A Cryptographic Tour of the IPsec Standards

Kenneth G. Paterson,
Information Security Group,
Royal Holloway, University of London,
Egham, Surrey, TW20 0EX, UK
kenny.paterson@rhul.ac.uk

Abstract

In this article, we provide an overview of cryptography and cryptographic key management as they are specified in IPsec, a popular suite of standards for providing communications security and network access control for Internet communications. We focus on the latest generation of the IPsec standards, recently published as Request for Comments 4301–4309 by the Internet Engineering Task Force, and how they have evolved from earlier versions of the standards.
Keywords: IP; IPsec; network security; cryptography; key management.

1 Introduction

IPsec provides security at the IP network layer of the TCP/IP protocol stack. This means that all IP packets can be protected, irrespective of the upper layer protocol being carried in the packet payloads, and that no re-engineering of applications is required in order to take advantage of the security provided by IPsec. The security provided by IPsec can also be made transparent to end users. For these reasons, IPsec forms the basis of many Virtual Private Networking (VPN) solutions, where it is used to provide security for communications over an untrusted network such as the Internet. IPsec has also been used by many vendors to build Remote Access Solutions (RAS), allowing organizations to control the access of roaming users to internal networks and hosts. IPsec has seen widespread deployment in recent years. For example, implementations of IPsec exist in Microsoft Windows 2000 and XP, HP's proprietary operating system HPUX, IBM's AIX operating system, and in the Linux kernel. Several other open source projects are also developing IPsec implementations and IPsec is now widely supported in commercial networking hardware. IPsec is mandatory in IPv6 (though we will focus on IPsec in IPv4 in this article).

IPsec is formally specified in a number of "standards" each of which is known as a Request For Comments (RFC) and is published by the Internet Engineering Task Force (IETF). They are all freely available from the IETF website at www.ietf.org. The majority of these IPsec documents are a result of a collaborative process coordinated by the IETF's IPsec working group. The first two generations of these documents (principally RFCs 1825–1829, published in 1995, and 2401–2412, published in 1998) are really only intended to provide a guide for implementors and are notoriously complex, difficult to interpret and lacking in overall structure. Ferguson and Schneier [39] provide a detailed critique of these and other perceived flaws in the IPsec standards. This lack of clarity has arguably hampered the adoption of

IPsec, in particular because of its impact on the interoperability of different vendors' IPsec implementations. The interested reader is invited to consult [38, 42] for introductions to IPsec that are more accessible than the RFCs themselves. The third and latest incarnation of the core IPsec standards were published as RFCs 4301–4309 in December 2005, and are somewhat more accessible. RFC 4301 [21], the architectural document for IPsec, gives a good overview of the design goals and the overall architecture of IPsec. However, the new RFCs are still a long and complex set of documents, totalling over 300 pages.

The set of security services offered by IPsec includes access control, connectionless integrity, data origin authentication, detection and rejection of replays (a form of partial sequence integrity), confidentiality (via encryption), and limited traffic flow confidentiality. These are all delivered using symmetric key techniques. The IPsec protocols also support automated key management, with key exchange protocols using both symmetric and asymmetric cryptographic techniques. Whilst commonly viewed as providing integrity and confidentiality services for data in transit, Section 3.1 of RFC 4301 makes it clear that the primary purpose of IPsec is to provide a form of access control for network traffic, by defining a boundary at which IP packets can be compared to IPsec policies, and either cross the boundary unchanged, have IPsec security mechanisms applied, or be discarded altogether. The effect of this IPsec processing, determined by policy, is to create a flexible method of implementing traffic separation and allow a limited form of firewalling.

In this paper, we discuss cryptography and cryptographic key management in the latest versions of the IPsec standards and attempt to explain the ways in which these aspects have developed from the earlier generations of standards. In particular, we trace the evolution of integrity protection and encryption mechanisms in IPsec, seeing how new algorithms have been adopted and how authenticated encryption is now an integral part of IPsec. We examine how IPsec handles key management via security policy, security associations and the IKE (and now IKEv2) key exchange protocol. We also comment on some of the ways in which cryptography in IPsec might develop in future.

Readers should bear in mind that the scope of this article is actually rather narrow: cryptography is an important component of IPsec, but it is only one of many aspects of a complex system of interacting components. Nor is our coverage of cryptography in IPsec comprehensive: only the RFCs tell the whole story.

2 Basic IPsec Concepts and Terminology

The IPsec protocols can be deployed in two basic modes: tunnel and transport. In tunnel mode cryptographic protection is provided for entire IP packets. In essence, a whole packet plus security fields is treated as the new payload of an outer IP packet, with its own header, called the outer header. The original, or inner, IP packet is said to be *encapsulated* within the outer IP packet. In tunnel mode, IPsec processing is typically performed at security gateways on behalf of endpoint hosts. The gateways could be perimeter firewalls or routers. The use of gateways means that hosts need not be IPsec-aware, but that security is provided from gateway-to-gateway rather than in an end-to-end fashion. By contrast, in transport mode, the header of the original packet itself is preserved, some security fields are inserted, and the payload together with some header fields undergo cryptographic processing. Transport mode is typically used when end-to-end security services are needed, and provides protection mostly for the packet payload. In either mode, one can think of the IPsec implementation as

intercepting normal IP packets and performing processing on them before passing them on (to the network interface layer in the case of outbound processing, or to the upper layers in the case of inbound processing).

There are two main IPsec protocols which specify the actual cryptographic processing applied to packets. These are called Authentication Header (AH) and Encapsulating Security Payload (ESP). AH provides integrity protection, data origin authentication and anti-replay services for packets through the application of MAC algorithms and the inclusion of sequence numbers in packets. AH is discussed in more detail in Section 3. ESP provides similar services to AH (though the coverage of integrity protection is more limited) and in addition provides confidentiality and traffic flow confidentiality services through symmetric key encryption and variable length padding of packets. ESP is discussed in more detail in Section 4.

IPsec policy acts in concert with these modes and protocols to determine the actual protocols (ESP or AH), modes of processing (tunnel or transport), algorithms, keys, and other cryptographic parameters which are used to cryptographically transform packets. Multiple nested layers of IPsec processing are allowed, and through this, complex transform sets can be built up. The topic of policy for IPsec is a large one and mostly beyond the scope of this article. However, we will need some of the basic concepts later when discussing key management aspects of IPsec. Essentially, each IPsec implementation contains a Security Policy Database (SPD), each entry of which defines processing rules for certain types of traffic. Each entry in the SPD points to one or more Security Associations (SAs) (or the need to establish new SAs). In turn, each SA gives the detailed parameters needed to determine the actual processing to be applied to an IP packet. Thus, IP packets are intercepted and compared to the SPD, each match with an SPD entry identifies a policy and a collection of SAs that implement the policy, and these SAs are "applied" to the packets. For more details of this process, the reader should consult RFC 4301 [21].

Naturally, appropriate and consistent entries in SPDs need to be in place at communicating endpoints before the security available in IPsec can be afforded to packets. The SAs that determine actual processing also need to be put in place at the endpoints. The SAs contain (amongst other information) cryptographic keys, intialization vectors and anti-replay counters for AH and ESP, and so we have stumbled across the IPsec key management problem. This problem can be solved manually, and this approach works well for small-scale deployments or testing purposes. However, for larger scale and more robust use of IPsec, an automated method is needed. The Internet Key Exchange (IKE) Protocol provides the preferred method for SA negotiation and associated cryptographic parameter establishment. The latest version of IKE, named IKEv2, provides a flexible set of methods for authentication and establishment of keys and other parameters, supporting both asymmetric and symmetric cryptographic methods. IKEv2 supersedes the original version of IKE specified in [9, 10, 11]. We examine IKEv2 and its evolution from the original IKE in Section 5.

2.1 IPsec document roadmap

As we have noted above, the IPsec documents are many, long and complex. We provide here a summary of the contents of RFCs 4301-4309 as a handy reference point. Our summary is extended from the one given in [21].

- Security architecture – RFC 4301 [21], explaining the architecture and components of IPsec and their interactions, as well as providing the overall philosophy of IPsec.

- Security protocols – RFCs 4302 [22] and 4303 [23], describing the AH and ESP protocols.
- Automated key management – RFC 4306 [26] defining IKEv2.
- Cryptographic algorithms for integrity and encryption – RFC 4305 [25], defining the mandatory, default algorithms for use with AH and ESP, plus a separate RFC for each cryptographic algorithm, e.g. RFC 4309 [29].
- Cryptographic algorithms and parameters for key exchange – RFC 4307 [27], defining the mandatory algorithms for use with IKEv2, and RFC 4304 [24], describing how extended sequence numbers can be negotiated in IKE.
- Cryptographic suites – RFC 4308 [28] providing recommendations for collections of algorithms (or suites) that can be adopted by system administrators.

The above list illustrates an important cryptographic design principle in IPsec: the separation of protocols (such as ESP, AH and IKEv2) from the algorithms used in those protocols. The specification of mandatory algorithms is needed to ensure interoperability and ease deployment, but the intention is that RFCs 4305 and 4307 will be updated from time-to-time to reflect the state-of-the-art in cryptography. The separation means that any updating can be done without necessitating any changes to the "core" IPsec protocols. The first generation of IPsec RFCs did not feature this separation: the process of separating protocols and algorithms was begun in the second generation of IPsec RFCs and completed in the third.

3 Authentication Header

The new version of AH is specified in RFC 4302 [22] and is little changed from the previous version in RFC 2402 [4]. Essentially, it provides integrity protection and data origin authentication for as much of the IP packet as is possible using a MAC algorithm. It also provides an optional anti-replay service. Certain fields of the IP packet header cannot be input to the MAC calculation because they may change during the packet's transit across a network and so are unpredictable to the receiver. The AH protocol adds its cryptographic protection by inserting a bit sequence called the Authentication Header into IP packets; this header has the particular format shown in Figure 1. We discuss the most important of these fields in the remainder of this section and give brief notes on the other fields. Detailed discussion of all the fields can be found in [22, Section 2].

3.1 AH Integrity Check Value Field and MAC algorithms

The Authentication Header contains an Integrity Check Value (ICV) or MAC value. Since the scope of the MAC calculation includes both the IP packet header and the Authentication Header containing the MAC value, the mutable fields and the MAC value are both set to zero for the purposes of MAC calculation and confirmation.

The length of the MAC depends on the particular MAC algorithm in use. Restrictions are that the MAC value must be an integral number of 32 bits in length and that the overall authentication header must be a multiple of 32 bits in length for IPv4, or a multiple of 64 bits in length for IPv6. A MAC value of 96 bits ensures that these criteria are all satisfied; otherwise some form of additional padding may be needed, which would potentially be wasteful of precious non-payload bytes.

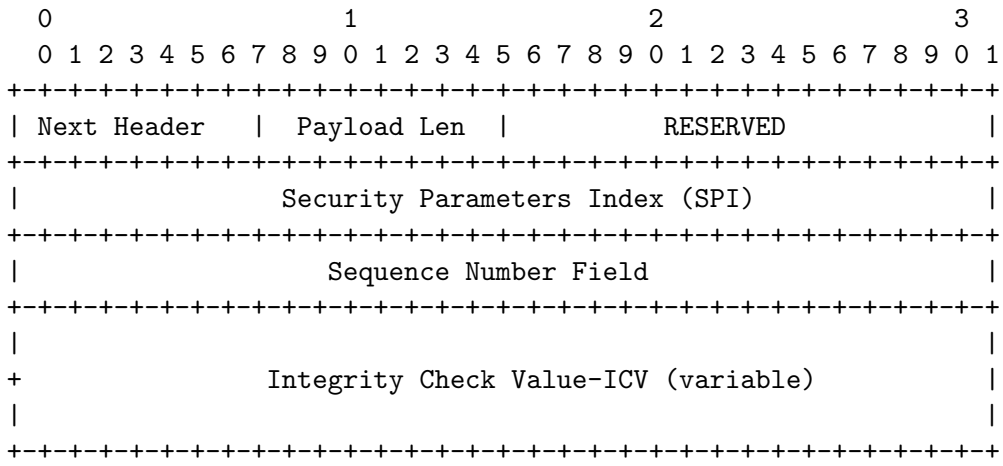

Figure 1: AH format according to RFC 4302 [22].

RFC 4302 does not specify any MAC algorithms itself. As with RFC 2402, this task is deferred to subsequent RFCs. RFC 4305 [25] specifies a number of MAC algorithms which are mandatory to implement for AH, or which may become mandatory in future. These are listed in Table 1. Notice that all three entries in this table refer to MAC algorithms with 96 bit outputs. Two of the MAC algorithms rely on the HMAC construction of RFC 2104 [2] (see also [32]), while the third uses a mode of operation of AES [48].

The second column of this table refers to the extent to which AH implementations are required to implement the various algorithms. The exact meanings of these entries are defined in [25]; the modifier "+" on "SHOULD" means that the requirement is likely to become a "MUST" at some future point. One change to note from RFC 2402 is that HMAC-MD5-96 has moved from a "MUST" implement to a "MAY" implement, reflecting weaknesses that have recently become apparent in the MD5 hash algorithm [53], even though these attacks do not currently appear to weaken the security of HMAC based on MD5. It is notable though that support for HMAC-SHA1-96 is still a "MUST", even after recent cryptanalytic results concerning SHA-1 [54] and when NIST has indicated that SHA-1 be gradually phased out of use in US Federal applications [47]. We expect to see the newer generation of SHA algorithms (SHA-256, SHA-384, and SHA-512) appearing in IPsec-related RFCs in future. Additional algorithms are certainly allowed. For example, a fourth MAC algorithm HMAC-RIPEMD-160-96 is specified for use with AH in RFC 2857 [14] but not mandated for support in [25].

All of the above mandated algorithms have a 96 bit MAC value, which may seem a little short given the expected lifetime of the IPsec standards. However, as have noted above, the choice of MAC output lengths is constrained in various ways, and a 96 bit output probably represents a reasonable compromise between competing demands of security and minimization of packet overheads.

3.2 AH Sequence Number Field

The Sequence Number Field is used to provide an optional anti-replay service. The service operates roughly as follows (both for AH and for ESP when an integrity service is offered by ESP). When an AH SA is first established and the anti-replay service is selected, a counter

Algorithm	Requirement	Key size (bits)	Output size (bits)	Reference
HMAC-SHA1-96	MUST	160	96	RFC 2404 [6]
AES-XCBC-MAC-96	SHOULD+	128	96	RFC 3566 [16]
HMAC-MD5-96	MAY	128	96	RFC 2403 [5]

Table 1: Mandatory MAC algorithms for AH specified in RFC 4305

stored in the SA is set to zero. It is then incremented for each packet that is processed by that SA, and the least significant 32 bits are inserted in the Sequence Number Field. The recipient maintains a sliding window of recently received sequence number values, and accepts only those packets whose sequence number lies within the window and which are not marked as having already been received. Only if the sequence number test passes does the MAC verification take place. The reason to use a sliding window instead of just a single counter at the recipient is that packets frequently arrive out of order in IP networks, and so rejecting all "old" packets would have a severe performance impact, with many packets being dropped. The fact that the Sequence Number Field is protected by a MAC means that it is infeasible for an attacker to construct a packet that will be accepted as both recent (within the window) and valid (having a correct MAC). Note that even though anti-replay is an optional service, the Sequence Number Field must be included in AH. The receiver may simply choose to ignore it.

There are significant differences in the handling of sequence numbers in RFC 2402 and RFC 4302. The counters used in RFC 2402 are only 32 bits long, so the entire counter is transmitted in the Authentication Header. An overflow of the counter is not allowed, and an attempt to transmit a packet that would result in an overflow is an auditable event if anti-replay has been selected. Instead, typical behaviour is for the sender to anticipate the overflow and attempt to establish a new SA ahead of this event. RFC 4302 allows an optional Extended Sequence Number (ESN) to be used. This is helpful in high-speed networks, where a 32-bit counter could easily overflow during normal operations. ESNs are 64 bits long, and the entire 64 bits is used in the MAC calculation by AH even though only the least significant 32 bits of the ESN are carried in the Sequence Number Field. For the purposes of MAC calculation, the most significant 32 bits are placed after the payload, meaning that the ESN is actually split into two parts rather than appearing as a sequence of 64 consecutive bits in the input to the MAC. This is somewhat unusual, but does allow the AH format to remain the same as that specified in RFC 2402 when 32 bit sequence numbers are used. The transmission of only half the ESN in AH leads to the need for a synchronization mechanism in the event that more than 2^{32} consecutive packets are lost. This is addressed in [22, Appendix B3]. RFC 4302 indicates that the default setting is to use ESNs rather than 32 bit sequence numbers; RFC 4304 [24] explains how IKE can be modified to allow negotation of ESNs.

3.3 Other fields in AH

The Next Header field in AH is a one byte (8-bit) field indicating the type of the payload following the Authentication Header. For example, a value of 4 indicates that what follows is an IPv4 packet, while a value of 6 indicates TCP.

The Payload Length field indicates the length of the Authentication Header in 32-bit words, minus 2. It is needed because the header can contain a MAC value of variable length.

The Security Parameters Index (SPI) field is a 32 bit value identifying the SA that was used during outbound AH processing. The SPI is shared between sender and recipient at the time of SA establishment and allows the recipient to quickly obtain the cryptographic parameters necessary to perform inbound processing.

3.4 The future of AH

As we shall see below, the cryptographic services provided by AH can also largely be provided by ESP (with one notable difference being the reduced coverage of integrity protection in ESP). Since ESP can also provide confidentiality services that AH cannot, one might expect to see AH playing a diminishing role in IPsec in future. Indeed this duplication led Ferguson and Schneier to suggest that AH be eliminated altogether as early as 1999 [39]. The NIST guide to IPsec VPNs [42] notes that some IPsec implementations no longer support AH, and according to RFC 4301 [21], *"Support for AH has been downgraded to MAY because experience has shown that there are very few contexts in which ESP cannot provide the requisite security services."* Thus support for AH is no longer a required part of IPsec implementations.

4 Encapsulating Security Payload

The new version of ESP is specified in RFC 4303 [23]. There have been some quite significant changes to the cryptographic elements of ESP since the previous version in RFC 2406 [8] and the first version in RFC 1827 [1]. The major innovation in RFC 4303 is the inclusion of support for what are called *combined mode algorithms*. These are cryptographic transforms that offer both confidentiality and integrity services in one package, more commonly called authenticated encryption algorithms in the cryptographic literature. As with RFC 2406, RFC 4303 also allows the use of one or both (but not neither) of an encryption algorithm and a MAC algorithm. By contrast, RFC 1827 only included support for encryption in ESP, with integrity protection coming from AH. We will discuss integrity protection in ESP in more detail in Section 4.4, and combined mode algorithm support in Section 4.5. Another difference from earlier versions of ESP is that RFC 4303 includes mechanisms for arbitrary length traffic padding as well as generation and processing of dummy packets. These features allow for limited traffic flow confidentiality, and we discuss them in Section 4.3. Before all that, however, we begin with discussion of encryption in ESP (Section 4.1) and of how ESP handles sequence numbers (Section 4.2).

4.1 ESP encryption

It is clear from the evolution of ESP that its primary purpose is to provide a confidentiality service. In transport mode, ESP encryption transforms an IP packet by encrypting specified parts of the packet and then inserting an ESP header and an ESP trailer into the packet. In tunnel mode, the operation is similar, except that the encrypted portion includes the entire inner packet and a new outer header is created. The resulting output (excluding the packet header) is illustrated in Figure 2. Note, however, that this diagram does not apply for combined mode algorithms, which may not have explicit IV or ICV fields, for example.

The ESP header consists of the SPI and Sequence Number Fields. This is followed by the IV (if any) and the encrypted data. The scope of encryption includes any padding, the Pad Length field and the Next Header field. The ICV (MAC value), if present, comes last.

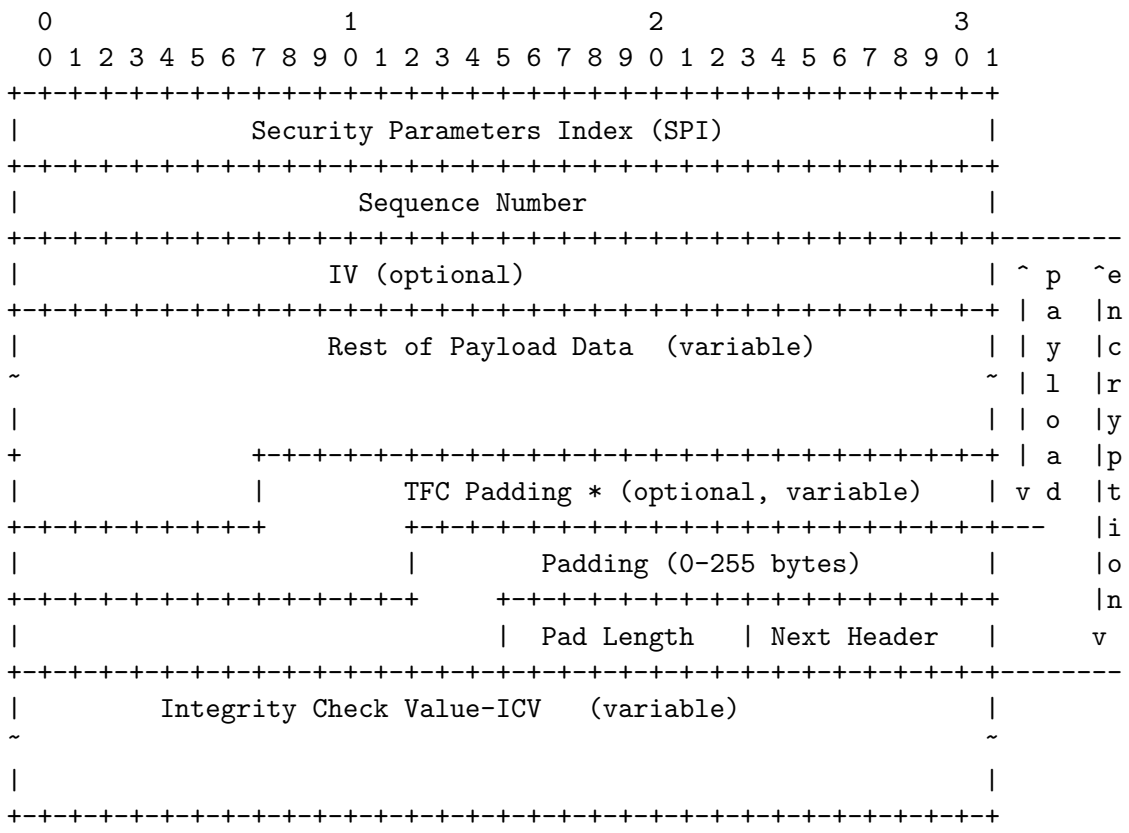

Figure 2: ESP format according to RFC 4303 [23], modified to show scope of encryption.

Algorithm	Requirement	Key size (bits)	Block size (bits)	Reference
NULL	MUST	0	N/A	RFC 2410 [12]
TripleDES-CBC	MUST−	192	64	RFC 2451 [13]
AES-CBC	SHOULD+	128	128	RFC 3602 [17]
AES-CTR	SHOULD	128	N/A	RFC 3686 [19]
DES-CBC	SHOULD NOT	56	64	RFC 2405 [7]

Table 2: Mandatory encryption algorithms for ESP specified in RFC 4305

The Security Parameters Index (SPI) field in the ESP header is used in the same way as the SPI field in AH. The same is true of the Next Header field. We will discuss ESP sequence numbers in Section 4.2 and the ICV field in Section 4.4.

As with RFC 4302, RFC 4303 does not discuss any specific encryption algorithms. However, RFC 4305 [25] specifies a number of encryption algorithms which are now mandatory to implement for ESP and these are listed in Table 2. Other algorithms are allowed. For example, RFC 2451 [13] specifies a number of other algorithms that could be used with ESP, while [30] is an RFC describing how the block cipher Camellia can be used with ESP.

Some comments on Table 2 are in order. Since ESP encryption is optional, support for the "NULL" algorithm is required. This algorithm is specified in its own RFC, RFC 2410, a document which has some amusement value. Concerning DES-CBC, RFC 4305 states: *"DES, with its small key size and publicly demonstrated and open-design special-purpose cracking hardware, is of questionable security for general use."* Our view is that this statement itself is questionable: 56 bits of security is adequate for many everyday applications, DES is still a broadly trusted algorithm, and inclusion of only algorithms with longer key lengths could potentially raise export control issues. Support for TripleDES-CBC is rated "MUST−", meaning that it is likely to be down-graded in future. This is the only algorithm in the list for which the RFCs state that a key larger than 128 bits must be supported, but the use of TripleDES with 48 rounds of encryption usually imposes a performance penalty. Therefore it would seem prudent to promote another high-strength algorithm (say AES-CBC with 256 bit keys) to "MUST", whether or not the down-grading of TripleDES-CBC takes place. RFC 3686 states that if AES-CTR is used in ESP, then it must be accompanied by a non-NULL integrity protection algorithm. This is because AES-CTR uses a mode of operation of AES to create a stream cipher, and so is consequently vulnerable to simple plaintext manipulation without some additional integrity protection.

As is apparent from Figure 2, two types of padding may be present in an ESP-protected packet. We will discuss TFC padding in Section 4.3. The remaining padding is included so that the total number of bytes in the processed sequence is a multiple of the number of bytes in a block of the encryption algorithm (for example, 8 for DES and 16 for AES). It is permissable for the padding to be of variable length and to extend over multiple blocks. This might aid in preventing traffic analysis. The Pad Length field must be included and indicates the total number of padding bytes (excluding the Pad Length byte itself). An ESP encryption algorithm may specify its own padding rule; otherwise a default rule is specified in [23, Section 2.4]. According to RFC 4303, when this rule is used, the receiver should inspect the padding fields, but the RFC does not specify what should happen in the event of an error. Possible actions involve dropping the packet silently, logging an error, or even reporting an error to the sender. However, it can be inferred from the text of [23, Section 2.4] that the offending

packet should at least be dropped. We know of one IPsec implementation (the Linux kernel implementation) where the padding is inspected but no action taken in the event of an error. Work of Vaudenay [52] and Canvel *et al.* [37] shows that improper handling of padding errors can undermine security.

4.2 ESP Sequence Number Field

Sequence numbers, including Extended Sequence Numbers (ESNs), are treated in largely the same way in RFC 4303 as they are in the AH RFC, RFC 4302. In particular, their use by the receiver is optional, but their inclusion in ESP headers is mandatory. The only real difference is that sequence numbers must be ignored by the recipient if the relevant ESP SA specifies the NULL integrity protection algorithm (in other words, if the SA only offers encryption). In this situation, ESP cannot offer an anti-replay service. If a combined mode algorithm is in use, the most significant bits of an ESN may actually be transmitted; if separate integrity and encryption algorithms are used, these bits are not transmitted, but are included in the MAC calculation by placing them in the ESP trailer, so they are split into two parts (as in AH).

4.3 Traffic flow confidentiality

ESP in RFC 4303 now has two mechanisms for providing traffic flow confidentiality, that is, the provision of spurious traffic to frustrate an attacker's attempts to gather information from the mere existence of IPsec protected traffic, or from statistics concerning that traffic. These mechanisms are variable length padding and dummy packets.

As we've seen above, the ESP padding is limited to 255 bytes, and this may not be sufficient to meet traffic flow confidentiality requirements. Section 2.7 of RFC 4303 outlines how special padding, called TFC padding, can be inserted after the payload data. This is in addition to the normal ESP padding. However, TFC padding can only be used if the receiver is able to unambiguously remove it using information about the proper payload length that is embedded in the payload itself. This will be possible, for example, in tunnel mode, where the Total Length field in the inner packet header gives the needed information.

Dummy packets can be indicated simply by using 59 for the protocol value in the Next Header field and otherwise creating a normal ESP header and trailer. (Recall that the Next Header field will be encrypted if encryption is selected, and so is hidden from the attacker.) Traditionally, this field indicated "No next header". According to RFC 4303, a receiver must discard any such packet without generating an error message.

The earlier version of ESP in RFC 2406 supported neither of these mechanisms, but did allow variable length padding up to a maximum of 255 bytes as part of the ESP encryption process.

4.4 Integrity protection in ESP

As we have noted above, the original version of ESP in RFC 1827 included only support for encryption and not integrity protection. Work of Bellovin [35] was influential in persuading the IPsec standards community of the need for strong integrity protection as an adjunct to encryption, in order to prevent active attacks from defeating the confidentiality services offered by ESP. For efficiency reasons, it was decided to include support for an optional

Algorithm	Requirement	Key size (bits)	Output size (bits)	Reference
HMAC-SHA1-96	MUST	160	96	RFC 2404 [6]
NULL	MUST	N/A	0	
AES-XCBC-MAC-96	SHOULD+	128	96	RFC 3566 [16]
HMAC-MD5-96	MAY	128	96	RFC 2403 [5]

Table 3: Mandatory MAC Algorithms for ESP specified in RFC 4305

integrity service along with confidentiality services in the second version of ESP, RFC 2406. This support has been preserved in RFC 4303.

As with AH, the integrity service in ESP is provided by a MAC algorithm. The inputs to the MAC calculation are the fields in the ESP header, payload and trailer, with the ICV bytes being set to zero for the purposes of MAC calculation. The calculation will include all ESN bits as noted above, allowing an anti-replay service to be offered if integrity protection is present. Details of how this service is implemented are largely as for AH, with the calculated MAC being placed in the ICV field. However, notice that no fields of the IP packet header are protected by the MAC in ESP, in contrast to AH. Thus the protection offered by the MAC algorithm is somewhat less than in AH. However, a roughly equivalent level of protection can be arranged for an IP header by using ESP in tunnel mode, in which case the integrity protection will cover the entire inner packet, including all the header fields.

When ESP provides both encryption and integrity protection, the encryption is performed first and then the MAC algorithm is applied. This allows rapid rejection of corrupted packets by the recipient, as the MAC will be checked before decryption needs to be performed. If the MAC fails, the receiver must reject the packet. Moreover, this is an auditable event [23].

In keeping with the separation principle, RFC 4305 [25] (rather than RFC 4303) specifies which MAC algorithms must be supported by IPsec implementations. The list of algorithms is replicated in Table 3. Notice that the algorithms are identical to those mandated for AH except for the inclusion of the NULL algorithm here. The NULL algorithm is needed to allow "encryption-only" operation of ESP. Note that while authentication and encryption can each be NULL in ESP, they must not both be NULL: ESP can offer one or both of encryption and integrity protection, but not neither. There is no RFC specifying the NULL MAC algorithm.

It might appear surprising that RFC 4303 still allows encryption-only operation of ESP, especially after Bellovin's work [35] demonstrating its security flaws. There is also significant theoretical work supporting the general adoption of integrity protection along with encryption [33, 34, 43, 44]. It appears that the need for backward compatibility with RFC 1827 meant that support for encryption-only configurations of ESP was mandated in RFC 2406. As compensation, the second generation of RFCs are explicit in spelling out the dangers of using encryption-only configurations. RFC 4303 removes the mandatory aspect, stating that implementations *may* support encryption-only, and repeats the warnings of RFC 2406. However, RFC 4303 indicates that ESP allows encryption-only because it *"may provide better performance and still provide adequate security, e.g., when a higher-layer authentication/integrity protection is offered independently."* In fact, in recent work [49], it has been shown that at least one implementation of IPsec is fatally weak in tunnel mode, encryption-only configurations: active attacks that are ciphertext-only and that can operate in essentially real-time have been demonstrated in the laboratory under realistic networking conditions against the

Linux kernel implementation of IPsec. It is argued in [49] that placing warnings of the dangers of encryption-only ESP in RFCs is probably insufficient to prevent inexpert end-users from selecting such configurations; indeed there exist many on-line configuration guides which either do not warn of the dangers or which actively encourage users to select encryption-only. It is also shown in [49] that provision of higher-layer integrity protection can not prevent the attacks. Thus the advice concerning such provision in RFC 4303 may be misleading. It should be noted, however, that the attacks in [49] are not attacks against the RFCs themselves. This is because they rely for their success on the failure of the Linux implementation to properly implement post-cryptographic processing checks specified in the architectural RFCs 2401 [3] and 4301 [21]. See also [46] for unimplemented sketches of similar kinds of attacks against transport mode ESP.

4.5 Combined mode algorithms

Combined mode algorithms offering both confidentiality and integrity services are new to ESP in RFC 4303. Under many circumstances, combined mode algorithms should provide significant efficiency gains compared to sequential execution of encryption and MACing. But their inclusion in ESP is not completely straightforward, because it is desirable to integrity protect more fields than need to be encrypted, and not all combined mode algorithms will support this requirement. In current cryptographic terminology, algorithms that do are said to solve the "authenticated encryption with associated-data" (AEAD) problem. Reference [51] provides a pragmatic overview of recent work in this area.

In order to accommodate the range of possible combined mode algorithms (those solving the AEAD problem and those not), no detailed ESP format is specified for combined mode. Instead, issues such as details of IVs, padding requirements, possible replication of SPI and Sequence Number fields in the processed payload, and whether or not an ICV field is needed, are deferred to RFCs for individual combined mode algorithms. Patents are an important aspect here, with many of the "popular" designs for combined mode algorithms being encumbered. Perhaps this explains in part why RFC 4305 neither mandates nor suggests any specific combined mode algorithms for use in ESP. However, one suitable algorithm that appear to be free of patents already exists as an RFC: AES CCM mode, as specified in RFC 4309 [29]. There is no version number in ESP, and no mechanism for a peer to discover which version of ESP another peer is using. Thus another reason for not mandating support for combined mode algorithms is that it assists in making ESP as specified in RFC 4303 backward compatible with earlier versions of ESP.

5 Cryptographic Key Management in IPsec

So far in our discussion of IPsec, we have assumed that all necessary cryptographic parameters, including keys and algorithms, are already in place before AH or ESP processing begins. In this section, we give a brief overview of how IPsec handles management of cryptographic keys and other parameters.

We have already noted that SAs are used as a repository for cryptographic parameters (including keys), and that SPIs provide a link between SAs and IPsec-protected packets themselves. At a higher level, the SPD defines policies for applying IPsec and links to SAs for implementing those policies. Typically, the SPD is populated by hand (usually via a vendor-supplied GUI) at each IPsec aware host. This is a reasonable approach, though the

likelihood of misconfiguration resulting in security failures or accidental denial of service is quite high because of the number and complexity of options available. It is also possible, but much less convenient, to set up the individual SAs manually, using static keys. This approach is reasonable for testing purposes or small-scale deployment of IPsec, but does not scale at all well. Moreover, it makes no provision for rekeying, the need for which can be triggered by a number of events including sequence number overflow, or an SA exceeding its lifetime (which can be determined, for example, by using a timer, or by IPsec reaching a pre-set limit on the amount of data the SA is allowed to process). A better approach is to use IKE or IKEv2, which provide automated mechanisms for setting up and managing SAs in accordance with policies in the SPD. We discuss IKE and IKEv2 next.

5.1 Internet Key Exchange

The IKEv2 protocol as defined in RFC4306 [26] is a very complex and flexible specification which had a long gestation period in the IPsec working group. We will only provide an overview of its features and a brief comparison with IKEv1 here. Further discussion of the IKEv2 design and features can be found in [40].

At a high level, IKEv2 is a cryptographic protocol involving the exchange of pairs of messages between two peers. In the first exchange, called the IKE_SA_INIT exchange, the two peers exchange information concerning cryptographic algorithms and other security parameters they are willing to use along with nonces and Diffie-Hellman (DH) values. The DH values are used to create keying material SKEYSEED from which a variety of keys are derived using a pseudo-random function (PRF). Included amongst these are keys for encryption and integrity protection algorithms for a special SA called the IKE SA. This SA defines parameters for a secure channel between the peers over which subsequent message exchanges take place. It provides ESP-like cryptographic processing of IKEv2 payloads (though without any support for combined mode algorithms). In the second exchange, now protected by the IKE SA and called the IKE_AUTH exchange, the two parties authenticate one another and set up a first SA to be placed in the SADB and used for protecting ordinary (i.e. non-IKE) communications between the peers. Thus four messages are needed to establish the first SA for general use.

Subsequent exchanges are known as CREATE_CHILD_SA and INFORMATIONAL exchanges. The first of these can be used to establish further SAs for protecting traffic, and may involve an exchange of new, ephemeral Diffie-Hellman values to provide perfect forward security (PFS). The second is used to exchange management information, IKEv2 error messages and so on.

The IKE_SA_INIT exchange allows two peers to negotiate many features of the ensuing exchanges, including parameters for the Diffie-Hellman exchange, algorithms to be used in the IKE SA, acceptable authentication methods, and so on. There are three main authentication methods supported by IKEv2: signature-based, MAC-based, and EAP-based. The first of these implies the use of public key cryptography and a PKI to support this; the second requires a pre-established symmetric key to be in place between the two peers (but note that, because of the structure of IKEv2, this key can be used to authenticate the negotiation of many SAs). A good discussion of the pros and cons of these two competing approaches can be found in [42, Section 4.2.2]. The third approach makes use of the Extensible Authentication Protocol (EAP) as defined in [20]. This allows a wide range of legacy authentication methods to be integrated with IKEv2.

All of the IKEv2 payloads comprising the above exchanges are specified at a bit/byte level in [26, Section 3]. Keeping in mind the degree of flexibility offered by IKEv2, it is then perhaps not so surprising that the IKEv2 specification is as long as it is.

RFC 4307 [27] specifies algorithms and parameters for use in IKEv2 which are mandatory to implement. Thus it is a counterpart to RFC 4305 which does the same job for AH and ESP. RFC 4307 identifies two Diffie-Hellman groups, one with a 1024-bit modulus from [11] and the other with a 2048-bit modulus from [15]. It also lists the same encryption and integrity protection algorithms (with the same levels of support) as are recommended for use with ESP in RFC 4305, see Tables 2 and 3. Finally, it lists three PRFs, PRF_HMAC_MD5 and PRF_HMAC_SHA1 from [2], and PRF_AES128_CBC from [18, 31]. This last PRF is identical to the function AES-XCBC-MAC-96 from [16] except that it does not truncate the output to 96 bits. PRFs are used for various purposes in IKEv2, including key derivation. RFC 4308 [28] defines two suites of cryptographic algorithms that are optional to implement. Each suite defines a full set of algorithms (including DH groups) needed for ESP encryption/integrity protection and for IKEv1/IKEv2.

Some of the major similarities and differences between the IKEv1 and IKEv2 protocols are as follows:

- Both protocols operate in two phases: Phase 1 (in main mode or aggressive mode) and Phase 2 (quick mode) in IKEv1 are comparable to IKEv2's IKE_SA_INIT and IKE_AUTH exchanges, respectively.
- Both protocols have options for identity-protection (via encryption of identity-related data), anti-denial-of-service (via cookies), and perfect forward security.
- Phase 1 of IKEv1 allows four different authentication methods in its two different modes, with a quite different message syntaxes being possible in the various cases. It is claimed that the equivalent function in IKEv2 is much simpler, with a single four-message protocol made up of two exchanges. However, the EAP authentication method does involve modifying the message structure of IKEv2 to some extent.
- IKEv1's quick mode is roughly equivalent to IKEv2's CREATE_CHILD_SA, though IKEv2 uses one message less than IKEv1.
- IKEv2 is more efficient in setting up the first non-IKE SA: this SA can be established as part of the IKE_AUTH exchange after four messages, while IKEv1 requires at least six messages (using aggressive mode's three messages followed by three more for quick mode).
- IKEv1 and IKEv2 both run over the unreliable UDP protocol, but IKEv2 adds retransmission and acknowledgement functions, so it is more reliable than IKEv1.
- IKEv2 uses an ESP-like transform to protect IKE payloads after the IKE_SA_INIT exchange, whereas IKEv1 uses somewhat different methods defined for the general purpose protocol ISAKMP in [10].

A more detailed list of differences between IKEv1 and IKEv2 can be found in [26, Appendix A].

It was intended at the outset of the IKEv2 design process that it should produce a leaner, cleaner, more complete and more easily understood design. By these measures, IKEv2 has

been only partially successful: it is still a fairly complex protocol, and bringing in support for EAP along with signature-based and MAC-based methods has led to some loss of clarity in the protocol description. Efforts are already underway to address certain deficiencies of the IKEv2 specification in IKEv2.1 [41].

6 Concluding Remarks

It seems fair to say that the different generations of IPsec RFCs represent a cryptographic evolution rather than a revolution. Some new features have been added (notably combined mode algorithms in ESP), and new algorithms and key lengths have been brought into play. The overall clarity and quality of the standards has definitely improved, perhaps as a consequence of having a better-informed pool of experts in the working group coupled with a greater understanding of the need to address interoperability through clearer specification.

The development of the RFCs does seem to have been constrained by the need to maintain backward compatibility. This is evident, for example, in the way that ESP still allows encryption-only configurations but does not mandate support for combined mode algorithms. This seems like a missed opportunity. The attempt to improve IKE has succeeded in many respects: IKEv2 is simpler and more self-contained, and the specification is more explicit. But including support for EAP has also introduced complexity, an almost inevitable consequence of the desire for flexibility.

IPsec has been analyzed from a cryptographic perspective over the years – see for example [50, 39]. There is also some evidence that the provable security philosophy has now had an influence on the IPsec RFCs. As examples, HMAC, defined and proven secure in [32] is heavily used in IPsec algorithms, while RFC 4303 cites [44] for evidence of the security issues of applying a MAC to data before encryption, and SIGMA, as defined in [36, 45] forms the basis for the design of IKEv2's signature-based authentication mode. Yet few of the cryptographic design features in the IPsec RFCs appear to have been chosen because they have a firm theoretical basis. Thus IPsec presents many interesting challenges for theoreticians seeking motivation from real-world security problems. The complexity of IKEv2 means that only simplified versions have been amenable to analysis to date [36, 45]. The full IKEv2 protocol surely deserves a formal analysis. (Protocols studied in theory tend to be describable in six lines or less – compare this to the nearly 100 pages needed to fully specify IKEv2 in RFC 4306!) The development of further, preferably patent-free, AEAD algorithms suitable for use in ESP would be useful. Studying the impact of recent cryptanalysis of hash functions on the security of HMAC could also be an interesting exercise: how severe can the attacks on the underlying hash function be while still maintaining a reasonable level of security for HMAC?

The IPsec working group has now been wound up after completing its efforts on the new generation of IPsec standards. But IETF standardization activity is likely to continue for sometime in three related areas concerned with deployment and flexibility of IPsec. The PKI4IPsec working group is developing PKI and certificate standards for IPsec, with the aim of eventually increasing the use of public key authentication methods in IPsec deployments. The IKE Mobility working group is working on extensions to IKEv2 protocol to enable its use in multihoming, mobile and roaming contexts. The Better-Than-Nothing Security (btns) working group is tasked with specifying extensions to the IPsec architecture to allow IPsec to support the use of unauthenticated SAs, with the goal of enabling simpler and more rapid deployment of IPsec in contexts where use of such unauthenticated SAs is appropriate.

In closing, we reiterate that cryptography is only one part of the IPsec puzzle. However, getting the cryptography right is vital in ensuring that IPsec delivers the security expected of it. We hope that this article will open up the new family of IPsec RFCs to a wider range of cryptographic researchers, encouraging them to work in a fascinating area located at the boundary between theory and practice. We also hope that this article has provided a useful and accessible summary of recent developments in IPsec for non-cryptographers.

References

[1] R. Atkinson, "IP Encapsulating Security Payload (ESP)." RFC 1827, August 1995.

[2] H. Krawczyk, M. Bellare and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication." RFC 2104, Feb. 1997.

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

[4] S. Kent and R. Atkinson, "IP Authentication Header." RFC 2402, Nov. 1998.

[5] C. Madson and R. Glenn, "The Use of HMAC-MD5-96 within ESP and AH." RFC 2403, Nov. 1998.

[6] C. Madson and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH." RFC 2404, Nov. 1998.

[7] C. Madson and N. Doraswamy, "The ESP DES-CBC Cipher Algorithm With Explicit IV." RFC 2405, Nov. 1998.

[8] S. Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)." RFC 2406, Nov. 1998.

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

[10] D. Maughan, M. Schertler, M. Schneider and J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)." RFC 2408, Nov. 1998.

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

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

[13] R. Pereira and R. Adams, "The ESP CBC-Mode Cipher Algorithms." RFC 2451, Nov. 1998.

[14] A. Keromytis and N. Provos, "The Use of HMAC-RIPEMD-160-96 within ESP and AH." RFC 2857, June 2000.

[15] M. Kojo and T. Kivinen, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)." RFC 3526, May 2003.

[16] S. Frankel and H. Herbert, "The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec." RFC 3566, Sept. 2003.

[17] S. Frankel, R. Glenn and S. Kelly, "The AES-CBC Cipher Algorithm and Its Use with IPsec." RFC 3602, Sept. 2003.

[18] P. Hoffman, "The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)." RFC 3664, Jan. 2004.

[19] R. Housley, "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)." RFC 3686, Jan. 2004.

[20] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)." RFC 3748, June 2004.

[21] S. Kent and K. Seo, "Security Architecture for the Internet Protocol." RFC 4301, Dec. 2005.

[22] S. Kent, "IP Authentication Header." RFC 4302, Dec. 2005.

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

[24] S. Kent, "Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP)." RFC 4304, Dec. 2005.

[25] D. Eastlake 3rd, "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)." RFC 4305, Dec. 2005.

[26] C. Kaufman (ed.), "Internet Key Exchange (IKEv2) Protocol." RFC 4306, Dec. 2005.

[27] J. Schiller, "Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)." RFC 4307, Dec. 2005.

[28] P. Hoffman, "Cryptographic Suites for IPsec." RFC 4308, Dec. 2005.

[29] R. Housley, "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)." RFC 4309, Dec. 2005.

[30] A. Kato, S. Moriai and M. Kanda, "The Camellia Cipher Algorithm and Its Use With IPsec." RFC 4312, Dec. 2005.

[31] P. Hoffman, "The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)." RFC 4434, Feb. 2006.

[32] M. Bellare, R. Canetti and H. Krawczyk, "Keying hash functions for message authentication." In *N. Koblitz (ed.), Advances in Cryptology – CRYPTO'96*, LNCS Vol. 1109, Springer-Verlag, 1996, pp. 1–15.

[33] M. Bellare and C. Namprempre, "Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm." In *T. Okamoto (ed.), Advances in Cryptology – ASIACRYPT 2000*, LNCS Vol. 1976, Springer-Verlag, 2000, pp. 531–545.

[34] M. Bellare and P. Rogaway, "Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography." In *T. Okamoto (ed.), Advances in Cryptology – ASIACRYPT 2000*, LNCS Vol. 1976, Springer-Verlag, 2000, pp.317–330.

[35] S. Bellovin, "Problem Areas for the IP Security Protocols." In *Proceedings of the Sixth Usenix Unix Security Symposium*, pp. 1–16, San Jose, CA, July 1996.

[36] R. Canetti and H. Krawczyk, "Security Analysis of IKE's Signature-based Key-Exchange Protocol." In *V. Shoup (ed.), Advances in Cryptology – CRYPTO 2002*, LNCS Vol. 3621, Springer-Verlag, 2002, pp. 143–161.

[37] B. Canvel, A.P. Hiltgen, S. Vaudenay and M. Vuagnoux, "Password Interception in a SSL/TLS Channel." In *D. Boneh (ed.), Advances in Cryptology – CRYPTO 2003*, LNCS Vol. 2729, Springer-Verlag, 2003, pp. 583–599.

[38] N. Doraswamy and D. Harkins. *IPsec: the new security standard for the Internet, Intranets and Virtual Private Networks (second edition)*, Prentice Hall PTR, 2003.

[39] N. Ferguson and B. Schneier, "A cryptographic evaluation of IPsec." Unpublished manuscript available from `http://www.schneier.com/paper-ipsec.html`, Feb. 1999.

[40] D. Harkins, C. Kaufman, T. Kivinen, S. Kent and R. Perlman, "Design Rationale for IKEv2." Internet draft, available from `http://tools.ietf.org/html/draft-ietf-ipsec-ikev2-rationale-00.txt`, expired Aug. 2002.

[41] P. Hoffman, "Internet Key Exchange Protocol: IKEv2.1." Internet draft available from `http://www.ietf.org/internet-drafts/draft-hoffman-ikev2-1-00.txt`, Jan. 2006, expires 5th July, 2006.

[42] S. Frankel, K. Kent, R. Lewkowski, A.D. Orebaugh, R.W. Ritchey and S.R. Sharma, "Guide to IPsec VPNs." NIST Special Publication 800-77 (Draft), January 2005.

[43] J. Katz and M. Yung, "Unforgeable encryption and chosen ciphertext secure modes of operation." In *B. Schneier (ed.), FSE 2000*, LNCS Vol. 1978, Springer-Verlag, 2001, pp. 284–299.

[44] H. Krawczyk, "The Order of Encryption and Authentication for Protecting Communications (Or: How Secure Is SSL?)." In *J. Kilian (ed.), Advances in Cryptology – CRYPTO 2001*, LNCS Vol. 2139, Springer-Verlag, 2001, pp. 310–331.

[45] H. Krawczyk, "SIGMA: The 'SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and Its Use in the IKE-Protocols." In *D. Boneh (ed.), Advances in Cryptology – CRYPTO 2003*, LNCS Vol. 2729, Springer-Verlag, 2003, pp. 400–425.

[46] C.B. McCubbin, A.A. Selcuk and D. Sidhu, "Initialization vector attacks on the IPsec protocol suite." In *9th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2000)*, IEEE Computer Society, 2000, pp. 171–175.

[47] NIST, "Brief Comments on Recent Cryptanalytic Attacks on Secure Hashing Functions and the Continued Security Provided by SHA-1." Available from `http://www.csrc.nist.gov/hash_standards_comments.pdf`, 25th August 2004.

[48] NIST Federal Information Processing Standards Publication 197, "Specification for the Advanced Encryption Standard (AES)." 26th Nov. 2001.

[49] K.G. Paterson and A.K.L. Yau, "Cryptography in Theory and Practice: The Case of Encryption in IPsec." In *S. Vaudenay (ed.), Advances in Cryptology – EUROCRYPT 2006*, LNCS Volume 4004, Springer-Verlag, to appear. Full version available from `http://eprint.iacr.org/2005/416`.

[50] P. Rogaway, "Problems with Proposed IP Cryptography." Internet draft available from `http://www.cs.ucdavis.edu/~rogaway/papers/draft-rogaway-ipsec-comments-00.txt`, April 1995.

[51] G. Rose, "Combining Message Authentication and Encryption." Unpublished manuscript available from `http://www.qualcomm.com.au/PublicationsDocs/Enc+MAC%20paper.pdf`.

[52] S. Vaudenay, "Security flaws induced by CBC padding – applications to SSL, IPSEC, WTLS..." In *L.R. Knudsen (ed.), Advances in Cryptology – EUROCRYPT 2002*, LNCS Vol. 2332, Springer-Verlag, 2002, pp. 534–545.

[53] X. Wang, D. Feng, X. Lai and H. Yu, "Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD." Cryptology ePrint Archive: Report 2004/199, available from `http://eprint.iacr.org/2004/199`, 2004.

[54] X. Wang, Y.L. Yin and H. Yu, "Finding Collisions in the Full SHA-1." In *V. Shoup (ed.), Advances in Cryptology – CRYPTO 2005*, LNCS 3621, Springer-Verlag, 2005, pp. 17–36.