A Survey of the Cryptographic Protocols of IPsec

Craig Tomkow
6-10539 85 Ave
Edmonton, AB T6E 2K5
Canada
ctomkow@gmail.com

ABSTRACT

The internet protocol security (IPsec) suite was introduced as a mechanism for securing internet protocol (IP) layer traffic. The suite emerged from a combination of historical security protocols and frameworks to establish a new standard. IPsec contains an internet key exchange (IKE) protocol that creates security associations (SA) between endpoints to facilitate the exchange of cryptographic details. There are two main header protocols, the authentication header (AH) and the encapsulating security payload (ESP) meant to achieve a combination of authentication, integrity, and secrecy. While extensive research has evaluated IPsec to ensure it is cryptographically sound, an emerging trend of simplification and reduced complexity turned IPsec into a de facto standard for secure IP communications.

Keywords

IPsec, Network Security, AH, ESP, IKE

1 Introduction

Internet protocol security (IPSec) has been the cryptographic workhorse of the last decade. IPsec is not a single encryption algorithm, it is a protocol suite for securing modern network and system infrastructure. The introduction of the IP Security request for comments (RFC) 2411 in 1998 [14] directly addressed a major concern of securing the rapidly growing Internet. While the transmission control protocol (TCP) and internet protocol (IP) were mature enough to result in the growth of a reliable infrastructure, security was largely missing. Prior to IPsec, the National Institute of Standards and Technology (NIST) worked on a security protocol at layer 3 (SP3) while an authentication header was published in RFC 1826 in 1995 [2]. Directly building upon the previous work, IPsec was introduced as a comprehensive protocol suite for the IP layer providing software developers an open standard to implement. This paper will survey the protocol suite covering the main cryptographic protocols while briefly touching of relevant research related to IPsec. Due to the complexity of IPsec, a comprehensive review is not possible. However, the fundamentals of IPsec will provide proper context for delving into the core cryptographic mechanics of IPsec. The resilience of IPsec allowed it to become the de facto standard for IP security.

2 IPsec Overview

To aid in the understanding of the following cryptographic protocols, a brief overview of IPsec is necessary. The aim is to provide confidentiality, integrity, and authenticity. This is accomplished through a comprehensive framework of standards and protocols defined by the internet engineering task force (IETF) in RFC 6071 [5]. Unlike security being implemented at the application layer such as the transport layer security (TLS) protocol, IPsec exists within the network layer of the OSI model. To accomplish this, implementation deals with adding or replacing IP headers to the data passing through endpoint gateways or hosts. Due to the modification of IP headers, the protocol was designed to accommodate the internet protocol version 4 (IPv4) and the internet protocol version 6 (IPv6). One of the most common use cases for IPsec involve creating a point to point virtual private network (VPN) connection. This allows a remote host, gateway, or network to communicate with another network securely. Another common use for IPsec is authenticating routers when communicating routing information via the open shortest path first (OSPF) routing protocol. Due to the flexibility of IPsec, it can be implemented between virtually any two endpoints.

Within IPsec there are a couple main concepts for the creation of a secure connection. The first is the requirement to exchange authenticated cryptographic keys for encryption of the payloads. Today, the internet key exchange (IKE) protocol, derived from earlier Oakley and SKEME protocols and the internet security association and key management protocol (ISAKMP) framework [6], provides the method of securely exchanging keys for encryption of traffic. Within the key exchange process, IKE also facilitates the creation of security associations (SA) which negotiate the cryptographic protocols between endpoints to facilitate secure communication. The SA is tied to a security parameter index (SPI) present in a security policy database (SPD) in the endpoints; it also exists in the header added to packets identifying which traffic belongs to a certain cryptographic communication stream. Another major concept in IPsec is the header type added to the traffic, either the authentication header (AH) or the encapsulating security payload (ESP). As it may be evident from their descriptions, the former provides authentication and integrity services while the latter can also provide secrecy. Finally, there is two operating modes that IPsec can be implemented, transport or tunnel mode. Transport mode dictates that the original IP header not be overwritten while tunnel mode adds a new IP header to the packet. It is also worth noting that both AH and ESP protocols can be used in each operating mode.

3 Cryptographic Protocols

To investigate the core protocols of IPsec, some clarification regarding protocol version is required. IPsec was released 1998 and since then, there have been multiple protocol iterations. To keep this text current, the latest versions of protocols are covered. IPsec currently is in version three (IPsec-v3) while IKE is in version two (IKEv2). This is important as the changes to the IKE protocol warrants attention as a significant amount of existing documentation only covers IKEv1.

3.1 Internet Key Exchange Version 2

The core goal of IKEv2 is to facilitate the creation of SAs. The initial exchange between two endpoints is meant to establish a secure communication stream to accommodate all future SA negotiation. IKEv2 first starts with the IKE_SA_INIT exchange which results in the IKEv2 SA. The messages exchanged include HDR, the IKEv2 header that sits between the payload and the user datagram protocol (UDP) header. The HDR includes things such as the SPI and IKE version. SA is the security association payload stating the supporting cryptographic algorithm(s) supported (IKE, ESP, AH), the transforms supported for each algorithm (encryption algorithm such as 3DES, integrity algorithm such as HMAC_SHA1_96, the pseudo random function (PRF) for authentication such as HMAC_MD5, and the Diffie-Hellman group such as 1024-bit MODP). Nonces (N_i and N_r), single use pseudo-random numbers are exchanged which are signed in the next exchange of information for authentication. The KE payload is the public Diffie-Hellman numbers required to create a shared key between endpoints for the IKEv2 SA symmetric encryption. Remember the Diffie-Hellman computation underpinning the creation of KE and the subsequent shared key.

$$KE_i = g^i mod\ p$$

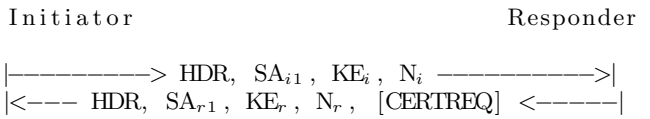
$$KE_r = g^r mod\ p$$

Each side then performs the same operation with the received KE value. The following computation below shows the final step to produce a shared key for symmetric encryption.

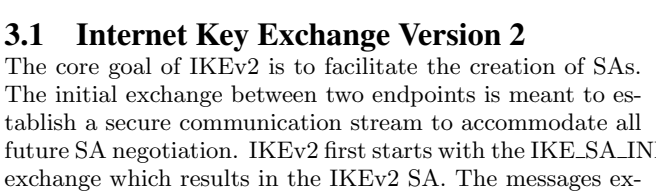
$$SK_e = {KE_i}^r mod\ p$$

$$SK_e = {KE_r}^i mod\ p$$

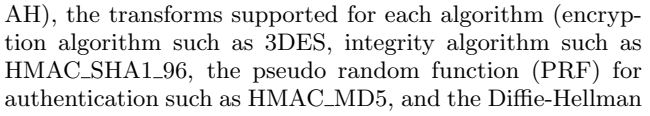
Below shows the IKE_SA_INIT exchange below, a total of two messages.



IKE_SA_INIT establishes an encrypted channel involving the negotiation of a symmetric encryption algorithm and the exchange of keys needed while agreeing on a hashing mechanism for integrity verification as part of the IKEv2 SA. Note, however, that the encrypted channel is not authenticated, therefore the next exchange is IKE_AUTH.
The IKE_AUTH exchange is meant to authenticate the encrypted communication, verifying the identity of the other side. The authentication is typically done through a digital signature which is used to sign the received nonce. A pre-shared secret can be used to generate the hashed message authentication code (HMAC) from the received nonce; the HMAC is represented by AUTH. Remember, in the IKE_SA_INIT exchange, the SA payload detailed the agreed upon PRF for generating the AUTH value. Additionally, it should be noted that a public key certificate can be optionally used to verify that the key used in the creation of the AUTH value indeed belongs to the ID indicated. Below is the IKE_AUTH exchange. Note that the payload and values within {..} are encrypted due to the IKE_SA_INIT process.



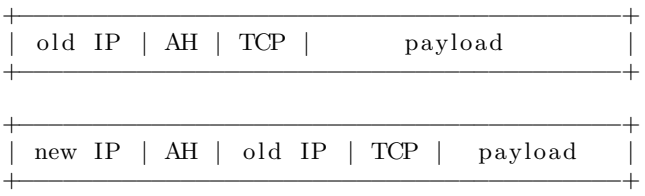
The two IKEv2 SA's (SA_1 and SA_2) set the groundwork for the secure negotiation of more SA's. The CREATE_CHILD_SA exchange is used for creating more SA's and also doubles as a mechanism to rekey any SA. The exchange is shown below.



Note, much of the details are similar: the security association, the nonce, and an optional Diffie-Hellman public number. The important difference are the traffic selectors (TS) which was also included in the IKE_AUTH exchange. The traffic selectors are variable in length that contain source/destination address and port information. This is required to determine which SA is associated with certain traffic flows. A source/destination address and port pairing can be associated with one SA, while other source/destination traffic and port can be associated with another SA.
While there are many other details surrounding IKEv2, the core goal of creating security associations between endpoints has been outlined. A key concept is the initial encryption tunnel created to facilitate further SA negotiation. Secondly, IKEv2 does not reproduce existing cryptographic protocols such as 3DES or MD5 but dynamically negotiates the cryptographic details and values required to establish an encrypted stream.

3.2 Authentication Header

The authentication header (AH) protocol is designed to verify origin of traffic and maintain the integrity of the payload; secrecy is not achieved due to the lack of payload encryption. The protocol specifies a header that is inserted into each packet. Recall that IPsec has two modes of operation, transport and tunnel mode. When AH operates in transport mode, the AH header is inserted before the IP header and after the transport header which ensures the packet is routed via the original IP header. When in tunnel mode, the AH header is inserted after the original IP header and a new IP header is added after the AH header. Therefore, the packet is routed via a new IP header. See the two packet structures below in transport and tunnel mode, respectively.



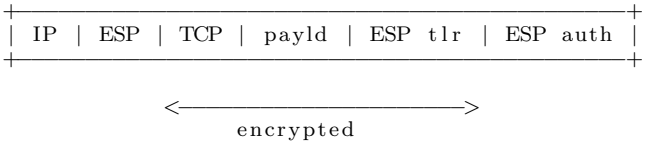
While the AH header specifies various details including the SPI which ties it to an SA, the authentication value in the header is the integrity check value (ICV). As with IKE_SA_INIT in the IKEv2 protocol, the cryptographic algorithm for computing the ICV is detailed within the SA. RFC 8221 [15] provides software designers requirements for what cryptographic protocols must be supported and what must not be. For example, below is a snapshot of such guidance for the ICV hashing algorithm.

MUST : AUTH_HMAC_SHA2_256_128
SHOULD : AUTH_HMAC_SHA2_512_256
MUST NOT : AUTH_HMAC_MD5_96

Whatever algorithm is negotiated between the endpoints, the ICV is computed over the immutable values within the headers and payload that is encapsulated by the AH header. Depending on the mode being used, it could include aspects of the AH header, IP fields, and payload. Special care needs to be taken to ensure the hashing is not done across mutable fields such as time to live (TTL) and differentiated services code point (DSCP), etc.
One last aspect of AH worth noting is it's anti-replay capability. A sequence number exists in the AH header which is incremented by one for each packet sent. This allows the receiver to discard any duplicate packets with a repeated sequence number, guarding against replayed packets.

3.3 Encapsulating Security Payload

While the AH protocol provides authentication and integrity, the encapsulating security payload (ESP) can provide those services as well as secrecy. ESP inserts a header near the front of the packet but also a trailer at the end of the packet as well. The ESP trailer has a couple uses. It contains the 'next header' protocol type (TCP, UDP, IPv6, etc.) and since the trailer is encrypted, it ensures confidentiality of the data. Furthermore, the trailer has a padding field that ensures the data to be encrypted is of a certain length ensuring compatibility with encryption algorithm length requirements such as the case with block ciphers. As with the AH protocol, ESP can operate in transport or tunnel mode dictating the placement of the ESP header which is identical to the AH protocol header. Additionally, there is an additional implicit ESP trailer (which includes padding) used in the integrity calculation, note however, the implicit trailer is not actually transmitted on the wire like the other ESP trailer. To provide a summary of ESP encryption of the packet, see the image below (in transport mode).



Some final notes regarding ESP. Authentication can be provided by using a built-in authentication mechanism (the encryption and integrity combined algorithm), or one can nest both ESP and AH protocols to achieve confidentiality, authentication, and secrecy. Doing so, however, adds unnecessary header overhead. Finally, remember that the negotiation of specific cryptographic algorithms occurs within the SA. As with the AH protocol, RFC 8221 [15] has specific requirements for cryptographic support, though ESP also has requirements for symmetric key algorithms.

MUST : ENCR_AES_CBC
SHOULD NOT : ENCR_3DES
MUST NOT : ENCR_DES

4 IPsec Research and Discussion

Due in-part to widespread adoption of IPsec, researchers have reviewed the suite to ensure it is cryptographically sound. While IPsec does not define asymmetric or symmetric encryption ciphers directly, it is nonetheless important in the implementation of cryptography solutions. As succinctly said by Ross Anderson, a preeminent professor of security engineering, 'the inappropriate use of mechanisms is one of the main causes of security failure' [1].
IPsec has been thoroughly evaluated by Ferguson and Schneier outlining key concerns [4]. Core among them is the complexity of the protocol suite. The increase in complexity adds to the likelihood of implementation error. Furthermore, it was pointed out that the lack of clear documentation of the standard adds to confusion and unclear understanding. In an attempt to clarify the protocol standards, IPsec-v3 tackles this issue by adding clarification and concision to the standards documents while implementing lessons learned [3]. Even so, a certain lack of clarity was discovered when the protocol suite was surveyed for this paper. Due to the number of RFCs needing to be referenced and at times having unclear language, it took great time and effort to decipher the precise instructions. However, Ferguson and Schneier do concede it is the best solution so far for IP layer security. More recent work has looked at the differences between IKEv1 and IKEv2 by S. Shaheen et al [13]. Their analysis of the two protocols revealed IKEv2 is less complex and more reliable than IKEv1, while addressing an increase of security concerns such as denial of service (DoS) and replay attacks. The researchers note that IKEv2 is specific to IPsec as well. This narrow focus may have contributed to a reduction of complexity resulting in a more secure protocol. Due to previous criticisms of complexity, it seems the current version of IKEv2 is directly addressing such issues to provide a more robust security suite. Over time, IPsec has matured to be an accepted de facto solution for securing IP traffic. Current research now seems to focus on the performance impact of IPsec and applying the protocol suite to novel areas. While IPsec is not perfect, it currently provides a reasonable security foundation to improve upon.

5 Conclusion

IPsec is the currently accepted solution for IP layer security. While the protocol suite can be complex at times, there has been a concerted effort for simplification. It contains the necessary components for negotiating the cryptographic details required for dynamically establishing a secure communication channel. While the protocol suite has continually been improved since its inception, a larger issue with the suite is ensuring implementations of IPsec stay current. Anecdotally, it seems that there was very little secondary sources for IPsec-v3 and IKEv2 compared to previous versions which does little to add clarity to the protocols. Even so, given enough time with the RFC standards, the protocols can be parsed to a sufficient level of understanding for most people to properly deploy a secured IPsec communication channel.

6 References

[1] Anderson, R. *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley Publishing, Indiana, 2008.

[2] Atkinson, R. IP Authentication Header. RFC 1826, RFC Editor, August 1995.

[3] Cisco. Ikev2 packet exchange and protocol level debugging, 2013.

[4] Ferguson, N., and Schneier, B. A Cryptographic Evaluation of IPsec. Tech. rep., Counterpane Internet Security Inc, 1999.

[5] Frankel, S., and Krishnan, S. IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap. RFC 6071, RFC Editor, February 2011.

[6] Harkins, D., and Carrel, D. The Internet Key Exchange (IKE). RFC 2409, RFC Editor, November 1998.

[7] Kaufman, C., et al. Internet Key Exchange Protocol Version 2 (IKEv2). RFC 7296, RFC Editor, October 2014.

[8] Kent, S. IP Authentication Header. RFC 4302, RFC Editor, December 2005.

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

[10] Kent, S., and Seo, K. Security Architecture for the Internet Protocol. RFC 4301, RFC Editor, December 2005.

[11] Nir, Y., et al. Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 8247, RFC Editor, September 2017.

[12] Pfleeger, C. P., et al. *Security in Computing*. Prentice Hall, Massachusetts, 2015.

[13] Shaheen, S., et al. Comparative analysis of internet key exchange protocols. *Communications in Computer and Information Science 721*, 1 (2017), 448–453.

[14] Thayer, R., et al. IP Security Document Roadmap. RFC 2411, RFC Editor, November 1998.

[15] Wouters, P., et al. Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH). RFC 8221, RFC Editor, October 2017.