEv2, together with a strong policy engine should be used in determining appropriate security policy for the given traffic streams and data over which it is being employed.

ESP is end-to-end and it will be impossible for the intermediate devices to verify that all the fields in the WESP header are correct. It is thus possible to modify the WESP header so that the packet sneaks past a firewall if the fields in the WESP header are set to something that the firewall will allow. The endpoint thus must verify the sanity of the WESP header before accepting the packet. In an extreme case, someone colluding with the attacker, could change

the WESP fields back to the original values so that the attack goes unnoticed. However, this is not a new problem and it already exists IPsec.

4. IANA Considerations

The WESP protocol number assigned by IANA out of the IP Protocol Number space is 141.

The USE_WESP_MODE notification number assigned out of the "IKEv2 Notify Message Types - Status Types" registry's 16384-40959 (Expert Review) range is 16415.

The SPI value of 2 has been assigned by IANA out of the reserved SPI range from the SPI values registry to indicate use of the WESP protocol within a UDP-encapsulated, NAT-T environment.

IANA has created a new registry for "WESP Flags" to be managed as follows:

The first 2 bits are the WESP Version Number. The value 0 is assigned to the version defined in this specification. Further assignments of the WESP Version Number are to be managed via the IANA Policy of "Standards Action" [RFC5226]. For WESP version numbers, the unassigned values are 1, 2, and 3. The Encrypted Payload bit is used to indicate if the payload is encrypted or using integrity-only ESP. The Padding Present bit is used to signal the presence of padding. The remaining 4 bits of the WESP Flags are undefined and future assignment is to be managed via the IANA Policy of "IETF Review" [RFC5226].

5. Acknowledgments

The authors would like to acknowledge the following people for their feedback on updating the definitions in this document:

David McGrew, Brian Weis, Philippe Joubert, Brian Swander, Yaron Sheffer, Pasi Eronen, Men Long, David Durham, Prashant Dewan, Marc Millier, Russ Housley, and Jari Arkko, among others.

Manav Bhatia would also like to acknowledge Swati and Maitri for their continued support.

6. References

6.1. Normative References

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

[RFC2410] Glenn, R. and S. Kent, "The NULL Encryption Algorithm and Its Use With IPsec", RFC 2410, November 1998.

[RFC3948] Huttunen, A., Swander, B., Volpe, V., DiBurro, L., and M. Stenberg, "UDP Encapsulation of IPsec ESP Packets", RFC 3948, January 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC4543] McGrew, D. and J. Viega, "The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH", RFC 4543, May 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

6.2. Informative References

[RFC3947] Kivinen, T., Swander, B., Huttunen, A., and V. Volpe, "Negotiation of NAT-Traversal in the IKE", RFC 3947, January 2005.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4306] Kaufman, C., Ed., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

[Heuristics] Kivinen, T. and D. McDonald, "Heuristics for Detecting ESP-NULL packets", Work in Progress, March 2010.

Authors' Addresses

Ken Grewal
Intel Corporation
2111 NE 25th Avenue, JF3-232
Hillsboro, OR 97124
USA

EMail: ken.grewal@intel.com

Gabriel Montenegro
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA

EMail: gabriel.montenegro@microsoft.com

Manav Bhatia
Alcatel-Lucent
Manyata Embassy
Nagawara Bangalore
India

EMail: manav.bhatia@alcatel-lucent.com