Draft NIST Special Publication 800-77 Revision 1

Guide to IPsec VPNs

Elaine Barker
Quynh Dang
Sheila Frankel
Karen Scarfone
Paul Wouters

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.800-77r1-draft

COMPUTER SECURITY

National Institute of Standards and Technology
U.S. Department of Commerce

Draft NIST Special Publication 800-77 Revision 1

Guide to IPsec VPNs

Elaine Barker
Quynh Dang
Sheila Frankel
Computer Security Division
Information Technology Laboratory

Karen Scarfone
Scarfone Cybersecurity
Clifton, VA

Paul Wouters
No Hats Corporation
Toronto, ON, Canada

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.800-77r1-draft

July 2019

U.S. Department of Commerce
Wilbur L. Ross, Jr., Secretary

National Institute of Standards and Technology
Walter Copan, NIST Director and Under Secretary of Commerce for Standards and Technology

Authority

This publication has been developed by NIST in accordance with its statutory responsibilities under the Federal Information Security Modernization Act (FISMA) of 2014, 44 U.S.C. § 3551 *et seq.*, Public Law (P.L.) 113-283. NIST is responsible for developing information security standards and guidelines, including minimum requirements for federal information systems, but such standards and guidelines shall not apply to national security systems without the express approval of appropriate federal officials exercising policy authority over such systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

National Institute of Standards and Technology Special Publication 800-77 Revision 1
Natl. Inst. Stand. Technol. Spec. Publ. 800-77 Rev. 1, 161 pages (July 2019)
CODEN: NSPUE2

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.800-77r1-draft

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

There may be references in this publication to other publications currently under development by NIST in accordance with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies, may be used by federal agencies even before the completion of such companion publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For planning and transition purposes, federal agencies may wish to closely follow the development of these new publications by NIST.

Organizations are encouraged to review all draft publications during public comment periods and provide feedback to NIST. Many NIST cybersecurity publications, other than the ones noted above, are available at https://csrc.nist.gov/publications.

Public comment period: *July 2, 2019* through *October 8, 2019*

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
Email: revision_of_SP800-77@nist.gov

All comments are subject to release under the Freedom of Information Act (FOIA).

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of other than national security-related information in federal information systems. The Special Publication 800-series reports on ITL's research, guidelines, and outreach efforts in information system security, and its collaborative activities with industry, government, and academic organizations.

Abstract

Internet Protocol Security (IPsec) is a widely used network layer security control for protecting communications. IPsec is a framework of open standards for ensuring private communications over Internet Protocol (IP) networks. IPsec configuration is usually performed using the Internet Key Exchange (IKE) protocol. This publication provides practical guidance to organizations on implementing security services based on IPsec so that they can mitigate the risks associated with transmitting sensitive information across networks. The document focuses on how IPsec provides network layer security services and how organizations can implement IPsec and IKE to provide security under different circumstances. It also describes alternatives to IPsec and discusses under what circumstances each alternative may be appropriate.

Keywords

communications security; Internet Key Exchange (IKE); Internet Protocol (IP); Internet Protocol Security (IPsec); network layer security; networking; virtual private network (VPN).

Acknowledgments

The authors, Elaine Barker, Quynh Dang, and Sheila Frankel of NIST, wish to thank everyone who has contributed to this revision of Special Publication (SP) 800-77, particularly Andrew Regenscheid, David Waltermire, and Lily Chen of NIST, and Dorothy E. Cooley and James Banoczi of the National Security Agency (NSA).

The authors acknowledge the following individuals (with their original affiliations) and organizations that assisted in the development of the original SP 800-77:

- The authors of the original version: Sheila Frankel of NIST, and Karen Kent, Ryan Lewkowski, Angela D. Orebaugh, Ronald W. Ritchey, and Steven R. Sharma of Booz Allen Hamilton
- Bill Burr, Tim Grance, Okhee Kim, Peter Mell, and Murugiah Souppaya of NIST
- Darren Hartman and Mark Zimmerman of ICSA Labs
- Paul Hoffman of the VPN Consortium
- Representatives from the Department of Energy, the Department of State, the Environmental Protection Agency, and the U.S. Nuclear Regulatory Commission

Audience

This document has been created for network architects, network administrators, security staff, technical support staff, and computer security program managers who are responsible for the technical aspects of preparing, operating, and securing networked infrastructures. The material in this document is technically oriented, and it is assumed that readers have at least a basic understanding of networking and network security.

Trademark Information

All names are registered trademarks or trademarks of their respective companies.

Call for Patent Claims

This public review includes a call for information on essential patent claims (claims whose use would be required for compliance with the guidance or requirements in this Information Technology Laboratory (ITL) draft publication). Such guidance and/or requirements may be directly stated in this ITL Publication or by reference to another publication. This call also includes disclosure, where known, of the existence of pending U.S. or foreign patent applications relating to this ITL draft publication and of any relevant unexpired U.S. or foreign patents.

ITL may require from the patent holder, or a party authorized to make assurances on its behalf, in written or electronic form, either:

a) assurance in the form of a general disclaimer to the effect that such party does not hold and does not currently intend holding any essential patent claim(s); or

b) assurance that a license to such essential patent claim(s) will be made available to applicants desiring to utilize the license for the purpose of complying with the guidance or requirements in this ITL draft publication either:

    i. under reasonable terms and conditions that are demonstrably free of any unfair discrimination; or
    ii. without compensation and under reasonable terms and conditions that are demonstrably free of any unfair discrimination.

Such assurance shall indicate that the patent holder (or third party authorized to make assurances on its behalf) will include in any documents transferring ownership of patents subject to the assurance, provisions sufficient to ensure that the commitments in the assurance are binding on the transferee, and that the transferee will similarly include appropriate provisions in the event of future transfers with the goal of binding each successor-in-interest.

The assurance shall also indicate that it is intended to be binding on successors-in-interest regardless of whether such provisions are included in the relevant transfer documents.

Such statements should be addressed to: revision_of_SP800-77@nist.gov.

Executive Summary

Internet Protocol Security (IPsec) is a suite of open standards for ensuring private communications over public networks. It is the most common network layer security control, typically used to encrypt IP traffic between hosts in a network and for creating a virtual private network (VPN). A VPN is a virtual network built on top of existing physical networks that provides a secure communications mechanism for data and control information transmitted between computers or networks. IPsec is also used as a component that provides the security for many other internet protocols. The User Datagram Protocol (UDP) usage guidelines [1] specify IPsec as one of the methods to secure UDP.

The Internet Key Exchange (IKE) protocol is most commonly used to establish IPsec-based VPNs. The terms IKE and IPsec are often used interchangeably, although that is not correct. In practice, the terms "IPsec VPN," "IKEv2 VPN," "Cisco IPsec," "IPsec XAUTH," and "L2TP/IPsec" all refer to IPsec-based VPN connections. Some examples of technologies and protocols that use IKE and/or IPsec are:

- 3rd Generation Partnership Project (3GPP) mobile phone telephony standard (Long-Term Evolution [LTE]/5th Generation [5G], Wireless Fidelity [WiFi] calling) [2], [3]
- Ethernet VPN (EVPN) and Virtual eXtensible Local Area Network (VXLAN) [4]
- Software-Defined Networking (SDN) and Software-Defined Wide Area Network (SDWAN)
- Segment Routing [5]
- Data Center Network Virtualization Overlay (NVO3) Networks [6]
- Generic Network Virtualization Encapsulation (GENEVE) [7]
- Smart Grid [8]
- Constrained Application Protocol (CoAP)
- Low-Power Wireless Personal Area Network (6LowPAN) [9]
- Routing protocol protection [10] such as Border Gateway Protocol (BGP)/BGP Monitoring Protocol (BMP) [11] and Open Shortest Path First (OSPFv3) [12]

VPNs protect communications carried over public networks such as the Internet as well as private networks such as fiber networks or Multi-Protocol Label Switching (MPLS) networks. A VPN can provide several types of data protection, including confidentiality, integrity, data origin authentication, replay protection, and access control. The primary VPN architectures are as follows:

- **Gateway-to-gateway.** This architecture protects communications between two specific networks, such as an organization's main office network and a branch office network, or two business partners' networks.

- **Remote access.** Also known as host-to-gateway, this architecture protects communications between one or more individual hosts and a specific network belonging to an organization. The remote access architecture is most often used to allow hosts on unsecured networks, such as traveling employees and telecommuters, to gain access to internal organizational services, such as the organization's email and Web servers.

- **Host-to-host.** A host-to-host architecture protects communication between two specific computers. It can be used when a small number of users need to use or administer a remote system that requires the use of inherently insecure protocols.
- **Mesh.** In a mesh architecture, many hosts within one or a few networks all establish individual VPNs with each other.

The guide provides an overview of the types of security controls that can provide protection for network communications that are widely used throughout the world. IP communications are composed of four layers that work together: application, transport, network, and data link. Security controls exist for network communications at each of the four layers. As data is prepared for transport, it is passed from the highest to the lowest layer, with each layer adding more information. Because of this, a security control at a higher layer cannot provide full protection for lower layers, because the lower layers add information to the communications after the higher layer security controls have been applied. The primary disadvantage of lower layer security controls is that they are less flexible and granular than higher layer controls. Accordingly, network layer controls have become widely used for securing communications because they provide a more balanced solution.

IPsec is a network layer security protocol with two main components:

- **Encapsulating Security Payload (ESP)** is the protocol that transports the encrypted and integrity-protected network communications across the network. If only integrity protection is needed without encryption, the ESP protocol can use NULL encryption. An older method for IPsec transport of non-encrypted data is to use the Authentication Header (AH) protocol, but this method is no longer recommended by this guidance.
- **Internet Key Exchange (IKE)** is the protocol used by IPsec to negotiate IPsec connection settings; authenticate endpoints to each other; define the security parameters of IPsec-protected connections; negotiate session keys; and manage, update, and delete IPsec-protected communication channels. The current version is IKEv2.

Optionally, IPsec can use the IP Payload Compression Protocol (IPComp) to compress packet payloads before encrypting them, but this has not been widely used.

Only implementations of NIST-approved cryptographic algorithms specified in Federal Information Processing Standards (FIPS) or NIST Special Publications (SPs) and contained in FIPS-validated cryptographic modules shall be used in IPsec VPN deployments for compliance with this guidance. The FIPS 140 [13] specification defines how cryptographic modules will be validated. One requirement of FIPS 140 is that the module be capable of operating in a mode where all algorithms are NIST approved. NIST-approved algorithms are specified in a FIPS (e.g., FIPS 180, *Secure Hash Standard*) or in a NIST Special Publication (e.g., SP 800-56A, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm*

Cryptography). Some implementations can run in both FIPS mode and non-FIPS mode, so it is important to set and verify the mode of operation of the IKE and IPsec modules.

The Cryptographic Module Validation Program (CMVP) is a joint effort between NIST and the Communications Security Establishment (CSE) of the Government of Canada for the validation of cryptographic modules against FIPS 140-2 [13]. The Cryptographic Algorithm Validation Program (CAVP) provides validation testing of FIPS-approved and NIST-recommended cryptographic algorithms and their individual components. Cryptographic algorithm validation is a prerequisite of cryptographic module validation.

Cryptographic recommendations in this document are based on the time of publication of this document and may be superseded by other publications in the future. Appendix F contains a list of relevant FIPS, SPs, and Internet Engineering Task Force (IETF) standards related to IKE and IPsec.

Approved algorithms and their options for IKE and IPsec as of this writing are listed in Table 1:

Table 1: Approved Algorithms and Options

Option	Recommended	Legacy	Expected
IKE			
Version	IKEv2	IKEv1	
IKEv2 exchanges	All	-	
IKEv1 exchanges	Main Mode, Quick Mode	Aggressive Mode	
Encryption	AES-GCM, AES-CTR, AES-CBC, AES-CCM (128, 192, 256-bit keys)	TDEA	
Integrity/Pseudo Random Function (PRF)	HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	HMAC-SHA-1	HMAC-SHA-3
Diffie-Hellman (DH) group	DH 14 to DH 21 RFC [64] and RFC 5114 [65]		DH 31 and DH 32, RFC 8031 [72]
Peer authentication	RSA, DSA, and ECDSA with 128-bit security strength (for example, RSA with 3072-bit or larger key)	RSA, DSA, and ECDSA with less than 112 bits of security strength	
Lifetime	24 hours		
IPsec			
Mode	tunnel mode, transport mode		
Protocol	ESP, IPComp	AH	
Version	IPsec-v3	IPsec-v2	
Encryption	AES-GCM, AES-CTR, AES-CBC, AES-CCM, (128, 192, 256-bit keys)		
Integrity	HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, AES-GMAC		HMAC-SHA-3
Perfect Forward Secrecy (PFS)	Same or stronger DH as initial IKE DH		

Option	Recommended	Legacy	Expected
Lifetime	8 hours		

Some of the cryptographic requirements will change at the end of 2020, see SP 800-131A [47] for details. Therefore, Federal agencies who want to provide IPsec VPN services after 2020 must ensure that their systems are upgradeable to the new NIST-approved algorithms and key lengths before the end of 2020, and that their IPsec VPN vendors guarantee that such upgrades will be available early enough for testing and deployment in the field.

The strongest possible cryptographic algorithms and key lengths that are NIST-approved should be used for authentication, encryption, and integrity protection unless they are incompatible with interoperability, performance, and export constraints.

In addition to providing specific recommendations related to configuring cryptography for IPsec, this guide presents a phased approach to IPsec planning and implementation that can help in achieving successful IPsec deployments. The five phases of the approach are as follows:

1. **Identify Needs**—Identify the need to protect network communications and determine how that need can best be met.
2. **Design the Solution**—Make design decisions in four areas: architectural considerations, authentication methods, cryptography policy, and packet filters. The placement of an IPsec gateway has potential security, functionality, and performance implications. An authentication solution should be selected based primarily on maintenance, scalability, and security. Packet filters should apply appropriate protections to traffic and not protect other types of traffic for performance or functionality reasons.
3. **Implement and Test a Prototype**—Test a prototype of the designed solution in a lab or test environment to identify any potential issues. Testing should evaluate several factors, including connectivity, protection, authentication, application compatibility, management, logging, performance, the security of the implementation, and component interoperability.
4. **Deploy the Solution**—Gradually deploy IPsec throughout the enterprise. Existing network infrastructure, applications, and users should be moved incrementally over time to the new IPsec solution. This provides administrators an opportunity to evaluate the impact of the IPsec solution and resolve issues prior to enterprise-wide deployment.
5. **Manage the Solution**—Maintain the IPsec components and resolve operational issues; repeat the planning and implementation process when significant changes need to be incorporated into the solution.

As part of implementing IPsec, organizations should also implement additional technical, operational, and management controls that support and complement IPsec implementations. Examples include establishing control over all entry and exit points for the protected networks, ensuring the security of all IPsec endpoints, and incorporating IPsec considerations into organizational policies.

Table of Contents

List of Appendices

List of Figures

List of Tables

1 Introduction

1.1 Purpose and Scope

This publication seeks to assist organizations in mitigating the risks associated with the transmission of sensitive information across networks by providing practical guidance on implementing security services based on Internet Protocol Security (IPsec). This document presents information that is independent of particular hardware platforms, operating systems, and applications, other than providing real-world examples to illustrate particular concepts. Specifically, the document includes a discussion of the need for network layer security services, then focuses on how IPsec provides them and how organizations can implement IPsec. The document uses a case-based approach to show how IPsec can be used to provide security for different scenarios. It also describes alternatives to IPsec and discusses the circumstances under which each alternative may be appropriate.

1.2 Document Structure

The remainder of this document is organized into the following sections and appendices:

- Section 2 discusses the need for network layer security, introduces the concept of virtual private networking (VPN), and defines the primary VPN architectures for IPsec.
- Section 3 explains the Internet Key Exchange (IKE) protocol.
- Section 4 covers the fundamentals of IPsec protocols, focusing on Encapsulating Security Payload (ESP).
- Section 5 describes the interactions between the IKE and IPsec subsystems.
- Section 6 provides information on troubleshooting common situations with IPsec VPNs.
- Section 7 points out issues to be considered during IPsec planning and implementation.
- Section 8 discusses several alternatives to IPsec and describes when each method may be appropriate.
- Section 9 presents several IPsec planning and implementation case studies that show how IPsec could be used in various scenarios.
- Section 10 briefly discusses future directions for IPsec.
- Appendix A defines the required configuration parameters for IKE and IPsec.
- Appendix B discusses the needs for IPsec-related policy and provides examples of common IPsec policy considerations.
- Appendix C contains configuration files referenced by the case studies in Section 9.
- Appendices D and E contain a glossary and acronym list, respectively.
- Appendix F lists the references.

2 Network Layer Security

This section provides a general introduction to *network layer security*—protecting network communications at the layer that is responsible for routing packets across networks. It first introduces the Internet Protocol (IP) model and its layers, then discusses the need to use security controls at each layer to protect communications. It provides a brief introduction to IPsec, primarily focused on the types of protection IPsec can provide for communications. This section also provides a brief introduction to VPN services and explains what types of protection a VPN can provide. It introduces different VPN architectures and discusses the features and common uses of each one.[1]

2.1 The Need for Network Layer Security

IP networking (sometimes called TCP/IP, although it encompasses more than just TCP, the Transmission Control Protocol) is the standard used throughout the world to provide network communications. IP communications are roughly composed of four layers that work together. When a user wants to transfer data across networks, the data is passed from the highest layer through intermediate layers to the lowest layer, with each layer adding additional information.[2] The lowest layer sends the accumulated data through the physical network; the data is then passed up through the layers to its destination. Essentially, the data produced by a layer is encapsulated in a larger container by the layer below it. The four IP layers, from highest to lowest, are shown in Figure 1.

Application Layer. This layer sends and receives data for particular applications, such as Domain Name System (DNS), web traffic via Hypertext Transfer Protocol (HTTP) and HTTP Secure (HTTPS), and email via Simple Mail Transfer Protocol (SMTP) and the Internet Message Access Protocol (IMAP).
Transport Layer. This layer provides connection-oriented or connectionless services for transporting application layer services between networks. The transport layer can optionally assure the reliability of communications. The Transmission Control Protocol (TCP), which provides reliable connection-oriented communications, and the User Datagram Protocol (UDP), which provides unreliable connectionless communications, are commonly used transport layer protocols.

[1] This document discusses only the most common VPN scenarios and uses of IPsec.

[2] At each layer, the logical units are typically composed of a header and a payload. The payload consists of the information passed down from the previous layer, while the header contains layer-specific information such as addresses. At the application layer, the payload is the actual application data.

Network Layer. This layer routes packets across networks. The Internet Protocol (IP) is the fundamental network layer protocol for TCP/IP. Other commonly used protocols at the network layer are the Internet Control Message Protocol (ICMP) and the Internet Group Management Protocol (IGMP).
Data Link Layer. This layer handles communications between the physical network components. The best-known data link layer protocols are Ethernet and the various WiFi standards such as the Institute of Electrical and Electronics Engineers (IEEE) 802.11.

Figure 1: IP Model

Security controls exist for network communications at each layer of the IP model. As previously explained, data is passed from the highest to the lowest layer, with each layer adding more information. Because of this, a security control at a higher layer cannot provide full protection for lower layers, because the lower layers perform functions of which the higher layers are not aware. The following items discuss the security controls that are available at each layer:

- **Application Layer.** Separate controls must be established for each application. For example, if an application needs to protect sensitive data sent across networks, the application may need to be modified to provide this protection. While this provides a high degree of control and flexibility over the application’s security, it may require a large resource investment to add and configure controls properly for each application.

  Designing a cryptographically sound application protocol is very difficult, and implementing it properly is even more challenging, so creating new application layer security controls is likely to create vulnerabilities. Also, some applications, particularly commercial off-the-shelf (COTS) software, may not be capable of providing such protection.

  While application layer controls can protect application data, they cannot protect communication metadata, such as source and destination IP addresses, because this information exists at a lower layer. Whenever possible, application layer controls for protecting network communications should be standards-based solutions that have been in use for some time. One example is Secure/Multipurpose Internet Mail Extensions (S/MIME) [14], which is commonly used to encrypt email messages. Another example is the Secure Shell (SSH) [15] protocol that encrypts remote login sessions.

- **Transport Layer.** Controls at this layer can be used to protect the data in a single communication session between two hosts, often called a *netflow*. Because IP information is added at the network layer, transport layer controls cannot protect it. In the past there have been many protocols that protect different netflows, but the current best practice is to use Transport Layer Security (TLS) [16] to protect TCP streams, and Datagram Transport Layer Security (DTLS) [17] to protect UDP datagrams.

The use of DTLS or TLS typically requires each application to support DTLS or TLS; however, unlike application layer controls, which typically involve extensive customization of the application, transport layer controls such as DTLS and TLS are less intrusive because they simply protect network communications and do not need to understand the application's functions or characteristics. Although using DTLS or TLS may require modifying some applications, these protocols are well-tested and are a relatively low-risk option compared to adding protection at the application layer instead.

Alternatively, an application could use a TLS proxy instead of building native support for DTLS or TLS. The transport layer can only provide transport security, not data origin security. For example, a TLS-based connection between two email servers protects the transport from eavesdroppers but does not protect the message content transmitted within that TLS connection from manipulation by one of the two email servers. DTLS and TLS are sometimes deployed as a generic VPN solution protecting all IP traffic instead of only protecting a netflow. Such VPNs, commonly called SSL-based VPNs, work on the network layer but use an application at the transport layer.

- **Network Layer.** Controls at this layer apply to all applications and are not application-specific. For example, all network communications between two hosts or networks can be protected at this layer without modifying any applications on the clients or the servers. In many environments, network layer controls such as IPsec provide a much better solution than transport or application layer controls because of the difficulties in adding controls to individual applications. Network layer controls also provide a way for network administrators to enforce certain security policies.

  Another advantage of network layer controls is that since IP information (e.g., IP addresses) is added at this layer, the controls can protect both the data within the packets and the IP information for each packet. However, network layer controls provide less control and flexibility for protecting specific applications than transport and application layer controls.

- **Data Link Layer.** Data link layer controls are applied to all communications on a specific physical link, such as a dedicated circuit between two buildings or a WiFi network. Data link layer controls for dedicated circuits are most often provided by specialized hardware devices known as *data link encryptors*; data link layer controls for WiFi networks are usually provided through WiFi chipset firmware. Because the data link layer is below the network layer, controls at this layer can protect both data and IP information.

  Compared to controls at the other layers, data link layer controls are relatively simple, which makes them easier to implement; also, they support other network layer protocols besides IP. Because data link layer controls are specific to a particular physical link or local WiFi signal, they are poorly suited to protecting connections to remote endpoints, such as establishing a VPN over the Internet.

An Internet-based connection is typically composed of several physical links chained together; protecting such a connection with data link layer controls would involve many parties and different protocols for each part of the physical chain. It is easier to consider the internet as a whole to be untrustworthy and use controls at the network, transport, or application layer. Data link layer protocols have been used for many years primarily to provide additional protection for specific physical links that should not be trusted.

Because network layer security controls can provide protection for many applications at once without modifying them, these controls have been used frequently for securing communications, particularly over shared networks such as the Internet. Network layer security controls provide a single solution for protecting all data from all applications, as well as protecting IP address, protocol, and port information. However, in many cases, controls at another layer are better suited to providing protection than network layer controls. For example, if only one or two applications need protection, a network layer control may be overkill. An application is often not aware of the (lack of) protection offered by the network or data link layer. Controls at each layer offer advantages and features that controls at other layers do not. Information on data link, transport, and application layer alternatives to network layer controls is provided in Section 8.

2.2 The IPsec Protocol

IPsec has emerged as the most commonly used network layer security control for protecting communications. IPsec is a framework of open standards for ensuring private communications over IP networks. The Internet Key Exchange (IKE) protocol is used to securely negotiate IPsec parameters and encryption keys. IKE is described in Section 3.

The IPsec Working Group at the Internet Engineering Task Force (IETF) is responsible for maintaining and publishing the standards for IKE and IPsec. Documents produced by IETF Working Groups are defined in two types of documents: Request for Comment (RFC), which are completed specifications; and Internet-Drafts, which are working documents that may become RFCs. IKEv2 is specified in [18]. The Encapsulating Security Protocol (ESP), the core IPsec security protocol, is specified in [19]. Algorithm implementation and usage guidelines are specified in [20] for IKEv2 and in [21] for IPsec. Various extensions to IKEv2 have their own RFC specifications. The IKE and IPsec protocols originated at the IETF almost three decades ago. Some of their history, such as the difference between IPsec-v2 and IPsec-v3, has been documented in the IPsec roadmap document [22].

Depending on how IPsec is implemented and configured, it can provide any combination of the following types of protection:

- **Confidentiality**. IPsec ensures that data cannot be read by unauthorized parties. This is accomplished by encrypting and decrypting data using a cryptographic algorithm and a secret key—a value known only to the two parties exchanging data. The data can only be decrypted by someone who has the secret key. While it is possible to use IPsec without encryption, it is not recommended.

- **Integrity**. IPsec determines if data has been changed (intentionally or unintentionally) during transit. The integrity of data can be assured by generating a message authentication code (MAC) value, which is a cryptographic checksum (hash) of the data made with a mutually agreed secret key (different from the encryption secret key). If the data is altered and the MAC's verification will fail.
- **Confidentiality and Integrity.** Both types of checks can be combined into one Authenticated Encryption with Associated Data (AEAD) algorithm. This combines symmetric encryption and cryptographic checksums into one process. Both parties still need to have the same secret key and additional data.
- **Peer Authentication.** Each IPsec endpoint confirms the identity of the other IPsec endpoint with which it wishes to communicate, ensuring that the network traffic and data is only transmitted to the expected and authorized endpoint.
- **Replay Protection.** The same data will not be accepted multiple times, and data is not accepted grossly out of order. This prevents attackers from copying and retransmitting valid IPsec encrypted data for malicious purposes. IPsec (like UDP) does not ensure that data is delivered in the exact order in which it was sent. The receiver has a Replay Window where it will store out of order received messages before decrypting and delivering these messages to the operating system in the right order.
- **Traffic Analysis Protection.** When IPsec's tunnel mode is used (see Section 4.1.1), a person monitoring network traffic does not know which parties are communicating, how often communications are occurring, or how much data is being exchanged. While the number and size of the encrypted packets being exchanged can be counted, the traffic flow confidentiality (TFC) capabilities of ESP can pad all packets to a single length (usually the maximum transmission unit [MTU]), and dummy packets can be sent to further obfuscate the timing of the actual communication.
- **Access Control.** IPsec endpoints can perform filtering to ensure that only authorized IPsec users can access particular network resources. IPsec endpoints can also allow or block certain types of network traffic, such as allowing Web server access but denying file sharing. This is called *policy-based IPsec*. *Routing-based IPsec* accepts all traffic at the IPsec policy layer, but both endpoints filter valid traffic by setting routes into a specific IPsec interface. In other words, the routing table acts as the policy filter.

  Policy-based IPsec is more secure than routing-based IPsec, as the security of the policy works independently from the security of the remote endpoint. Policy-based IPsec is not vulnerable to accidental or malicious routing table changes, and it prevents leaking packets to the local network, since local packets do not use the routing table. IPsec-based access control works independently from other access control mechanisms, such as firewall services or other mandatory access control mechanisms.
- **Perfect Forward Secrecy (PFS).** IPsec endpoints create session keys that are changed frequently, typically once an hour. Afterwards, the endpoints wipe the old session keys from volatile memory, and no entities are left with a copy of these private decryption keys. Since expired keys are not saved, any encrypted traffic monitored and stored cannot

be decrypted at a later time by compromising an IPsec endpoint and obtaining the encryption/decryption keys belonging to past IPsec sessions.

Normally, new keys are generated based on the generated shared secret of the original key exchange using a key derivation function (KDF). To guarantee that new key material has no relationship to the old key exchange, fresh session keys can, optionally, be generated by performing a new Diffie-Hellman (DH) key exchange instead of reusing the old key exchange's generated shared secret to generate new session keys. This method of using a fresh key exchange provides *perfect forward secrecy (PFS)*.

- **Mobility.** The outer IP address of an endpoint can change without causing an interruption of the encrypted data flow. Since the application is communicating using the inner (encrypted) IP address, it does not matter that the outer IP address changes. This allows a device to switch from WiFi to Ethernet to mobile data without application interruption.

2.3 Virtual Private Networking (VPN)

The most common use of IPsec implementations is providing VPN services. A *VPN* is a virtual network, built on top of existing physical networks, that can provide a secure communications mechanism for data and IP information transmitted between networks or between different nodes on the same network. Because a VPN can be used over existing networks, such as the Internet, it can facilitate the secure transfer of sensitive data across public networks. This is often less expensive than alternatives such as dedicated private telecommunication links between organizations or branch offices. Since dedicated private communication lines are often multi-tenant solutions themselves, such as those partitioned via Multi-Protocol Label Switching (MPLS) [23] and run by third-party telecommunication companies, even those dedicated links are now usually protected by an IPsec VPN. Remote access VPNs provide flexible solutions, such as securing communications between remote workers and the organization's servers. A VPN can be established within a single network to protect particularly sensitive communications from other parties on the same network, or even deploy a mesh of IPsec connections between all nodes in a single network so that no unencrypted data ever appears on the network. Section 2.4 discusses these different deployment models.

Below are further discussions of the cryptographic security services provided by IPsec for VPNs.

2.3.1 Confidentiality

VPNs use symmetric cryptography to encrypt and decrypt their command and data channels. Symmetric cryptography is generally more efficient and requires less processing power than asymmetric cryptography, which is why symmetric encryption is typically used to encrypt the bulk of the data being sent over a VPN. NIST-approved algorithms that implement symmetric encryption include Advanced Encryption Standard (AES) and Triple Data Encryption Standard

(3DES)[3]. One of the NIST-approved symmetric encryption algorithms is AES-Galois Counter Mode (AES-GCM); see Table 1 for the other NIST-approved symmetric encryption algorithms.

2.3.2 Integrity

Integrity is provided by a message authentication algorithm. The algorithm takes input data and a secret integrity key and produces a message authentication code (MAC). The data and MAC are sent across the network. The receiver calculates the MAC on the received data using the same secret integrity key (which has been previously established between the sender and receiver). If there is any change in the message or/and its MAC, a verification of the MAC will fail, and the message can be discarded. Common algorithms that implement integrity protection are:

- The keyed-hash message authentication code (HMAC) algorithm specified in FIPS 198 [24], which uses a hash function from FIPS 180 [25] (i.e., Secure Hash Algorithm (SHA): SHA-1 or the SHA-2 family of hash functions)[4]
- A mode of AES, as specified in FIPS 197 [26]. Included modes are AES-Cipher Block Chaining (AES-XCBC),[5] AES-Cipher-Based Message Authentication Code (AES-CMAC) [27], and AES-Galois Message Authentication Code (AES-GMAC) [28]

2.3.3 Establishment of Shared Secret Keys

VPNs typically use the DH key exchange algorithm to create a confidential communication channel to calculate a shared key between the two endpoints that an eavesdropper cannot obtain or compute. DH key exchanges can be based on finite field cryptography ("classic" or "modular" DH) or on elliptic curve (ECDH). After performing the DH key exchange and calculating the shared key, the endpoints still need to authenticate to each other to ensure that the confidential communication channel is set up with the expected party, and not somebody else.

2.3.4 Peer Authentication

A digital signature algorithm is used for peer authentication. It uses two separate keys: a public key and a private key. The private key is used to digitally sign the data, and the public key is used to verify the digital signature. These keys are often referred to as *public/private key pairs*. When an individual's private key is used to digitally sign data, only that same individual's corresponding public key can be used to verify the digital signature. Common algorithms that are used to generate and verify digital signatures include RSA, the Digital Signature Algorithm (DSA), and the Elliptic Curve Digital Signature Algorithm (ECDSA).[6] NIST-approved digital signature algorithms are specified in [29].

---

[3] Triple DES is deprecated and is expected to be disallowed in the near future.

[4] The term HMAC-SHA-2 is used to describe three members of the HMAC-SHA-2 family, HMAC-SHA256, HMAC-SHA384 and HMAC-SHA512

[5] While commonly deployed on Internet of Things (IoT) devices, AES-XCBC is not a NIST-approved integrity algorithm.

[6] NIST-approved algorithms must also be used for digital signatures. See https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program for information on such algorithms.

VPNs usually use asymmetric cryptography for identity authentication. This can be in the form of raw public/private key pair or X.509 certificate-based public/private key pair. A VPN entity is authenticated by proving it has possession of the private key of a known public/private key pair as well as the secret key computed by the parties during the DH key exchange. This binds the private communication channel (i.e., the VPN) to the expected identities. The public key can verify this proof without having a copy of the private key. Thus, as long as both parties each have the other's public key and their own private key, they can establish an authenticated private channel through which they can communicate.

A less secure method of identity authentication is using a preshared key (PSK). Parties authenticate each other's identity based on the fact that no one else has possession of this shared key, which must be established out-of-band.[7] A VPN entity's identity is authenticated by proving that it has possession of the PSK as well as the secret key computed by the parties during the DH key exchange. This binds the private communication channel to the expected identities. The main disadvantage of VPNs using PSKs for authentication is that all parties that know the PSK can impersonate every other party in the group. PSKs are also vulnerable to online and offline dictionary attacks. That means that PSKs must be highly random (providing at least 112 bits of security strength) and must not be based on simple words or phrases, otherwise an attacker observing the key exchange can attempt to use an offline brute force attack to find the PSK by calculating the authentication payload based on dictionary words and comparing the generated authentication payloads to the observed authentication payload. Unfortunately, experience has shown that administrators often use weak PSKs that are vulnerable to dictionary attacks.

2.3.5 Deployment Risks

VPNs do not remove all risk from networking, particularly for communications that occur over public networks. One potential problem is the strength of the implementation. For example, flaws in an encryption algorithm or the software implementing the algorithm could allow attackers to decrypt intercepted traffic, and random number generators that do not produce sufficiently random values could provide additional attack possibilities. Another issue is encryption key disclosure; an attacker who discovers a symmetric key could decrypt previously recorded or current traffic. An attacker obtaining the private key of a public/private key pair (or PSK) used for identity authentication could potentially pose as a legitimate user.

Another area of risk involves availability. A common model for information assurance is based on the concepts of confidentiality, integrity, and availability. Although VPNs are designed to support confidentiality and integrity, they generally do not improve availability, the ability for authorized users to access systems as needed. In fact, many VPN implementations actually tend to decrease availability somewhat because they add more components, complexity, and services to the existing network infrastructure.

---

[7] *Out-of-band* refers to using a separate communications mechanism to transfer information. For example, the VPN cannot be used to exchange the keys securely because the keys are required to provide the necessary protection.

Risks are highly dependent upon the chosen VPN architecture and the details of the implementation. Section 2.4 describes the primary VPN architectures.

2.4 Primary IPsec-Based VPN Architectures

There are four primary architectures for IPsec-based VPNs:

- Gateway-to-gateway
- Remote access
- Host-to-host
- Mesh

2.4.1 Gateway-to-Gateway

IPsec-based VPNs are often used to provide secure network communications between two networks. This is typically done by deploying a VPN gateway onto each network and establishing a VPN connection between the two gateways. Traffic between the two networks that needs to be secured passes within the established VPN connection between the two VPN gateways. The VPN gateway may be a dedicated device that only performs VPN functions, or it may be part of another network device, such as a firewall or router. Figure 2 shows an example of an IPsec network architecture that uses the gateway-to-gateway model to provide a protected connection between the two networks.

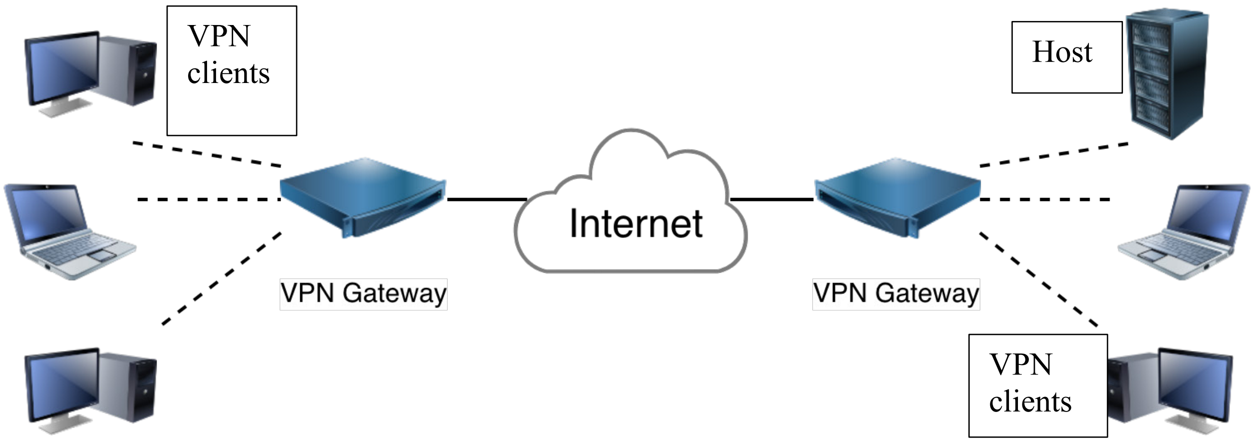

Figure 2: Gateway-to-Gateway VPN Architecture Example

This model is relatively simple to understand. To facilitate VPN connections, one of the VPN gateways issues a request to the other to establish an IPsec connection. The two VPN gateways exchange information with each other and create an IPsec connection. Routing on each network is configured so that as hosts on one network need to communicate with hosts on the other network, their network traffic is automatically routed through the IPsec connection, protecting it appropriately. A single IPsec connection establishing a tunnel between the gateways can support all communications between the two networks, or multiple IPsec connections can each protect different types or classes of traffic. The gateways connect to each other using IPv4 or IPv6 protocols. When using tunnel mode, the IP address family of the outer ESP packets transmitted between the gateways does not need to be the same as the IP address family of the encrypted IP packets. For example, an IPsec connection between the hosts on IPv6 addresses 2001:db8:1:2::45 and 2001:db8:1:2::23 could be used to transport IPv4 traffic from 192.0.2.0/24 to 198.51.100.0/24. These types of IPsec connections are often called 6in4 or 4in6 to denote the inner and outer IP families.

Figure 2 illustrates a gateway-to-gateway VPN that does not provide full protection for data throughout its transit. In fact, the gateway-to-gateway architecture only protects data between the two gateways, as denoted by the solid line. The dashed lines indicate that communications between VPN clients and their local gateway, and between the remote gateway and destination hosts (e.g., servers) are not protected by the gateway-to-gateway architecture. The other VPN models provide protection for more of the transit path. The gateway-to-gateway architecture is most often used when connecting two secured networks, such as linking a branch office to headquarters over the Internet. The gateway-to-gateway architecture is the easiest to implement in terms of user and host management. Gateway-to-gateway VPNs are typically transparent to users; the use of a gateway-to-gateway VPN connection is not noticeable to them. Also, the users' systems and the target hosts (e.g., servers) do not need to have any VPN client software installed, nor should they require any reconfiguration, to be able to use the VPN.

If the gateway-to-gateway VPN connects two different organizations, it is possible that some special DNS configuration is required if machines in one network need to be able to reach machines in the other network by DNS name. If machines are found by their IP address, no special DNS handling is required.

2.4.2 Remote Access

An increasingly common VPN architecture is the remote access architecture. The organization deploys a VPN gateway onto its network; each remote access user then establishes a VPN connection between their device (host) and the VPN gateway. As with the gateway-to-gateway architecture, the VPN gateway may be a dedicated device or part of another network device. Figure 3 shows an example of an IPsec remote access architecture that provides a protected connection for the remote user.

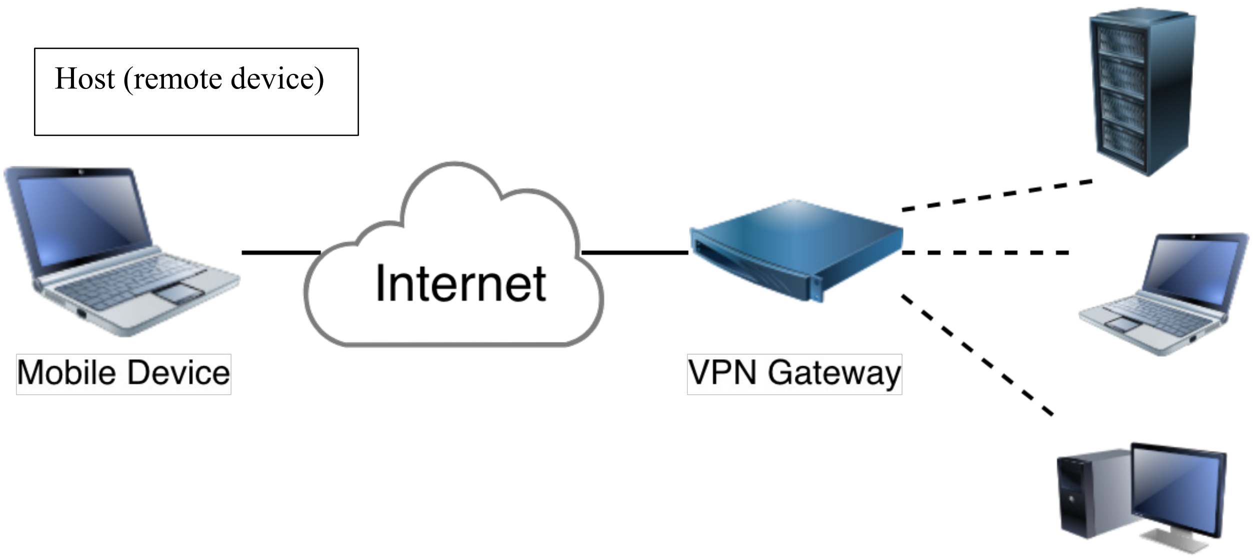

Figure 3: Remote Access VPN Architecture Example

In this model, IPsec connections are created as needed for each individual mobile device, which have been configured to act as IPsec clients with the organization's IPsec gateway. When a remote user wishes to use computing resources through the VPN, the host initiates communications with the VPN gateway. The user is typically asked by the VPN gateway to authenticate his identity before the connection can be established. The VPN gateway can perform the authentication itself or consult a dedicated authentication server. The client (the remote device in Figure 3) and gateway exchange information, and the IPsec connection is established. The user can now use the organization's computing resources, and the network traffic between the user's host (the remote device in Figure 3) and the VPN gateway will be protected by the IPsec connection.

Some organizations do not want to receive all the internet traffic generated by a remote host. If that host is browsing the internet, that traffic will not go through the VPN connection. Only traffic for the organization itself will be sent over the VPN connection. This is called a *split-tunnel VPN*. Other organizations do not trust the remote hosts to directly communicate with the internet while being connected via a VPN connection to the organizational computer resources, since that Internet connection could be used to attack or infiltrate the VPN connection. If an organization normally has a strict firewall preventing unauthorized access by the hosts in the local network, it would not want a remote host to bypass this security when it is connecting from a remote location. In that case, a remote host will send all its traffic via the VPN connection to the VPN gateway; this allows IPsec protection to be applied to this traffic as well. Traffic received and decrypted by the VPN gateway that is not meant for the local organization can be sent further to the organization's firewall for inspection, and then sent onwards through the organization's internet connection. Reply traffic similarly will flow back via the organization's firewall to the VPN gateway and will then be sent via the VPN connection to the remote host.

As shown in Figure 3, the remote access VPN does not provide full protection for data throughout its transit. The dashed lines indicate that communications between the gateway and the destination hosts (e.g., servers) on the right side of the figure are not protected. The remote access VPN architecture is most often used when connecting hosts on unsecured networks to resources on secured networks, such as linking traveling employees around the world to headquarters over the Internet. The remote access VPN is somewhat complex to implement and maintain in terms of user and host management (the VPN gateway (or a designated device) must manage credentials of all of the remote machines (hosts) and their authorized users and all of these might change often.) Remote access VPNs are typically not transparent to users because they must authenticate before using the VPN. Also, the user's device needs to have a VPN connection configured. Some devices do not allow more than one VPN connection to be active at a time.

Remote access users can find themselves on networks that, intentionally or not, cause VPN connections to fail. Some unintentional failures can be worked around by always having the latest software and IPsec VPN features supported.[8] Standard IKE runs over the UDP protocol, and ESP can also use UDP. Some networks block all UDP packets, causing IKE and ESP-over-UDP traffic to be dropped. As a method of last resort, IPsec communication can be tunneled over TCP, which is a more universally accepted protocol. For added insurance, TLS can be used in conjunction with TCP to work around network failures with native IPsec packets.

Modern devices often have more than one network interface, and the user can switch between different network interfaces automatically. For instance, when a mobile device loses a WiFi connection, it can automatically fall back to a mobile network (LTE/5G) provider. IPsec provides mobility support to ensure that the VPN connection keeps working without interruption when switching between such networks.

2.4.3 Host-to-Host

The host-to-host VPN architecture is used for a variety of reasons. For security reasons, some hosts may only accept connections protected by a VPN. This makes it more secure against unauthenticated access attempts. For example, if the web server software on the host is

[8] A common unintentional breaking of IPsec happens when a network does not handle IP fragmentation correctly. This can cause the setup of the IPsec connection to fail. Modern implementations of IPsec support their own IKE fragmentation that ensures the network layer never needs to fragment IKE packets.

vulnerable to a specific attack, it is only exposed to those who also have VPN credentials to contact the host. Another common issue is the presence of attackers performing port scans or dictionary attacks against the login method (for example, SSH). With a VPN, these ports are not accessible to attackers.

In this case, the organization configures the server to provide VPN services, and the system administrators' machines (or some users' machine) to act as VPN clients. The system administrators use the VPN client when needed to establish protected connections to the remote server. Figure 4 shows an example of an IPsec network architecture that uses the host-to-host architecture to provide a protected connection to a server for an administrator (or just a user).The point of a host-to-host VPN connection is that the traffic is protected all the way from one end to the other of the connection.

In this model, IPsec connections are created as needed for each individual VPN user. Users' hosts have been configured to act as IPsec clients with a remote host that is server. When a user wishes to use resources on the server, the user's host initiates IPsec communications with the server. The server acts as an IPsec server that requests the user to authenticate before the connection can be established. The user's host and the server exchange information, and if the authentication is successful, the IPsec connection is established. The user can now access the server, and the network traffic between the user's host and the server will be protected by the IPsec connection.

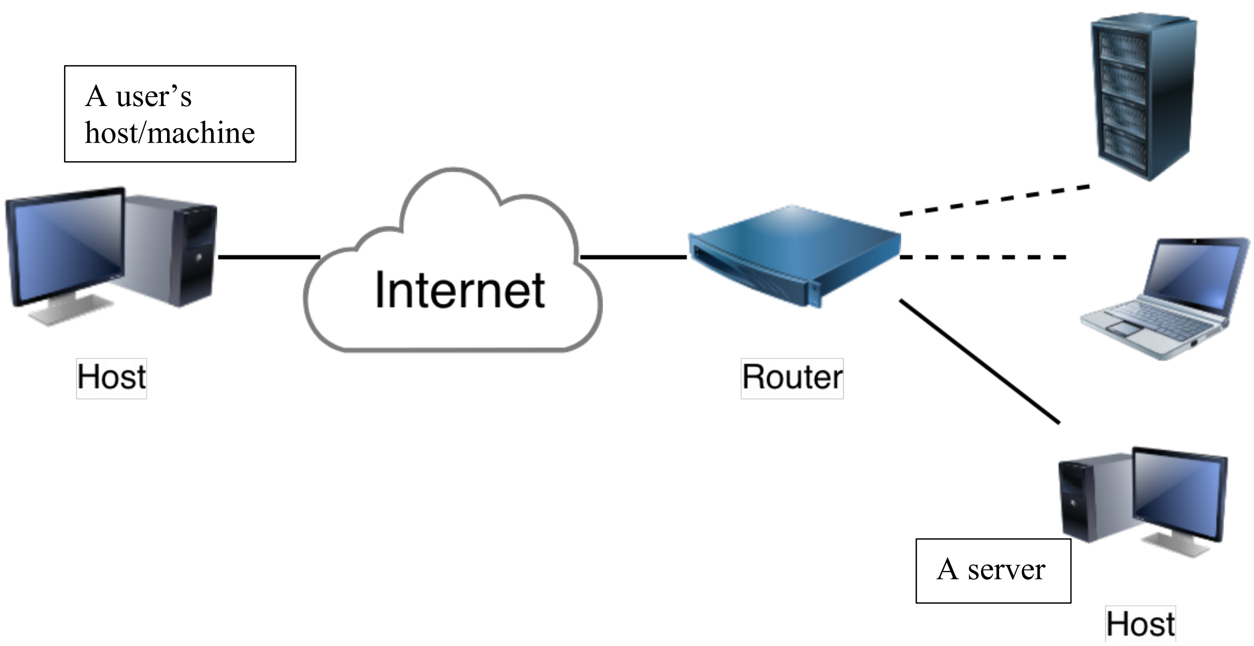

Figure 4: Host-to-Host VPN Architecture Example

As shown in Figure 4, the host-to-host VPN provides protection for data throughout its transit. This can be a problem because network-based firewalls, intrusion detection systems, and other network devices cannot be deployed to inspect the traffic in transit, which effectively circumvents certain layers of security.[9] The host-to-host VPN is most often used when a small number of trusted users need to use or administer a remote system that requires the use of insecure protocols (e.g., a legacy system) and which can be updated to provide VPN services.

Host-to-host VPNs can be resource-intensive to implement and maintain in terms of configuration management. Host-to-host VPNs are not transparent to users because they must authenticate the user before using the VPN. Also, all end user systems and servers that will participate in VPNs need to have VPN software installed and/or configured. However, the host-to-host architecture can be deployed in a more automated way that requires no end user interaction to establish a VPN.

A special case of host-to-host VPNs is a large-scale host-to-host IPsec deployment. This is typically used when one wants to encrypt all connections within a network, cloud, or datacenter. Whenever one node in such a network wishes to communicate with another node in the network, it first establishes an IPsec connection. This is also called *mesh encryption*. Usually, these IPsec connections are packet triggered. An application sends a packet to a remote host. The kernel of the host on which the application runs receives the packet from the application and determines that it does not have an IPsec connection to that remote host, so it triggers the setup of an IPsec connection. Once the IPsec connection is established, the packet is encrypted and sent to the remote host. This way, no unencrypted packet is ever sent over the network. Hosts authenticate each other using X.509 certificates or Domain Name System Security Extensions (DNSSEC). These types of authentication are based on a shared trust anchor, an X.509 certificate authority (CA) or a DNSSEC zone key. This allows hosts to be added to a network without the need to reconfigure all other hosts to learn about the newly deployed host.

One advantage of this type of IPsec architecture is that every host is responsible for its own protection; no large expensive IPsec gateways are required, which also means there is no single point of failure added to the network architecture. Hosts in a network can be configured to insist on IPsec, or to attempt IPsec but to allow cleartext communication if that fails. This architecture can be combined with the gateway-to-gateway architecture, where hosts within one network can initiate IPsec to hosts in the network, extending the network mesh encryption to both networks. The two networks are connected by a gateway-to-gateway architecture so the internet can still be used to connect these two networks, at the cost of packets being encrypted twice—once by the host-to-host deployment and once by the gateway-to-gateway deployment.

2.4.3.1 SDN-Based VPN Encryption

Software Defined Networking (SDN) is an architecture of dynamic cloud networking. An *SDN network* (sometimes called a *Software Defined Wide Area Network*, or *SDWAN*) is a network with a Security Controller and compute nodes. All the nodes (hosts) are configured by the Security Controller, usually via the Network Configuration Protocol (NETCONF) [30]. For nodes within a network, or for nodes between two different networks, the node consults its local

[9] Device placement can also be an issue in remote access and gateway-to-gateway architectures, but in those architectures, it is usually possible to move devices or deploy additional devices to inspect decrypted data. This is not possible with a host-to-host architecture.

Security Controller. If the nodes have enough resources to set up IPsec, the Security Controllers can relay the authentication and connection parameters to their respective nodes, and the two nodes can then negotiate the IPsec VPN connection.

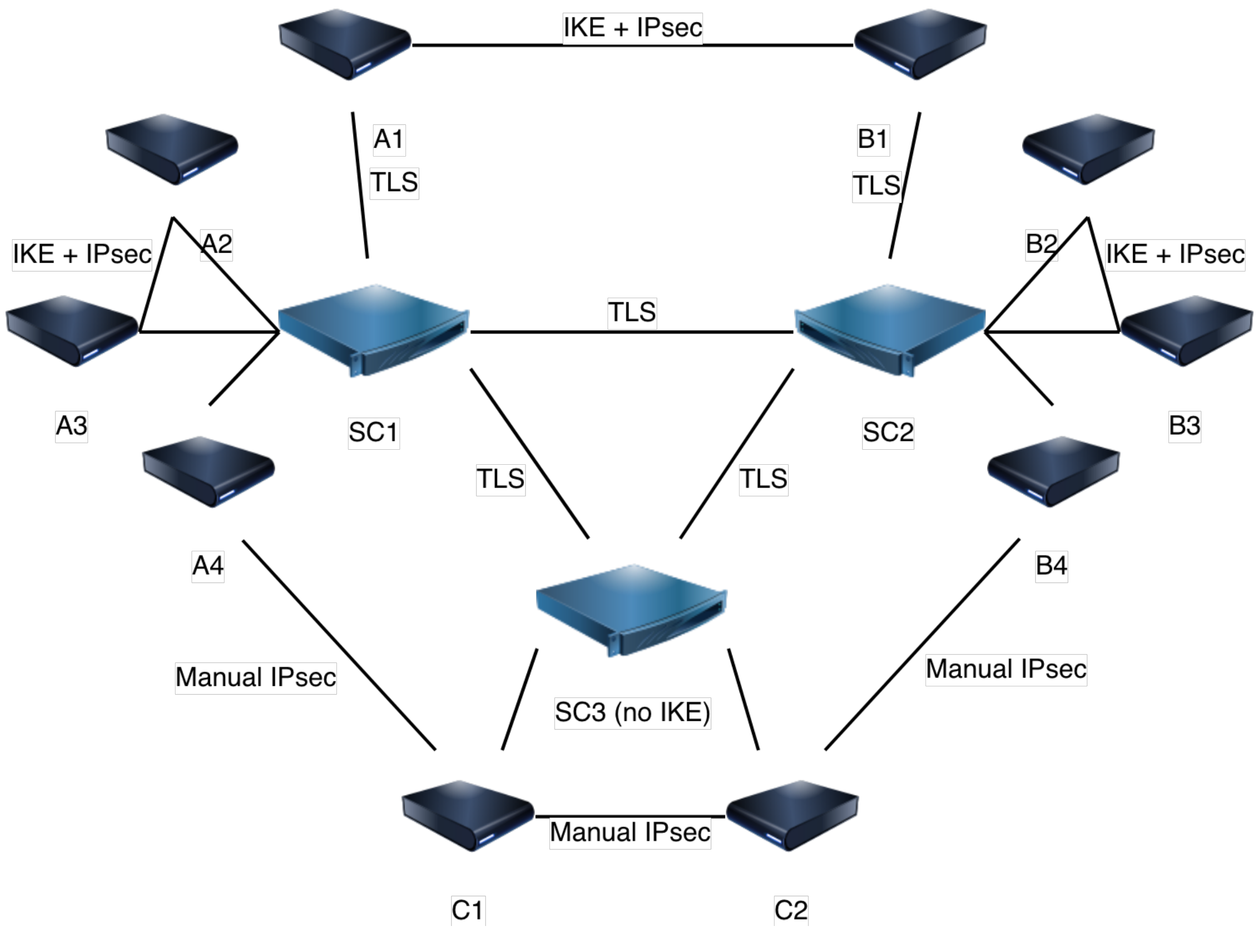

Figure 5: SDWAN Architecture Example

This is shown in Figure 5 for communication between the nodes A1 and B1 (at the top of the figure). Host A1 contacts its Security Controller SC1. SC1 and SC2 (host B1's Security Controller) negotiate the IKE and IPsec parameters and convey them to their respective hosts (A1 or B1, as appropriate). Host A1 can now initiate an IKE session with B1 and an IPsec connection is established between A1 and B1. The IPsec secret key material is only known by the A1 and B1 nodes and not by the Security Controller. The hosts could optionally transfer these secret keys to their Security Controller to facilitate monitoring via decryption by the Security Controller or another dedicated monitoring device that takes its configuration from the Security Controller.

If the hosts do not have enough resources to negotiate IPsec with many other nodes, each Security Controller can negotiate an IPsec connection on behalf of one of their hosts, and then give the keying material and security policies for the IPsec connection to that host. The two hosts receive the exact IPsec policies and the same encryption keys from their Security Controllers to install in their IPsec subsystems (key exchange is performed by the 2 corresponding Security Controllers). This latter method is called an *IKEless IPsec connection*. It is not the preferred method since, in this case, the Security Controllers are aware of all the secret keys used by their hosts, and the Security Controllers (or whoever manages to get control of one of them) can decrypt all the host-to-host IPsec protected traffic or masquerade as one of the hosts under its control.

A third method for configuring hosts by a Security Controller is for the hosts to give their key-exchange public keys to the Security Controller. When two devices establish an IPsec connection, the Security Controller distributes each device's key-exchange public key and a nonce to the other device. Each of the two devices uses the public and nonce from the other device along with own private key to generate a secret shared key which is then used for an IPsec connection. The Security Controller does not know the private keys or the shared key of the IPsec devices. Therefore, the Security Controllers cannot decrypt any host-to-host communication and cannot masquerade as one of the hosts.[10]

2.4.3.2 Anonymous IPsec VPN

The hardest part of rolling out an IPsec deployment is the authentication mechanisms, which depend on the prior deployment of a CA or other identity verifier. If a network only needs to protect itself against passive attackers—that is, attackers that can eavesdrop but not send their own malicious packets—then anonymous IPsec can be used. Therefore, anonymous IPsec connections are typically host-to-host connections and not gateway-based connections because an IPsec gateway typically requires authentication of the connecting host and authenticates itself to that host. A variant of this is server-only authenticated IPsec. This works similarly to regular HTTPS connections where a client connects to the server and the server has to authenticate itself to the client, but the client remains anonymous. Any client authentication then happens at the application layer, and not at the network layer.

The advantage of anonymous IPsec is that it can be rolled out quickly. Once in place and protecting against passive attackers, the configuration can be slowly migrated to an authenticated IPsec deployment that also protects against active attacks.

Due to its security risk, anonymous IPsec VPNs are discouraged by NIST.

2.5 Summary

Section 2 describes the IP model and its layers—application, transport, network, and data link—and explains how security controls at each layer provide different types of protection for IP communications. IPsec, a network layer security control, can provide several types of protection for data, depending on its configuration. The section describes VPNs and highlights the VPN architectures. IPsec is a framework of open standards for ensuring private communications over IP networks that is the standard used for network layer security control. It can provide several

[10] This is currently specified in an IETF draft document, draft-carrel-ipsecme-controller-ike [31].

types of protection, including maintaining confidentiality and integrity, preventing packet replay attacks and traffic analysis, and can incorporate access restrictions.

- IKE is the protocol that is used to negotiate, update, and maintain IPsec connections.
- A VPN is a virtual network built on top of existing networks that can provide a secure communications mechanism for data and IP information transmitted between networks.
- VPNs can be used to secure communication between individual hosts (host-to-host) or between multiple networks (gateway-to-gateway), or to provide secure remote access for mobile devices to a home or enterprise network. Hosts within a network can build a mesh of IPsec connections between all nodes or can use a Security Controller to assist them with building up VPN connections to other nodes.
- Although VPNs can reduce the risks of operating over an insecure network, they cannot eliminate it. For example, a VPN implementation may have flaws in algorithms or software that attackers can exploit. Also, VPN implementations often have at least a slightly negative impact on availability, because they add components and services to existing network infrastructures.

3 Internet Key Exchange (IKE)

When two hosts want to set up an IPsec connection with each other, they need to negotiate the parameters of the IPsec connection, such as the source and destination IP addresses that are allowed, the encryption algorithms to use, and the cryptographic key material to use for the encryption and decryption of packets. The hosts also need to authenticate each other. All of this is done using the Internet Key Exchange (IKE) protocol. The version of the IKE protocol described in this section is IKE version 2 (IKEv2) and is specified in RFC 7296[11] [18]. The differences between IKEv1 and IKEv2 are described at the end of this section.

Typically, IKE runs as a privileged process, while IPsec usually runs as part of the operating system kernel. The IKE process is responsible for configuring the kernel for IPsec. The kernel is responsible for the actual packet encryption and decryption operations. The IKE process can insert a policy into the kernel that will instruct the kernel to warn the IKE process when an unencrypted packet matching certain source and destination IP addresses and/or other criteria is about to be transmitted. If the peers can mutually authenticate each other, and agree on other policy details, then the IKE process can negotiate an IPsec tunnel that covers this packet. This is used for creating IPsec tunnels on demand.

3.1 Overview of IKE

The IKE protocol can be considered the command channel. The IPsec protocol is the data channel; it encrypts and decrypts the IP packets and verifies that the source and destination IP address conform to the negotiated policies. The IKE protocol command channel itself also needs to be encrypted to ensure the privacy of the parameters of the IPsec connection. In other words, first the IKE encrypted connection is established, and then one or more IPsec connections are established through the protected IKE command channel.[12] An IKE's connection establishment is called an *IKE Security Association* (IKE SA) [18].[13] An IPsec connection is called an *IPsec SA* or *Child SA*.[14] Both IKEv2 SAs and IPsec SAs are identified by their Security Parameters Index (SPI) numbers; for IKEv1, other fields are used as the SA identifier until the IPsec SPIs are established.

The IKE protocol consists of UDP messages on port 500 and 4500. As shown in Figure 6, each IKE packet consists of a fixed IKE header (the first five lines of the figure) followed by the variable-length IKE data.

---

11 The base protocol is defined in [18], but many IKE extensions have their own RFCs.
12 The IKEv2 protocol has been optimized to do some of this in parallel. As a result, the first IKE connection and the first IPsec connection are established at the same time.
13 An IKE SA is also called a Parent SA. In IKEv1, these were called ISAKMP SA or "Phase 1".
14 In IKEv1, these were called "Phase 2".

<table>
<tr><th>Byte 1</th><th colspan="2">Byte 2</th><th>Byte 3</th><th>Byte 4</th></tr>
<tr><td colspan="5">IKE SA Initiator’s SPI</td></tr>
<tr><td colspan="5">IKE SA Responder’s SPI</td></tr>
<tr><td>Next Payload</td><td>Major IKE Version</td><td>Minor IKE Version</td><td>Exchange Type</td><td>Flags</td></tr>
<tr><td colspan="5">Message ID</td></tr>
<tr><td colspan="5">Length of total message (IKE header plus data)</td></tr>
<tr><td colspan="5">IKE DATA</td></tr>
</table>

Figure 6: The IKEv2 Packet Format

The initiator of an IKE exchange generates a four-byte Initiator SPI. The responder generates the four-byte Responder SPI. In the first IKE packet sent by the Initiator, the Responder SPI is 0x00000000. The SPI numbers uniquely identify an established IKE SA. Each endpoint selects the IKE decryption key for an encrypted IKE message based on the SPI numbers.

An IKE session consists of IKE packet *exchanges*. Each exchange consists of a single request packet and a single reply packet. If there is any packet loss, it is the initiator’s responsibility to retransmit its request.[15] Each exchange packet has a message ID, which starts at zero and is incremented for each message exchange. The message ID allows detecting retransmitted packets and handling out-of-order IKE packets. There is a distinct message ID for messages started at each IKE peer.

The IKEv2 protocol uses two exchanges to establish an IKE SA and an associated IPsec SA. The IKE SA is then used to send and receive further configuration and management commands. The first exchange is called IKE_SA_INIT, and the second exchange is called IKE_AUTH. Together these two exchanges are referred to as the *initial exchanges*. Once these two exchanges are completed, both the initiator and the responder have established the IKE SA and one IPsec SA. Once the IKE SA is established, other additional exchange types are used to establish additional IPsec SAs, rekey the existing IKE SA or IPsec SAs, make configuration changes, perform a liveness detection of peers, and terminate IKE or IPsec SAs.

The following sections describe the IKE exchanges in detail and explain how they work together to establish IPsec connections.

3.2 IKE Exchange Types

The exchange type for additional IPsec SA messages is called CREATE_CHILD_SA. Another common exchange type is the INFORMATIONAL exchange, which is used for notification messages such as IPsec SA deletions, rekeying, liveness (dead peer detection), and mobility updates. Each exchange can relay additional information about supported features or algorithms using Notify payloads.

[15] In IKEv1, either party could retransmit, which led to race conditions and amplification attacks.

3.2.1 The IKE_SA_INIT Exchange

The IKE_SA_INIT exchange sends the cryptographic IKE proposals for setting up the encrypted IKE SA. Each proposal consists of a list of components needed to establish an IKE SA. These components are called *transforms*. For IKEv2, four types of transforms are required: encryption (AEAD algorithms or encryption algorithms), integrity (none for AEAD[16], or a MAC otherwise), (Elliptic Curve) Diffie-Hellman, and Pseudo Random Function (PRF). The IKE_SA_INIT exchange also includes data that will be used to generate a shared secret that is used to derive symmetric keys to protect later traffic between the two peers, such as the sender's (EC)DH public value (carried in the Key Exchange [KE] payload), a random nonce (in the nonce payload), and both IPsec SPIs (in the IKE Header). The initiator can propose multiple alternative transform combinations, and the responder picks out its preferred proposal with preferred transforms and returns a single proposal with those transforms and its own KE and nonce payloads and a responder SPI.

The initiator needs to know or guess the cryptographic policy that is accepted by the responder. The initiator sends a list of transforms that represents its policy. For the initiator's most preferred (EC)DH Key Exchange algorithm, it will include the corresponding KE payload (e.g., a EC public key). If it turns out that the responder does not allow this (EC)DH algorithm, the responder will reply with an INVALID_KE notification that contains the responder's preferred value based on the list that the initiator sent. The initiator can use this to create a new IKE_SA_INIT packet with a proper KE payload that is acceptable to both initiator and responder policies.

Since an (EC)DH computation is CPU intensive, a malicious entity could send many spoofed IKE_SA_INIT messages, causing the responder to perform multiple (EC)DH calculations, resulting in a denial of service attack. When a responder deems it is under attack, it may respond to an IKE_SA_INIT message with a special COOKIE payload, instead of the regular payloads. The initiator has generated this COOKIE value so it can determine that it has recently generated this COOKIE for a client that is still using the same IP address as when it was given this COOKIE payload. The initiator must resend its IKE_SA_INIT message and include the given COOKIE. This assures the responder that the initiator is a participant in the IKE exchange and not simply sending malicious packets using a forged (spoofed) IP address.

The IKE_SA_INIT exchange is also used to detect the presence of network address translation (NAT) devices. If NAT is detected, the IKE negotiation will move to port 4500, and the IPsec connection will be configured to use UDP or TCP encapsulation to avoid problems with the NAT device rewriting the IP address of the IPsec packets. Often, NAT routers also drop all IP protocols except UDP and TCP, so by encapsulating the IPsec (ESP) packets into UDP or TCP, the packets will not be dropped by the NAT router. The endpoint behind the NAT device will also send one-byte KEEPALIVE packets, typically at 20 second intervals, to ensure that the NAT device will keep the port mapping open that is used by the endpoint behind NAT. This is

[16] AEAD algorithms combine encryption and integrity using a single private key. For the IKEv2 protocol, AEAD algorithms are listed as encryption algorithms. The (separate) integrity algorithm for AEAD is either not included or the special value for None is used.

especially important with deployments of Carrier Grade NAT (CGN) that are typically deployed on mobile data networks (LTE/5G). The KEEPALIVE packets serve no purpose beyond passing the NAT device and are discarded by any endpoint IPsec stack that receives them.

After the IKE_SA_INIT exchange has completed, both endpoints have performed the (EC)DH key exchange and have generated the secret value called the SKEYSEED. All encryption and authentication keys will be derived from this value using the negotiated PRF transform.[17] From here on, all further packets are encrypted. However, both the initiator and the responder still need to authenticate each other's identity.

3.2.2 The IKE_AUTH Exchange

The peers still need to verify each other's identities and prove that the initial unencrypted IKE SA messages were not modified in transit. The IKE_AUTH exchange contains the payloads needed for the receiver to authenticate the sender and its previous IKE_SA_INIT exchange. The IKE_AUTH exchange also contains payloads to negotiate the first IPsec SA, such as the proposals and transforms to negotiate the cryptographic parameters, the source/destination packet policies for the IPsec SA in the form of traffic selectors for the initiator (TSi) and responder (TSr), and other options such as the mode of the IPsec SA and Configuration Payload requests for obtaining an IP address and a DNS nameserver IP address.

Since authentication can involve X.509 certificates and intermediary CA certificates, this packet can end up being larger than the network MTU. To work around networks that do not handle IP fragmentation properly, the IKE protocol itself supports fragmentation to prevent fragmentation at the network layer. Typically, only the IKE_AUTH packets trigger IKE fragmentation.

Typical authentication methods are X.509 certificates, raw public keys (e.g., RSA or ECDSA), or PSKs. IKE supports the Extensible Authentication Protocol (EAP). If EAP authentication is required, more than one IKE_AUTH exchange might be required to complete the authentication. The authentication method can be different between the two endpoints, although they often use the same method. One example of using different authentication methods by each party is a remote access VPN where the server is authenticated using its X.509 certificate, but clients are authenticated via EAP-TLS.[18]

Once the IKE_SA_INIT and IKE_AUTH exchanges have successfully completed, the two hosts have set up an IKE SA and an IPsec SA. Any further communication will be sent using the encrypted and authenticated IKE SA.

[17] Usually, the integrity algorithm and the PRF negotiated are the same algorithm. When using an AEAD cipher that does not require an integrity algorithm, the PRF negotiated is obviously a different algorithm—usually a hash function from the SHA-2 family.

[18] Since IPsec is usually a system service, using a certificate on the client would require administrative privileges on the client. If EAP credentials are used on the client instead, they could be stored in the non-administrative user's own profile.

3.2.2.1 Traffic Selectors

The IKE_AUTH exchange negotiates the IPsec SA network parameters, such as source and destination IP address, address family, source and destination ports, and protocol, using traffic selectors. A traffic selector consists of:

- The traffic selector type (e.g., IPv4 or IPv6 type)
- The IP address range (start address and end address)
- The IP protocol number (0 means all protocols)
- The port range (start and end port, 0-65535 means all ports)[19]

Additional traffic selector components are possible, too, such as Network Label or Security Context.

Traffic selectors are negotiated in sets of two. A set of two traffic selectors denotes the policy for the source and destination traffic of one (inbound or outbound) IPsec SA. The IKE_AUTH request contains at least the TSi and TSr. The TSi describes the sending and receiving address of the initiator, and the TSr describes the sending and receiving address of the responder.

IKEv2 allows the concept of narrowing, where the responder picks a subset of the TSi/TSr that the initiator requested. This facilitates setting up a number of smaller-range IPsec SAs instead of one large network-to-network IPsec SA. This can enhance parallel processing. It is also used for the initiator obtaining an IP address from the responder where the initiator requests every address on the internet (by requesting 0.0.0.0/0) and is narrowed down by the responder to one IP address (for example, 192.0.2.1/32).

An additional traffic selector pair can be included that contains the actual source, destination, and protocol values from the packet that triggered the IKE session at the initiator. This assists the responder in narrowing traffic selectors to a range that includes the traffic that the initiator wants to send to the responder.

3.2.2.2 Configuration Payloads

Optionally, during IKE_AUTH, the hosts can also exchange Configuration Payloads (CPs). The initiator can request a number of configuration options, and the responder can respond with appropriate values. The main CPs are:

- Internal IPv4 and IPv6 address and netmask
- Internal IPv4 and IPv6 DNS server to use as generic DNS resolver
- Internal IPv4 or IPv6 subnet
- Internal IPv4 or IPv6 Dynamic Host Configuration Protocol (DHCP) relay address
- Internal DNS domains for domains that must be resolved via the VPN
- Internal DNSSEC trust anchors to use for internal DNSSEC-signed domains

---

[19] For protocols without ports, 0 is used. For protocols with no ports but types, such as ICMP, the value is used to denote type ranges.

- Application version

All these CPs enable the remote access VPN client to find and use resources on the remote network. And by obtaining an IP address on that remote network, other hosts on that network can potentially reach the remote VPN clients as if they were present locally. CPs are not used and are ignored on gateway-to-gateway and host-to-host IPsec deployments.

CPs are the successor to the IKEv1 non-standard XAUTH and ModeCFG payloads.

3.2.3 The CREATE_CHILD_SA Exchange

The CREATE_CHILD_SA exchange is used for three separate tasks:

- Create an additional IPsec SA
- Rekey an IPsec SA
- Rekey the IKE SA

Creating an additional IPsec SA uses similar IPsec payloads as those used to create the initial IPsec SA in the IKE_AUTH exchange. Either endpoint can initiate a CREATE_CHILD_SA exchange. Lifetimes for IKE and IPsec SAs are not negotiated. Each peer is responsible for rekeying the relevant SAs before the lifetime of their local policy is exceeded.

Rekeying is the process of creating fresh cryptographic keys for an IKE SA or IPsec SA. IKE and IPsec keys are ephemeral and only stored in volatile memory for the duration of the session. Once an SA is rekeyed, the old cryptographic keys are wiped from memory. In the event of a compromise of one of the IPsec hosts, only the current session keys are still in memory and previously recorded sessions cannot be decrypted. IKE SA and IPsec SA session keys typically have a lifetime of one to eight hours. A rekey request can be for one of the IPsec SAs or for the IKE SA. A new IPsec SA is negotiated and installed. The outbound IPsec SA is used immediately. Once traffic is received on the new inbound IPsec SA, the old IPsec SAs are deleted. This ensures that rekeying does not lead to any traffic flow interruptions or leaking of unencrypted packets. Once an IKE rekey is complete, the associated IPsec SAs of the old IKE SA are transferred to the new IKE SA. The old IKE SA is then deleted.

3.2.4 The INFORMATIONAL Exchange

The purpose of the IKE INFORMATIONAL exchange is to provide the endpoints with a way to send each other status and error messages. Some commonly used informational messages are:

- Delete one or more IPsec SAs
- Delete this IKE SA
- Liveness probe (aka Dead Peer Detection (DPD))
- Mobility IP address updates for Mobile IKE (MOBIKE)

Either endpoint can initiate an informational exchange. The other endpoint is obliged to return an answer to prevent the initiator (of the informational exchange) from retransmitting. A delete message denotes the SPI of the IPsec SAs or IKE SA to be deleted. Deleting the IKE SA will also cause all of its IPsec SAs to be deleted.

An endpoint that has not received any IPsec traffic in a while might want to verify if the remote endpoint is still alive. To do so, it can send an informational exchange message (i.e., a probe message) containing zero payloads.[20] An endpoint receiving such an informational message must respond with an empty informational message. If these probes are not answered for a configured time period, the IKE SA and IPsec SA are terminated.

A mobile device that is switching its connection (e.g., from LTE/5G to WiFi) needs to send an informational message with a notification to its remote endpoint. The remote endpoint uses both the content of the informational message, as well as the IP addresses observed from the IKE packet itself, as an indication for which IP address to use as the updated IP address for the mobile endpoint. Successful decryption of the packet (with properly incremented Message ID to prevent replays) verifies the new IP address to use. This process is called Mobile IKE (MOBIKE) and is specified in [32].

3.3 IKE Authentication Models

Different deployments require different authentication methods. Usually, hosts authenticate each other using the same authentication method. But sometimes a client host authenticates a server host differently from the method used by the server to authenticate the client.

3.3.1 Certificate-Based Authentication

This method, also called *machine certificate authentication*, is most often used for deploying IPsec within an organization when it involves a large number of devices. The organization can set up a new internal X.509 certificate deployment or reuse an existing X.509 certificate-based solution. Setting up a new host does not require any changes to the already deployed hosts. Certificate Revocation Lists (CRLs) and the Online Certificate Store Protocol (OCSP) can be used to revoke a particular certificate. Remote access VPN clients are often authenticated using X.509 certificates. Cloud (mesh) encryption also often uses certificate-based authentication.

A host that requires the other end to authenticate itself using certificates can send a CERTREQ payload (during IKE_SA_INIT or IKE_AUTH). Both parties then exchange their certificates in CERT payloads during the IKE_AUTH exchange. Intermediate CAs can also be sent as part of the CERT payload.[21]

Since certificate-based authentication requires certificates generated by CAs that may not be trusted by the organizations verifying the certificates, this method is not always a usable solution to connect two different organizations, as one (or both) of the organizations would need to trust

[20] There will be one encrypted payload containing zero payloads. These probes are sometimes combined with other features, in which case other payloads may be present within the encrypted payload.

[21] Some implementations have (wrongly) implemented sending multiple intermediate CA chains using PKCS#7. This has caused some interoperability issues. It is best to avoid intermediate CAs when possible.

an external CA party not under their own control. For US government organizations, the Federal Bridge CA can be used as a mutually trusted CA.

3.3.2 Extensible Authentication Protocol (EAP)

EAP is a framework for adding arbitrary authentication methods in a standardized way to any protocol. It uses a model of a client, a server, and a backend authentication, authorization, and accounting (AAA) server. The client initiates an EAP authentication to the server. The server forwards these messages to and from the AAA server. The AAA server will let the server and client know that the client and server have successfully authenticated each other. AAA protocols with EAP support include RADIUS [33] and Diameter [34].

The most common EAP method used with IKEv2 is EAP-Transport Layer Security (EAP-TLS), although EAP-Microsoft Challenge Handshake Authentication Protocol version 2 (EAP-MSCHAPv2) is used as well. EAP-TLS uses certificates issued to users, instead of certificates issued to hosts. Some devices, such as mobile phones, often do not make such a distinction. However, laptops generally have non-privileged users that cannot modify the operating system's machine certificate store. These users cannot install a machine certificate but can install a certificate for themselves for use with EAP-TLS.

Usually, Clients use EAP to authenticate themselves to the server, but the server is authenticated by the clients using regular certificate-based authentication.

3.3.3 Raw Public Key Authentication

Authentication using the raw public key of the other entity in a communication (there are no certificates which bind the public key with the other entity's identity) is mostly used for Internet of Things (IoT) devices or when authentication of the public keys is done via publication in DNSSEC.[22] IoT devices often do not have the memory, storage, or CPU capacity to perform X.509 certificate validation. These devices often have a hard-coded public key of the other end in firmware for authenticating its signatures.

When public keys are stored in DNS, and the DNS is secured against tampering or spoofing using DNSSEC, there is no more need to use X.509 certificates. Certificates provide trust via the entity that signs the certificate, but in this case the DNS itself containing the public key is already signed. The trust anchor is not a CA, but a DNSSEC trust key responsible for that part of the DNS hierarchy. And instead of certificates stating the validity period of the public key, raw public keys in DNS are valid as long as these are still published in the DNS. DNSSEC prevents replaying of old DNS data by adding signature lifetimes to DNS records. This type of deployment is most commonly used within a single administrative network, similar to machine-based certificate authentication.

[22] DNSSEC is a system of digital signatures to authenticate DNS content. The DNSSEC core specifications are defined in IETF RFCs 4033, 4034, and 4035.

3.3.4 Pre-shared Secret Key (PSK) Authentication

PSK-based authentication is often deployed because it is the easiest to configure. Each end of the communication has the identity of the other end and their pre-shared key. It does not require generating public keys or certificates or running an EAP infrastructure. It is most commonly used for gateway-to-gateway deployments, as it does not involve adding a third-party trust anchor to the VPN gateway device.

Some deployments use a PSK shared with all remote access VPN clients. Once the PSK has been obtained by an attacker, it can be used to impersonate the remote access VPN server. Even if the clients are using one-time passwords (OTPs), a man-in-the-middle attacker can obtain an OTP and log in as the remote user to the real remote access VPN. Therefore, group PSKs are strongly discouraged.

PSKs are often derived from dictionary words and are less than 32 characters long. Such insecure deployments are vulnerable to offline dictionary attacks.[23] PSKs must have a high entropy value. A good PSK is pseudo-randomly created and has at least 128 bits of entropy.

3.3.5 NULL Authentication

NULL authentication is a special kind of authentication. It really means that no authentication is required. There are two common use cases for this.

The first use case is to deploy IPsec to a large number of nodes where the goal is to only protect against passive attacks. It does not protect against attackers that can perform a man-in-the-middle attack. An advantage is that no authentication system, such as certificates, EAP, or DNSSEC needs to be deployed. For small-scale deployments this method should never be used, and strong PSKs should be used instead. Sometimes a NULL authentication deployment is gradually upgraded to an authenticated deployment.

The second use case only uses NULL authentication for the initiator. The responder still authenticates itself to the client using another authentication method, such as by a machine certificate. This creates a situation that is similar to HTTPS-based web sites: the client remains anonymous, but the server is authenticated. This is the method used for internet-based opportunistic IPsec, where two IPsec hosts attempt to establish an IPsec connection without a pre-existing configuration or knowledge of each other. This usually involves authentication based on DNSSEC or a widely acknowledged CA such as Let's Encrypt.[24] The advantage of this type of deployment is that only the servers need to have an identity for authentication. The clients (usually laptops and phones) do not need to have any kind of identity and can remain anonymous, at least at the network layer. Similar to HTTPS, the application layer might require the client to authenticate before it is allowed to access a particular resource.

---

23 Technically, the attacker needs to man-in-the-middle the VPN client for one IKE_INIT and IKE_AUTH exchange; then the attacker can go offline for the dictionary attack,

24 Let's Encrypt is a non-profit CA that has automated the deployment of free SSL/TLS certificates used to secure website communication, but their certificates can be used for IKE/IPsec as well. https://www.letsencrypt.org

NIST does not recommend the use of NULL authenticated-based IPsec. Any deployment of NULL authenticated IPsec must be categorized as being identical to plaintext unprotected network traffic.

3.4 Network Address Translation (NAT)

During the IKE_SA_INIT exchange, both endpoints exchange information about what they believe their IP address is.[25] The other end will confirm if that matches the source address of the packet they received. If the endpoints detect that a NAT is present, they will move further IKE communication from port 500 to port 4500. The change of UDP port was originally done to prevent bad interaction with NAT devices that tried to support “IPsec passthrough”. This feature caused more harm than good, and by moving to a new port, the IPsec passthrough modifications performed by NAT devices were avoided.

These days, no NAT devices perform IPsec passthrough. Once an IPsec SA has been negotiated, the hosts will also enable UDP or TCP encapsulation of ESP packets to facilitate traversing the NAT over a single port. This avoids two problems. The first problem is that NAT devices commonly only support UDP and TCP, meaning that IPsec (ESP) packets would not be dropped by some NAT devices. The second problem is that the NAT device needs to keep a port mapping between the internal device’s ports used and how these ports are mapped onto the NAT device’s public facing ports. It is easiest if one device behind the NAT device only needs one port mapping for IKE and IPsec (ESP) traffic. The host behind NAT will also send one-byte keepalive packets to ensure that the NAT device does not expire its NAT port mapping if the VPN does not produce any traffic for some time. Otherwise, if the remote IPsec host starts sending traffic towards the NAT device, the NAT device would no longer remember which internal device to forward that traffic to, and the IPsec connection would no longer function.

Some cloud providers issue an ephemeral or semi-static public IP address to some virtual machines inside their cloud. The virtual machines are deployed with only an internal [35] IP address. The cloud infrastructure uses NAT to translate the public IP address to the virtual machine’s private IP address. This NAT will also trigger the NAT traversal mechanism of IKE. This poses another problem. If the IPsec tunnel is configured with the public IP address as the tunnel endpoint, the virtual machine cannot create packets with its public IP address as the source address, since this public IP address is not configured on the machine itself. Packets received after decryption are dropped because the operating system is not looking for packets with the public IP address. A common workaround is for such virtual machines to configure the public IP address on one of their network interfaces.

3.5 IKE Fragmentation

IKE packets can be larger than the common ethernet MTU of 1500 bytes. If these packets are sent over the network, they will most likely be fragmented. Too often, those fragments will be dropped by a firewall and the host will fail to receive the fragments for reassembly. This problem

[25] Technically, they exchange SHA-1 hashes of their IP addresses so as to add some level of privacy regarding the pre-NAT IP addresses used.

is avoided by using IKE fragmentation, which fragments the packets at the application layer instead of the network layer.

IKEv2 fragmentation is specified in RFC 7383 [36]. The main difference with the IKEv1 vendor-specific implementations is that IKEv2 fragments are encrypted. This makes it harder for an attacker to interfere. Note that while the fragments are encrypted, the fragments are not (yet) authenticated because the IKE exchange has not yet completed. Once all fragments have been received, the original IKE packet can be reconstructed and processed as if it was received in one packet.

IKEv2 fragmentation is supported for every exchange type except IKE_SA_INIT. Typically, only the IKE_AUTH exchange requires fragmentation, since that exchange carries the big X.509 certificates.

3.6 Mobile IKE (MOBIKE)

It is common these days that devices, such as mobile phones and laptops, have multiple network interfaces. This allows those devices to switch to cheaper and/or faster networks when available. Phones may use the local WiFi network at the office or at home and mobile networks (5G/LTE) at other locations. Switching also happens when an existing network connection suddenly degrades. Switching networks changes the source IP address used by the device. VPN traffic is still sent to the old, no longer used IP address until the device establishes a new IPsec connection.

MOBIKE [32] addresses this issue. It assumes that an internal IP address is assigned by the VPN on the device using CPs. This internal IP address will remain with this device, regardless of the outer IP address used by the device. Once a device switches between its network interfaces, it will send an INFORMATIONAL exchange packet with an UPDATE_SA_ADDRESS notification. This packet will be sent using the new IP address. The VPN server will be able to recognize the IPsec SA based on the SPI numbers, despite the fact that it is suddenly coming from a different IP address. Once decrypted and authenticated, the VPN server will notice the UPDATE_SA_ADDRESS payload and change the endpoint IP address (and port if encapsulation is used due to NAT). It will reply with a confirmation message. At this point, all IPsec SA traffic is sent and received using the client's new IP address. Since the VPN client's applications are only using the obtained VPN IP address for communication to the remote access network, and this IP address does not change when the device itself changes its network interface and outer IP address, all existing connections remain intact. The applications are not even aware that the network interfaces have switched.

A device that wakes up from battery saving mode will generally send a MOBIKE update whether or not its IP address changed. This ensures any NAT state updates that have happened since the device went to sleep are reported back to the VPN server. For example, the NAT device might have terminated the unused NAT port mapping between the device and the VPN server. The MOBIKE packet will create a new fresh NAT port mapping entry, and the VPN server will immediately be able to update the client's IP address and port number and activate the updated VPN connection.

MOBIKE allows for more complicated setups with multiple IP addresses. While MOBIKE can be used as a failover mechanism for the gateway-to-gateway architecture, care should be taken with such a deployment. If one of the endpoints is compromised, its state could be copied onto a machine on the other side of the world, and a MOBIKE update message could be sent to redirect all traffic to the rogue location. The most secure option is to disable MOBIKE unless the IPsec configuration is for a remote access VPN client.

3.7 Post-Quantum Preshared Keys (PPKs)

It is unclear when a quantum computer will become available. Sufficiently large quantum computers will be able to break the finite field (classic) DH and ECDH key exchanges within the timeframe in which it would be expected that IPsec traffic should remain confidential. That is, the key exchange could be broken in weeks or months, while the expectation of confidentiality would be in the timeframe of decades. Adversaries could store today's encrypted communications for later decryption using quantum computers. This problem is not unique to IKE. Other encryption protocols, such as TLS, suffer from the same problem. It is expected that in the near future, quantum-resistant algorithms will be standardized and deployed for IKE, TLS, and other protocols. Until then, some deployments of IKE and IPsec might use PPKs to strengthen the current algorithms against potential future attacks using quantum computers.

With the exception of IKEv1 using a very strong PSKs, all IKEv1 and IKEv2 configurations are vulnerable to quantum computers. IKEv2 supports Postquantum Preshared Keys (PPKs) [37] as a countermeasure. For the purpose of defending against quantum computers, the PPK works similarly to the PSK in IKEv1 in that the PPK is mixed into the key derivation process in addition to the DH values. The PPK must be a cryptographically strong random key and is exchanged out of band. PPKs are identified by a static or ephemeral PPK Identity. This can be used to protect the identity of the connecting clients and facilitates the use of OTPs as the source of the PPK.

IKEv2 allows the gradual migration of a network from not using PPK to using PPK. First, some hosts are configured with PPK, and when two hosts both support PPK and have each other's PPK ID for which they find a matching PPK, the hosts will use the PPK as an additional input to create the KEYMAT and SKEYSEED that are used as input to the PRFs that generate the keying material for the IKE and IPsec SAs. Once all hosts support PPK, their configurations can be updated to mandate PPK.

While this protects the IPsec SAs since their key material derivation depends on the PPK, the initial IKE SA DH process is not protected by the PPK and can still be broken by a quantum computer. This will lead to a loss of privacy of the IKE identities and other information exchanged during the initial IKE Exchange, such as the traffic selectors used for the first IPsec SA. This can be prevented if the IKE implementation allows setting up a childless IKE SA (without IPsec) and then immediately rekeying the IKE SA. This rekeyed IKE SA is protected by the PPK, and IPsec SAs can then be set up using this new IKE SA without exposing any information to adversaries with quantum computers.

PPKs shall have at least of 128 bits of entropy.

3.8 IKE Redirect

The IKE Redirect [38] notify payload allows an IPsec server to send a redirection request to connecting or connected VPN clients. This can be used to reduce the load of overloaded IPsec servers or to take a server out of use (for instance, to update its operating system). Clients being redirected MUST use the same credentials they were originally using before being redirected. A redirection message includes an IP address or DNS name of the forwarding VPN that the VPN client will need to initiate a connection with .

Redirected messages sent in IKE_AUTH are only processed after both ends have authenticated each other. This allows a server to only send specific clients to another server, for instance all clients of a certain customer in a multi-tenant deployment or some individual power users generating a lot of traffic. But it still requires that the (overloaded) server performs full IKE exchanges to all connecting clients, only to redirect them to different server hosts.

Redirected messages sent in IKE_SA_INIT are not authenticated. Clients that accept such redirected messages should take necessary precautions to prevent denial of service attacks. The advantage for the host performing the redirection is that it can redirect clients without performing a full IKE exchange.[26] The disadvantage is that redirections in IKE_SA_INIT cannot select the specific clients for redirection by their IDs, since the client ID has not yet been transmitted to the server.

Redirected messages can be used to provide a redundant set of servers for the gateway-to-gateway deployment. A failing server can redirect clients to the other (backup) server. In such an architecture, it is recommended that redirect messages be limited for each endpoint based on preconfigured IP addresses.

3.9 Differences Between IKEv2 and the Obsolete IKEv1

The IKEv2 protocol builds on the lessons learned with IKEv1. IKEv2 is simpler, faster, and more secure. IKEv2 has some important new features over IKEv1, such as mobility support (MOBIKE), support for newer cryptographic algorithms, anti-distributed denial of service (DDoS) support, and server redirection support. It is recommended that existing IKEv1 installations be upgraded to IKEv2.

For those familiar with IKEv1, the main differences between IKEv1 and IKEv2 are:

- IKEv1 was designed to be a far more general-purpose key exchange protocol, but many extraneous features ended up not being used at all. IKEv2 no longer has these features.
- Some IKEv1 protocol extensions are now part of the IKEv2 core specification, such as IKE fragmentation[27], NAT Traversal, and Liveness Detection—formerly called Dead Peer Detection (DPD). This means that these features are always available in IKEv2.

---

[26] Most importantly, it can skip the DH calculation, which is the most expensive operation of an IKE exchange.

[27] Technically, IKE fragmentation is a separate RFC, but it is implemented by most vendors.

- IKEv1 has a large number of exchange types to choose from (Main Mode, Aggressive Mode, Revised Mode, etc.) With IKEv2, there is no choice of exchange methods, so this no longer needs to be explicitly configured.
- The IKEv2 exchange has anti-DDoS protection using cookies.
- When an IKEv1 endpoint uses the wrong PSK to encrypt a message, the other endpoint is unable to decrypt the encrypted message. For the endpoint receiving this erroneous message, it has no way to distinguish this error from other problems such as packet corruption.
- In IKEv1, both endpoints are responsible for retransmissions, leading to conflicting retransmits and denial of service vectors. In IKEv2, only the exchange initiator is responsible for retransmission.
- In IKEv1, the IKE SA can expire while the IPsec SA is still active. This could lead to strange scenarios with DPD. In IKEv2, every IPsec SA has an IKE SA. If the IKE SA expires, all IPsec SAs are torn down as well. This guarantees that every IPsec SA has a functional control channel, which was not the case with IKEv1.
- In IKEv1, rekeying always requires a reauthentication of the two end points. Some proprietary extensions allow rekeying without reauthentication. Reauthentication is not always desirable, especially with the use of OTPs or hardware tokens requiring the use of a PIN or fingerprint for activation by the user (such as a VPN client), as it would require human interaction to keep the IPsec connection alive. In IKEv2, rekeying and reauthentication are separate processes with their own lifetimes.
- In IKEv1, transport mode and compression are negotiated, and a mismatched configuration would lead to a fatal IKE error. In IKEv2, the initiator can request these, but if the responder does not confirm those requests, the IPsec SA is established in tunnel mode (or without compression).
- In IKEv1, the IKE SA and IPsec SA can use different DH groups during key establishment (i.e., the DH group used to establish the IKE SA can be different than the DH group used to establish the IPsec SA). This is possible because the IKE and IPsec parameters are negotiated in 2 different message exchanges, taking place at different times. In IKEv2, there is only one exchange of parameters, and the first IPsec SA is established using the IKE SA DH group. Subsequent IPsec SAs can perform an additional DH exchange, thus ensuring the property of PFS; that exchange can use a different group. However, when configuring multiple IPsec SAs, there is no guarantee which one will be brought up first, either through an operator or by on-demand tunnel establishments. Therefore, in IKEv2 the DH group selected should be the same for the IKE SA and the IPsec SAs.
- In IKEv1, ESP encapsulation can only happen in UDP. IKEv2 can also use TCP and TLS encapsulation on any port. The TCP/TLS encapsulation cannot be negotiated and must be configured manually or via configuration provisioning. TCP port 4500 is often the default used. This might require firewall-rule updates.
- When migrating from IKEv1 to IKEv2, an upgrade of the algorithms used is strongly recommended. 3DES, MD5, SHA-1 and DH Group 2 and 5 should not be used. Instead, AES-XCBC with HMAC-SHA-2 or AES-GCM with either DH group 14 or an ECDH group (19, 20, or 21) should be used.

- IKEv2 Traffic Selector negotiations allow narrowing. This helps with creating multiple parallel IPsec SAs per traffic flow, which generally improves performance as hardware (i.e., central processing units [CPUs] and network interface cards [NICs]) can then handle multiple parallel streams at once.
- In IKEv1 it is not always possible to detect different groups of clients early enough to select the right authentication mechanism or the right PSK. This complicates multi-tenant VPNs. In IKEv2, the initiator can optionally send the expected ID of the peer in the IDr payload. This allows the responder (i.e., the server) to always select the proper tenant group.
- IKEv1 with PSK has the side effect of offering quantum computing resistance. In IKEv2 this is no longer the case, but a separate RFC [37] specifies how to use PPKs to gain the same protection in IKEv2.

3.10 Manual Keying

While it is possible to hard-code the IPsec information using out-of-band communication—called *manual keying*—this MUST NOT be used. The IKE protocol handles a number of other security properties, none of which are enforced when using manual keying. Encryption keys would never be refreshed when a fixed key is manually input and used, so any compromise would allow an attacker to decrypt all previously monitored traffic under the fixed key. Some values, such as nonces, counters, and IVs, must never be used more than once, otherwise the encryption may become vulnerable (weaken).

The only time that manual keying might be acceptable is if another trusted entity, such as a Security Controller in the SDWAN paradigm, assumes these responsibilities. Another example is the 3GPP protocol, which negotiates the IPsec parameters between a cell tower and handset using a non-IKE protocol.

Administrators sometimes mistakenly believe that manual keying is easier to set up than automated keying via IKE. However, manual keying is much harder to set up than IKE.

Manual keying is typically only used for software testing and IPsec benchmark tests.

This recommendation discourages the use of manual keying.

3.11 IKE Summary

- IPsec uses IKE to create security associations, which are sets of values that define the security of IPsec-protected connections. The first IPsec SA is created in conjunction with the IKE SA during the initial exchanges.
- The IKE SA is used to securely communicate IPsec configuration, status, and management information, such as setting up additional IPsec SAs, rekey events, deletions, and other notifications.
- IKEv2 is faster, more versatile, and uses more modern cryptography compared to IKEv1. IKEv1 should not be used for new deployments, and existing deployments using IKEv1 should be converted to IKEv2 when possible.

4 The IPsec Protocols

IPsec is a collection of protocols that assist in protecting communications over networks.[28] This section focuses on the primary component of IPsec, the Encapsulating Security Payload (ESP), which protects the confidentiality and integrity of data packets. The section also briefly covers the other IPsec components, the IP Payload Compression Protocol (IPComp) and the Authentication Header (AH) protocol. All the parameters and cryptographic keys needed by the IPsec protocols are negotiated using the IKE protocol as described in Section 3.

4.1 Encapsulating Security Payload (ESP)

ESP is the core IPsec security protocol. It has largely been unchanged since its second version, published in 1998. The current version (IPsec-v3) was specified in RFC 4303 in 2005 [19]. It contains only a few updates to the IPsec-v2 specification in RFC 2406 [39]. Since all the changes to ESP are either backwards compatible or are new features that would need to be negotiated via IKE before these are enabled for ESP, there are no compatibility issues between IPsec implementations receiving and sending ESP packets. Regardless, practically all current implementations support IPsec-v3. Features only available in IPsec-v3 are:

- Support for AEAD algorithms
- Extended Sequence Numbers (ESNs)
- Enhanced policy support (via Security Policy Database [SPD]/Security Association Database [SAD])
- Padding support
- Dummy packet support

The use of padding and the capability of sending dummy messages increase traffic flow confidentiality (TFC) by making it harder for an eavesdropper who cannot decrypt the packets to deduce anything from the encrypted packet sizes or timings.

ESP provides encryption and integrity protection. The outer header is not fully protected, allowing for routers that forward ESP packets to still modify certain flags, such as Quality of Service (QoS) and Time to Live (TTL) values.

ESP's encryption functionality can be disabled through the selection of the Null ESP encryption algorithm or the AES-GMAC AEAD algorithm. AES-GMAC is a variant of the AES-GCM algorithm that provides integrity protection without encryption. ESP can be used to provide either encryption and integrity protection; or only integrity protection. AH deployments should be migrated to these ESP algorithms. ESP supports AEAD and classic (non-AEAD) encryption with integrity methods.

28 RFC 4301 provides an overview of IPsec [40].

4.1.1 Tunnel Mode and Transport Mode

ESP has two modes: transport and tunnel. In *tunnel mode*, (see Figure 7), a new packet is constructed that contains the (original) IP packet being sent through the tunnel by 1) placing an ESP header and trailer around the original IP header and its payload, 2) encrypting the original header, payload and ESP trailer, 3) computing an integrity check value (ICV) over the ESP header and the encrypted data, 4) placing the ICV at the end of the packet being constructed, and 5) adding a new IP header to the beginning of the packet. The ICV computation does not include the new IP header.

The new IP header lists the endpoints of the ESP tunnel (such as two IPsec gateways) as the source and destination of the packet, and contains as its payload the entire, now encrypted, original packet. Because of this, tunnel mode can be used with all VPN architectures described in Section 2.4. As shown in Figure 7, tunnel mode can encrypt and protect the integrity of both the data and the original IP header for each packet. Encrypting the original IP header and its payload protects their confidentiality; encrypting the original IP header conceals the nature of the communications, such as the actual source or destination of the packet, protocol, and ports used that would indicate which application is likely being used. The ICV is used to detect any changes to the data over which the ICV is computed.

New IP Header	ESP Header	Original IP Header	Original IP data containing Transport and Application Protocol Headers and Data (optional TFC padding)	ESP Trailer (ESP padding, Next Header)	ESP Integrity Check Value - ICV (variable)
		Encrypted			
	Authenticated (Integrity Protection)				

Figure 7: ESP Tunnel Mode Packet

ESP tunnel mode is used for gateway to gateway deployments, remote access VPNs, and various network virtualization deployments. It is also required when the IPsec connection needs to traverse a NAT, which rewrites the outer IP address.

For host-to-host deployments within data centers, local networks, and virtual machines where no NAT is deployed, ESP transport mode is often used. In *transport mode* (see Figure 8), ESP uses the original IP header instead of creating a new one. The ESP payload and trailer are encrypted, and an ICV is computed over the ESP header and the encrypted data. Integrity protection is not provided for the IP header. The overhead of the transport mode is less than for the tunnel mode because it does not have to create an entire new IP header.

Transport mode is incompatible with NAT. For example, in each TCP packet, the TCP checksum is calculated on both the TCP and IP fields, including the source and destination addresses in the IP header. If NAT is being used, one or both of the IP addresses are altered, so NAT needs to recalculate the TCP checksum. If ESP is encrypting packets, the TCP header is encrypted; NAT cannot recalculate the checksum, so NAT fails. This is not an issue in tunnel mode; because the entire TCP packet is hidden, NAT will not attempt to recalculate the TCP checksum of the inner encrypted packet, only of the outer IP address which is not part of the ESP encryption. However, tunnel mode and NAT have other potential compatibility issues.[29] Section 7.2.1 provides guidance on overcoming NAT-related issues.

IP Header	ESP Header	Transport and Application Protocol Headers and Data	ESP Trailer (ESP padding, Next Header)	ESP Integrity Check Value- ICV (variable)
		Encrypted		
	Authenticated (Integrity Protection)			

Figure 8: ESP Transport Mode Packet

4.1.2 Encryption with Separate Integrity Protection

ESP uses symmetric cryptography to provide encryption for IPsec packets. Accordingly, both endpoints of an IPsec connection protected by ESP encryption must use the same key to encrypt and decrypt the packets. When an endpoint encrypts data, it divides the data into small blocks (for the AES algorithm, blocks of 128 bits each), and then performs multiple sets of cryptographic operations (known as rounds) using the data blocks and key. Encryption algorithms that work in this way are known as *block cipher algorithms*. When the other endpoint receives the encrypted data, it performs decryption using the same key and a similar process, but with the steps reversed and the cryptographic operations altered.

After encryption has been performed, the first step for providing integrity protection is to create a MAC on a message using a MAC algorithm and a secret key shared by the two endpoints. The MAC is added to the packet, and the packet is sent to the recipient. The recipient can then regenerate the MAC using the shared key and confirm that the two MACs match, thus determining whether the data has been modified. IPsec mostly uses a keyed-hash message authentication code (HMAC) algorithm [41] for integrity protection, which uses approved hash functions. Examples of HMAC are HMAC-SHA-256 and HMAC-SHA-1. Another common non-HMAC integrity algorithm is AES Cipher Block Chaining MAC (AES-XCBC-MAC-96) [42].[30]

4.1.3 AEAD Encryption with Built-In Integrity

Encryption with separate integrity protection (as described in Section 4.1.2) requires two separate cryptographic processes over the data using two different secret keys. AEAD combines these two processes. This significantly increases performance. It also provides more constant-time processing when errors occur, resulting in a more robust error handling process that is less susceptible to timing attacks. The reverse process produces either the plaintext data or an error indication. For IKEv2 and ESP, AES-GCM is specified in [43] as an AEAD algorithm. Due to

[29] One possible issue is the inability to perform incoming source address validation to confirm that the source address is the same as that under which the IKE SA was negotiated. Other possible issues include packet fragmentation, NAT mapping timeouts, and multiple clients behind the same NAT device.

[30] Federal agencies are required to use NIST-approved algorithms and FIPS-validated cryptographic modules. HMAC with a hash function from the SHA-2 family is NIST-approved, but AES-XCBC-MAC-96 is not.

the way that IKEv1 handles the separation of encryption from data integrity protection in IKE packets, AEAD algorithms cannot be used in IKEv1. IKEv1 can, however, still negotiate AEAD algorithms for ESP.

The nonce used by an AEAD algorithm must be unique for every encryption operation with the same secret key but does not need to be unpredictable.[31] The nonce in IKE is built using an implicit part (the salt) and an explicit part (the initialization vector, or IV). The implicit part is based on the keying material calculated from the DH key exchange and negotiated PRF, similarly to how secret encryption keys are generated. This value is never transmitted and binds the encryption to the DH channel. The explicit part is transmitted and usually based on an increasing, and thus unique, counter. Reuse of the IV with the same secret key compromises the security of the data. Thus, these algorithms must be used in conjunction with IKE, and cannot be used with static or manual keys. An SA must be terminated before the counter reaches its maximum possible value.

4.1.4 Common ESP Algorithms

Examples of common algorithms used by ESP are AES-GCM [44] and AES-Cipher Block Chaining (AES-CBC) [45] with a SHA-2-HMAC. Most algorithms have limitations on the amount of data that can be safely encrypted with a single key, and requirements for auxiliary parameters.

The Triple DES (3DES) encryption algorithm is no longer recommended. It is much slower than AES-GCM and AES-CBC, and it requires more frequent rekeying to avoid birthday attacks due to its smaller block size of 64 bits. The HMAC-MD5 and HMAC-SHA-1 integrity algorithms are also no longer NIST-approved.

For the latest cryptographic recommendations, see NIST SP 800-131A [47] and FIPS 140 [13].

4.1.5 ESP Packet Fields

ESP adds a header and a trailer around each packet's payload. As shown in Figure 9, each ESP header is composed of two fields:

- **SPI.** Each IPsec SA (inbound and outbound) contains an SPI value, which acts as a unique identifier for the IPsec SA. The endpoints use these SPI values, along with the destination IP address and (optionally) the IPsec protocol type (in this case, ESP) to determine which SA is being used, and which decryption key should be used.
- **(Extended) Sequence Number.** Each packet is assigned a sequential sequence number, and only packets within a sliding window of sequence numbers are accepted. This provides protection against replay attacks because duplicate packets will use the same

[31] The terms nonce and IV have not seen consistently use between NIST and IETF publications. In general, what is required is the use of a guaranteed unique non-secret value. Note that the IV needed for the AEAD algorithm is separate from the integrity check value (ICV) used in each packet to ensure that two identical plaintext payloads encrypt to different encrypted payloads (and thus cannot be detected as identical).

sequence number. This also helps to thwart denial of service attacks because old packets that are replayed will have sequence numbers outside the window and will be dropped immediately without performing any more processing. Originally (in IPsec-v2) the sequence numbers for IPsec packets were defined as a 32-bit number. Current hardware can transmit 100 gigabits per second (Gbps), or about 150 million packets per second, meaning that the 32-bit sequence number space would be exhausted in 30 seconds. It would be impractical to rekey an IPsec SA every 30 seconds, so IPsec-v3 [19] introduced Extended Sequence Numbers (ESNs). If negotiated with IKE, the IPsec SA is installed with 64-bit sequence numbers. The ESP wire format is unchanged, however, and only the lower 32 bits of the Sequence Number are transmitted in the ESP packet. Each endpoint keeps track of the higher 32-bit value and performs all integrity calculations based on the entire 64-bit sequence number.[32]

The next part of the packet is the payload. It is composed of the encrypted payload data and the IV, which is not encrypted. This is helpful in deterring traffic analysis. The IV is used during encryption. Its value is different in every packet, so if two packets have the same content, the inclusion of the IV will cause the encryption of the two packets to have different results. This makes ESP less susceptible to cryptanalysis.

To obfuscate the length and frequency of information sent over IPsec, the protocol allows for sending dummy data called *traffic flow confidentiality (TFC) padding*. TFC padding can be added to the unencrypted data before encryption, or it can be injected as a whole new packet with only padding being encrypted to a certain size between real encrypted data transmissions. An observer cannot tell if TFC is enabled, and more importantly, can no longer make any reasonable assumptions based on packet size or frequency. One common deployment of TFC is to pad all packets to the maximum MTU value, resulting in all ESP packets sent being the exact same length. This would increase the amount of encrypted data sent, so on links where transmission costs depend on the amount of data sent (e.g., LTE/5G), there is a cost associated with using TFC.

The third part of the packet is the ESP trailer, which contains at least two fields and may optionally include one more:

- **ESP Padding.** An ESP packet may optionally contain padding, which is additional bytes of data that make the packet larger and are discarded by the packet's recipient. Because ESP uses block ciphers for encryption, padding may be needed so that the encrypted data is an integral multiple of the block size. Padding may also be needed to ensure that the ESP trailer ends on a multiple of four bytes.
- **ESP Padding Length.** This number indicates the length of the padding in bytes. The Padding Length field is mandatory.

32 It is assumed that an application would notice a packet loss of 2^{32} packets, which would lead the hosts to use a different high-order 32-bit value and fail the integrity check of the packet. [48] does specify a method of coping with such an unusual situation.

- **Next Header.** In tunnel mode, the outer (original) IP header is followed by an inner (new) IP header; thus, the next payload is an IP packet, so the Next Header value is set to four, indicating IP-in-IP (one IP packet tunneled in another IP packet). In transport mode, the payload is usually a transport layer protocol, often TCP (protocol number 6) or UDP (protocol number 17). Every ESP trailer contains a Next Header value.
- **Integrity Check Value (ICV).** This is used to verify the integrity of the encrypted data. For AES-GCM and AES-Counter with CBC-MAC (AES-CCM), it consists of an 8, 12, or 16-byte Authentication Tag. The 16-byte ICV value is recommended by NIST and by RFC 8247 [20]. The recipient of the packet can recalculate the ICV value to confirm that the portions of the packet other than the outermost IP header have not been altered in transit.

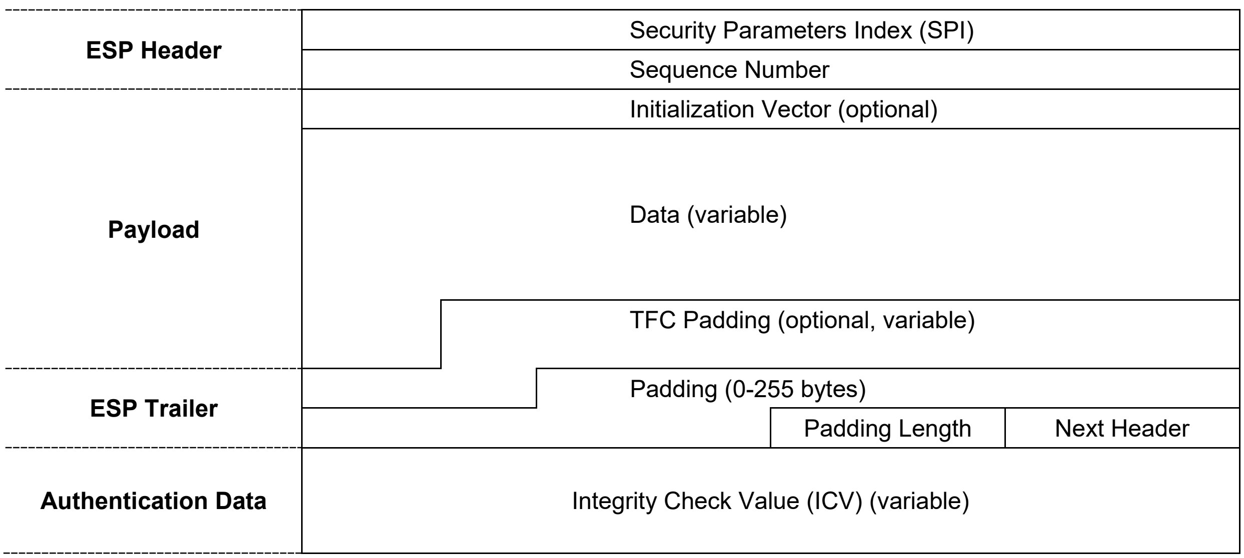

Figure 9: ESP Packet Fields

4.1.6 How ESP Works

Reviewing and analyzing actual ESP packets can provide a better understanding of how ESP works. Figure 10 shows the bytes that compose an actual ESP packet and their ASCII representations. The ESP packet only contains four sections (ignoring the link layer): IP header, ESP header, encrypted data (payload and ESP trailer), and (optionally) authentication information. By examining the encrypted data, it is not possible to determine if this packet was generated in transport mode or tunnel mode. However, because the IP header is unencrypted, the IP protocol field in the header does reveal which IPsec protocol the payload uses (in this case, ESP). As shown in Figure 7 and Figure 8, the unencrypted fields in both modes (tunnel and transport) are the same.

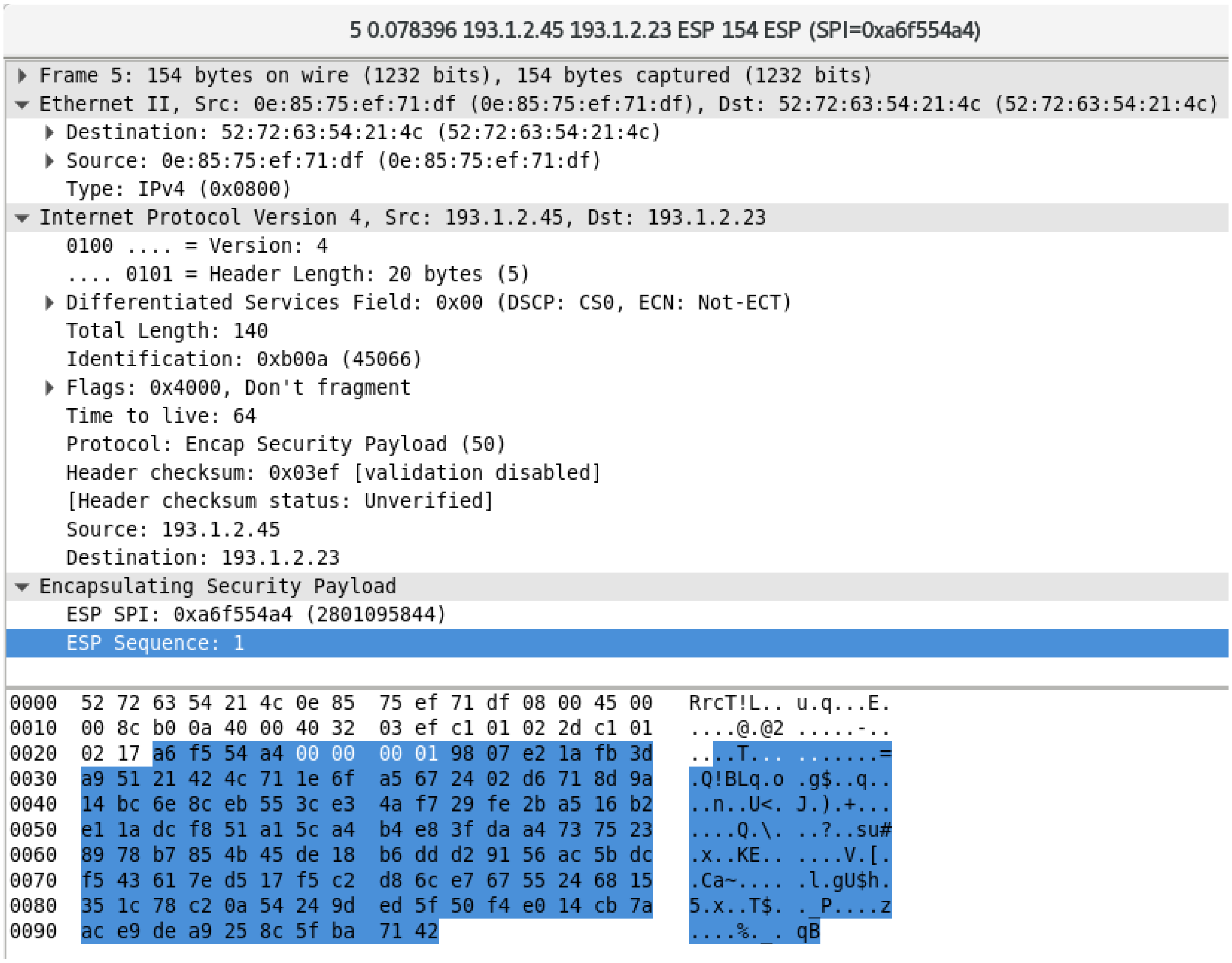

Figure 10: ESP Packet Capture Using Wireshark, Showing Sequence Number 1

Although it is difficult to tell from Figure 10, the ESP header fields are not encrypted. Figure 11 shows a network traffic capture, made with the tcpdump tool, of encrypted traffic generated by the ping command, followed by an IKE session, followed by another ping that is now protected by ESP. Each direction uses its own negotiated SPI value for its packets, which corresponds to an ESP connection being composed of two one-way connections, each with its own SPI. Both hosts initially set the sequence number to 1, and both incremented the number to 2 for their second packets. The tcpdump tool labels IKE packets as “isakmp”, a legacy name from the IKEv1 protocol.

```
13:45:34.118804 IP 203.0.113.1 > 198.51.100.1: ICMP echo request, id 27083, seq 2, length
64
13:45:34.118850 IP 198.51.100.1 > 203.0.113.1: ICMP echo reply, id 27083, seq 2, length 64
13:45:39.469941 IP 203.0.113.1.isakmp > 198.51.100.1.isakmp: isakmp: parent_sa
ikev2_init[I]
13:45:39.472043 IP 198.51.100.1.isakmp > 203.0.113.1.isakmp: isakmp: parent_sa
ikev2_init[R]
13:45:39.481690 IP 203.0.113.1.isakmp > 198.51.100.1.isakmp: isakmp: child_sa
ikev2_auth[I]
13:45:39.525826 IP 198.51.100.1.isakmp > 203.0.113.1.isakmp: isakmp: child_sa
ikev2_auth[R]
13:45:39.587728 IP 203.0.113.1 > 198.51.100.1: ESP(spi=0xc55ed62b,seq=0x1), length 120
13:45:39.587773 IP 198.51.100.1 > 203.0.113.1: ESP(spi=0xf6fc7c09,seq=0x1), length 120
13:45:40.646761 IP 203.0.113.1 > 198.51.100.1: ESP(spi=0xc55ed62b,seq=0x2), length 120
13:45:40.646800 IP 198.51.100.1 > 203.0.113.1: ESP(spi=0xf6fc7c09,seq=0x2), length 120
```

Figure 11: tcpdump Capture of ping, IKE, and ESP Packets

4.2 ESP Encapsulation

ESP packets cannot traverse a NAT device in all circumstances. If an IPsec connection uses transport mode, changing the IP address on the packets will invalidate the integrity checks imposed by IPsec. The NAT device cannot rewrite the ICV because it does not have access to the keying material needed to do so. For all intents and purposes, the NAT device is a malicious actor that IPsec protects against.

The ESP protocol has no ports. If multiple clients send ESP from behind the same NAT router, it would be difficult to track the ESP packets to the respective clients, as they would all have the same destination IP—that of the NAT device. And while SPI numbers are uniquely generated for each IPsec host, there is no guarantee that two hosts behind the same NAT will not end up picking the same SPI number for an IPsec SA. Furthermore, often NAT routers do not understand or translate anything other than the UDP and TCP protocols, causing ESP packets to be dropped by the NAT device.

4.2.1 UDP Encapsulation of ESP

To overcome these issues, ESP can be encapsulated in UDP (ESPinUDP). The NAT device can rewrite the IP address of the outer UDP packet and track multiple clients by the UDP port number. For historical reasons,[33] when IKE detects a NAT during the negotiation, it switches the IKE negotiation from UDP port 500 to UDP port 4500. It uses a regular UDP packet header, followed by a four-byte header with all zeroes (Non-ESP Marker) following the UDP header. Then the IKE header follows.

ESPinUDP also uses port 4500 to ensure that the NAT device only has one NAT mapping for all traffic (ESP and IKE). Following the regular UDP packet header, the ESP header follows. The first four bytes of the ESP header is the SPI number, which cannot be 0. Thus, an implementation receiving a packet on port 4500 can determine whether the packet is an ESPinUDP packet or an

[33] Some NAT devices tried to be helpful by looking at the SPI and rewriting or multiplexing these. It just made things break more. The solution was to avoid UDP port 500 completely to avoid any NAT “helper” algorithms. IKEv2 even allows skipping UDP port 500 altogether and using UDP port 4500 for all IKE messages.

IKE packet, depending on whether or not it sees the SPI number of the non-ESP marker. Usually, the kernel receiving an ESPinUDP packet will just strip the UDP header away without bothering with the UDP checksum (which not all NAT routers properly recalculate) and process the remaining ESP data as if it was received as an ESP packet without encapsulation. If the kernel detects an IKE packet, it will send this packet to the IKE process for processing by the IKE daemon.

Starting with IKEv2, even if no NAT was detected, endpoints need to support receiving ESP and ESPinUDP packets on all their IPsec SAs. Each endpoint may decide when to use encapsulation and when not to. IKEv2 also allows initiating a new IKE_SA_INIT on UDP port 4500, bypassing UDP port 500 completely.

4.2.2 TCP Encapsulation of ESP

Implementations supporting TCP encapsulation [49], where ESP packets are wrapped into a TCP stream, can also choose to use TCP. This provides a much-needed method to prevent IPsec from being easily filtered and blocked. Lacking TCP encapsulation was one of the reasons why SSL VPNs came into existence, as these could not be easily blocked by blocking the IPsec protocols (UDP port 500 and 4500 and protocol ESP). TCP encapsulation ports cannot be negotiated, as this would require that the negotiations start on the well-known port susceptible to blocking. Therefore, the TCP port has to be preconfigured manually or via the IPsec client provisioning system.

The ESP in TCP encapsulation uses an ASCII prefix tag of “IKETCP” so that an additional layer can be used, such as TLS. In that case, encrypted packets are encapsulated using a TCP connection that uses TLS. The packet processor can read the prefix and detect the start of an IKE/ESP stream, in which case it can send this traffic to the proper handler. Since restrictive networks often still (have to) allow access to HTTPS websites, using TLS on port 443 to protect (or really, hide) the TCP stream containing the encapsulated ESP packets will yield the best results. However, networks are often only misconfigured to drop all UDP traffic. Moving to ESP encapsulation on TCP port 4500 without TLS framing will usually be enough to be able to establish IPsec connections.

Implementations are encouraged to regularly try to go back to UDP encapsulation. TCP encapsulation means there are possibly two TCP layers involved in a packet: the TCP connection being encrypted and the TCP connection carrying the ESP packet. These two TCP layers will both independently determine retransmissions. Especially when there is packet loss, these two TCP streams will badly interfere with each other.

4.3 IP Payload Compression Protocol (IPComp)

ESP can be deployed with IPComp. Before a packet is encrypted, the packet will be considered for compression. If the packet is very small already, such as an ICMP message, no compression is done, and the packet is encrypted as is; otherwise, the packet is compressed. However, various compression algorithms do not guarantee that an attempted compression does not end up being larger than the original. If this turns out to be the case, the original packet is encrypted without compression. If the compressed result is smaller, the compressed packet is encrypted. On the receiving end the packet is decrypted, and if it was compressed, it will be decompressed.

However, applications that send large amounts of data usually already compress their data. At that point, attempting to compress already compressed data will not yield smaller packets, and a host only ends up wasting CPU cycles at the IPsec layer attempting futile compression. As such, IPsec level compression has not seen widespread use. This might change in the near future with the emergence of IoT devices and other battery-powered devices that use mobile data (LTE/5G). These devices save battery power by transmitting fewer bytes, even if that reduction requires more CPU power for compression.

4.4 Authentication Header (AH)

As with ESP, AH can be used in tunnel mode and transport mode. It only offers integrity algorithms and provides no confidentiality. The ESP protocol can use null encryption (ESP algorithm number 12) with an integrity algorithm such as HMAC-SHA-2[34] to accomplish the same as AH. Alternatively, ESP can use an AEAD algorithm such as AES-GMAC (ESP algorithm number 21) to offer integrity without confidentiality to replace AH.

The use of AH is discouraged in this publication. The IETF has specified that AH is an optional IPsec protocol, which means it is not mandatory to implement and might not be available with all IPsec implementations. It is recommended that null encryption with the ESP protocol be used instead of the AH protocol when encryption is not desired.

Some implementations support the legacy IPsec-v2 ESP without authentication in combination with AH. This is usually referred to as *AH+ESP*. This combined mode (ESP for encryption and AH for integrity) is no longer recommended [20], as it provides no advantage over regular ESP with authentication. Regular ESP with authentication also reduces the MTU compared to AH+ESP, due to the additional overhead of an AH header plus an ESP header versus just an ESP header with authentication.

NIST discourages the use of AH.

4.5 Summary

This section has described the IPsec protocols ESP, IPComp, and AH. The following summarizes the key points from the section:

- The IKE protocol is used to manage IPsec security associations.
- ESP is the main IPsec protocol and provides integrity protection for all packet headers and data, with the exception of a few IP header fields that routinely change unpredictably in transit. Since those header fields can change as the packet travels from sender to receiver, they cannot be included in the integrity check calculation; if they were, that

[34] HMAC-SHA-2 is used throughout the document to mean HMAC using a hash function from the SHA-2 family of hash functions specified in FIPS 180 [25].

value would then be different for the sender and the receiver. ESP also provides confidentiality protection through the use of encryption, encrypting the data. It does not encrypt the headers, since the header fields are used to correctly process and deliver the data as it traverses the Internet.

-
- ESP can be used in transport mode and tunnel mode.
  - In tunnel mode, ESP provides encryption and integrity protection for an encapsulated IP packet, as well as integrity protection for the ESP header of the outer (constructed) IP packet.
  - In transport mode, ESP provides encryption and integrity protection for the payload of the IP packet, as well as integrity protection for the ESP header. Transport mode is not compatible with NAT. Transport mode can only be used for host-to-host deployments. It is commonly used for large scale host-to-host mesh deployments within an administrative domain without NAT.
- ESP in tunnel mode is the most commonly used IPsec mode because it can encrypt the entire original IP packet, which conceals the true source and destination of the packet. ESP in tunnel mode is a requirement for gateway-to-gateway communications. ESP in tunnel mode can be encapsulated in UDP and TCP, making it compatible with NAT.
- ESP can add padding to packets and send dummy packets, further complicating attempts to perform traffic analysis.
- ESP can use IPComp but rarely does because the gains made from data compression depend strongly on the type of traffic sent. Applications sending a lot of data typically compress their data before providing it to the lower layers for transmission. Applying IPComp to already compressed data would waste CPU power.
- AH has been obsoleted and should not be implemented or deployed. If encryption is undesirable, ESP with null encryption (ESP-NULL or AES-GMAC) should be used instead of AH.

5 Deployment of IPsec Using IKE

This section describes the interactions between the IKE and IPsec subsystems. The interaction depends on the implementation. This section describes the standard protocols used to communicate between IKE and IPsec. However, some devices have their own proprietary method of communication. In general, the concepts explained in this section will apply to those proprietary implementations as well.

The IKE protocol is usually implemented as an application running on the operating system, whereas the IPsec protocol is generally implemented in the kernel of the operating system. Some devices implement the IPsec subsystem in userland, but for the remainder of this chapter it is assumed that IPsec is implemented in the kernel.

The communication between IKE and IPsec is usually implemented using the PF_KEYv2 [50] or NETLINK [51] protocol. Linux uses NETLINK with the XFRM application programming interface (API), whereas BSD-based systems use PF_KEYv2.[35]

This section puts IKE and IPsec components together to illustrate how IPsec sessions are set up and executed. Each example includes the use of IKE to establish SAs.

5.1 IPsec States and Policies

Each IPsec SA has a state and a policy. While each state must have a policy, not all policies need to have a state. For example, on-demand IPsec connections have a policy that allows the kernel to detect that an outgoing packet should trigger an IKE negotiation. Once the IKE SA has been established and an IPsec SA has been negotiated, the IKE daemon will install an IPsec state with corresponding policies. During the negotiation, the kernel can drop the packet, cache the packet for later transmission, or let the packet go out unencrypted. Usually UDP packets are dropped, since their unreliable nature requires that applications sending these packets need to know when to transmit their packets anyway. TCP packets are usually cached because TCP retransmissions are usually very slow, and it would make the on-demand tunnel very slow if the first TCP packet is always lost. Leaking packets in cleartext is only done when the network considers the IPsec protection optional instead of mandatory.

Once an IPsec SA has been established between two hosts, all traffic that falls within the IPsec SA policy MUST be IPsec-protected. If for some reason unencrypted traffic is received, it is assumed to have been forged, and the traffic will be dropped.

5.1.1 The Security Association Database (SAD)

The kernel maintains a state for each IPsec SA. An IPsec connection between two hosts consists of a pair of IPsec SAs, one for inbound and one for outbound traffic. These IPsec states are

[35] Linux uses the "ip xfrm" command, FreeBSD uses the "setkey" command, and OpenBSD uses the "ipsecctl" command.

contained in the Security Association Database (SAD). Figure 12 shows an example of an IPsec SA using an AEAD algorithm.

```
src 198.51.100.1 dst 203.0.113.1
    proto esp spi 0xba293cd3(3123264723) reqid 1(0x01) mode tunnel
    replay-window 32 seq 0x00000000 flag af-unspec (0x00100000)
    aead rfc4106(gcm(aes)) 0x2ee20e32be3017c1878b9ae514081ba1d[…] 128
    anti-replay context: seq 0x148a3, oseq 0x0, bitmap 0xffffffff
    lifetime config:
      limit: soft (INF)(bytes), hard (INF)(bytes)
      limit: soft (INF)(packets), hard (INF)(packets)
      expire add: soft 0(sec), hard 0(sec)
      expire use: soft 0(sec), hard 0(sec)
    lifetime current:
      102600783(bytes), 84090(packets)
      add 2019-01-06 21:57:45 use 2019-01-06 21:57:50
    stats:
      replay-window 0 replay 0 failed 0

src 203.0.113.1 dst 198.51.100.1
    proto esp spi 0x6273ec0a(1651764234) reqid 1(0x01) mode tunnel
    replay-window 32 seq 0x00000000 flag af-unspec (0x00100000)
    aead rfc4106(gcm(aes)) 0x0afaf19501d6d94174bb3036b84d59d78e[…] 128
    anti-replay context: seq 0x0, oseq 0x7829, bitmap 0x00000000
    lifetime config:
      limit: soft (INF)(bytes), hard (INF)(bytes)
      limit: soft (INF)(packets), hard (INF)(packets)
      expire add: soft 0(sec), hard 0(sec)
      expire use: soft 0(sec), hard 0(sec)
    lifetime current:
      2422796(bytes), 30761(packets)
      add 2019-01-06 21:57:45 use 2019-01-06 21:57:50
    stats:
       replay-window 0 replay 0 failed 0
```

Figure 12: Example of an ESP IPsec SA (Inbound and Outbound) Using an AEAD Algorithm on Linux

If a non-AEAD algorithm is used, such as AES-CBC with HMAC-SHA-1, the SA will contain the encryption and integrity keys separately. Figure 13 illustrates this. Note that this example uses FreeBSD, which calls the AES algorithm by its original candidate name, Rijndael.

```
2001:db8:1:2::23 2001:db8:1:2::45
        esp mode=tunnel spi=1675186937(0x63d952f9) reqid=1(0x00000001)
        E: rijndael-cbc  1dd058ed 63905223 147979df 1865bfb3
        A: hmac-sha1  fde84c78 b2c90386 600927e3 1eb3dcf8 3163d053
        seq=0x00000000 replay=0 flags=0x00000000 state=mature
        created: Feb  2 17:29:42 2019   current: Feb  2 17:37:19 2019
        diff: 457(s)    hard: 3600(s)   soft: 2960(s)
        last:                           hard: 0(s)      soft: 0(s)
        current: 0(bytes)       hard: 0(bytes)  soft: 0(bytes)
        allocated: 0    hard: 0 soft: 0
        sadb_seq=1 pid=1404 refcnt=1
2001:db8:1:2::45 2001:db8:1:2::23
        esp mode=tunnel spi=3301523791(0xc4c9414f) reqid=1(0x00000001)
        E: rijndael-cbc  d32b7287 8e0ef003 3a2bac01 4b14d0c7
        A: hmac-sha1  1a3b1fc7 091e76f5 860456f2 5342ceaa bc33a3d3
        seq=0x00000000 replay=4 flags=0x00000000 state=mature
        created: Feb  2 17:29:42 2019   current: Feb  2 17:37:19 2019
        diff: 457(s)    hard: 3600(s)   soft: 2611(s)
        last:                           hard: 0(s)      soft: 0(s)
        current: 0(bytes)       hard: 0(bytes)  soft: 0(bytes)
        allocated: 0    hard: 0 soft: 0
        sadb_seq=0 pid=1404 refcnt=1
```

Figure 13: Example of an ESP IPsec SA Using a Non-AEAD Algorithm on FreeBSD

The IPsec SA state information consists of:

- The SPI that uniquely identifies the IPsec SA
- The IP addresses of the local and remote host that send and receive IPsec packets
- Cryptographic algorithms and their key material for encryption and integrity
- A link to the associated Security Policy (sometimes called reqid)
- The mode (tunnel or transport)
- The encapsulation state (transport protocol, port numbers, and optional framing)
- The current and maximum byte and packet counters allowed
- The current and maximum timers for idleness and age allowed
- Anti-replay context such as the current sequence number
- A link to the IPComp state if present
- Flags indicating various properties (TFC padding, etc.)

The maximum counters and lifetimes have a soft and hard value. When the soft value is reached, the kernel will notify the IKE daemon so it can take preventative action. When the hard value is reached, the IPsec SA is deleted by the kernel, and the IKE daemon is notified. Each time a packet is encrypted or decrypted, this state is updated appropriately.

5.1.2 The Security Policy Database (SPD)

The kernel maintains a list of IPsec policies in the Security Policy Database (SPD). The policy describes the nature of the traffic that matches a policy rule, and links it to the state used to encrypt or decrypt the packet. Policies without states are used for on-demand IPsec connections. Figure 14 shows examples of two policies corresponding to the SAs in Figure 12.

```
src 192.168.13.6/32 dst 0.0.0.0/0
    dir out priority 1040383 ptype main
    tmpl src 198.51.100.1 dst 203.0.113.1
        proto esp reqid 1 mode tunnel

src 0.0.0.0/0 dst 192.168.13.6/32
    dir in priority 1040383 ptype main
    tmpl src 203.0.113.1 dst 198.51.100.1
        proto esp reqid 1 mode tunnel
```

Figure 14: Examples of Policies Corresponding to Figure 12 on Linux

The IPsec Security Policy information consists of:

- The IP addresses of the IPsec gateways
- The source IP addresses allowed in classless inter-domain routing (CIDR) format
- The destination IP addresses in CIDR format
- The transport protocol covered (0 for all)
- The source and destination port ranges (0 for all)[36]
- A link to the associated SA state
- Direction (inbound, outbound, or forward[37])
- Priority of the policy compared to other policy rules
- IPsec protocol (ESP, AH, IPComp)
- Mode (transport or tunnel)
- IPComp information

Using the SPD and SAD, packets are processed for encryption and decryption, and all the security policies are applied. If a policy violation is detected, the packet is dropped—for example, when an encrypted packet is decrypted into a packet with a source address that is not allowed by the Security Policy of the SA.[38] A policy can also point to a non-IPsec SA target. Commonly implemented targets are PASS (never encrypt with IPsec), DROP, REJECT (DROP and send an ICMP message), and HOLD (cache the packet until an IPsec SA has been established).

Looking at the SAD and SPD entries of the previous figures, it can be seen that the host with IP address 198.51.100.1 is allowed to send ESP packets to the host with IP 203.0.113.1. The encrypted IP packet included can only have the source IP address 192.168.13.6 but can have any destination IP address. It is using AES-GCM as the AEAD encryption algorithm. In other words,

---

[36] For protocols without ports but with types, such as ICMP, the types are encoded as port numbers.
[37] Not all IPsec implementations have a forward policy. Think of it as a firewall within the IPsec subsystem.
[38] The SAD and SPD can be seen using the “ip xfrm” command on Linux. On BSD systems, the “setkey” tool can be used.

there is a VPN client running on 198.51.100.1 that started a VPN connection to the VPN server on 203.0.113.1 and received the internal IP address 192.168.13.6.

The IP address family of the IPsec host does not need to match the IP address family of the included encrypted IP packets. Figure 15 shows policies for two IPsec gateways using IPv6 addresses that are used to connect two IPv4 subnets with each other.

```
src 192.0.0.0/24 dst 192.0.2.0/24
    dir out priority 1042407 ptype main
    tmpl src 2001:db8:1:2::45 dst 2001:db8:1:2::23
        proto esp reqid 16389 mode tunnel

src 192.0.2.0/24 dst 192.0.0.0/24
    dir in priority 1042407 ptype main
    tmpl src 2001:db8:1:2::23 dst 2001:db8:1:2::45
        proto esp reqid 16389 mode tunnel

```

Figure 15: Example of IPsec Policies for a Gateway Architecture Connecting IPv4 Subnets using IPv6 on Linux

The output of the commands to inspect the current SAD and SPD differs per vendor. Figure 16 shows the SAD and SPD entries for an IPv6 in IPv4 IPsec connection in tunnel mode using the `ipsecctl` command on OpenBSD.

```
FLOWS:
flow esp in from 2001:db8:0:1::/64 to 2001:db8:0:2::/64
  peer 203.0.113.1 srcid FQDN/east dstid FQDN/west type use
flow esp out from 2001:db8:0:2::/64 to 2001:db8:0:1::/64
  peer 203.0.113.1 srcid FQDN/east dstid FQDN/west type require

SAD:
esp tunnel from 198.51.100.1 to 203.0.113.1 spi 0x03f86d3a
 auth hmac-sha2-256 enc aes-256
esp tunnel from 203.0.113.1 to 198.51.100.1 spi 0x4df47d50
 auth hmac-sha2-256 enc aes-256
```

Figure 16: Example of IPsec States and Policies Connecting IPv6 Subnets using IPv4 on OpenBSD (line breaks added)

5.1.3 SAD Message Types

Regardless of the implementation, the following types of messages are sent between the IKE and IPsec subsystems:

- IKE to IPsec:
    - Add, update, or remove an IPsec SA State
    - Add, update, or remove an IPsec SA Policy
    - Get IPsec SA information (byte counters, idleness)
    - Request a list of supported IPsec cryptographic algorithms

- IPsec to IKE:
    - Packet notification (with source/destination packet header information)
    - Invalid SPI notification (IPsec packet received without matching SA with SPI)
    - IPsec SA deleted (due to max life or max counter)

5.2 Example of Establishing an IPsec Connection Using IKE

In this example, the goal is to establish an IPsec connection that provides encryption and integrity protection services between endpoints A and B. The IPsec architecture is gateway-to-gateway; endpoint A uses gateway A on network A, and endpoint B uses gateway B on network B. If an IKE SA is not already in place, a packet will trigger the establishment of an IKE SA. In IKEv2, this is accompanied by the establishment of an IPsec SA as well:

1. Endpoint A creates and sends a regular (non-IPsec) packet that has a destination address of endpoint B.
2. Network A routes the packet to gateway A.
3. Gateway A matches the packet's characteristics against those in its SPD. It determines that the packet should be protected by encryption and integrity protection through ESP. Because the SPD entry does not have a pointer to the SAD, it knows that no IPsec SA is currently established.
4. Gateway A initiates an IKE SA negotiation with Gateway B. At the end of the negotiation, the IKE SA has been established, along with all the parameters and keying material required for the IPsec SA.
5. The parameters specify that ESP tunnel mode will be used and that it will provide encryption and integrity protection. A pair of unidirectional IPsec SAs is created for the ESP tunnel and added to the SAD. The IPsec SAs are attached to the SPD entries. Each SA provides protection only for traffic going in one direction.
6. Gateway A can finish processing the packet sent by endpoint A in step 1.
7. Gateway A modifies the packet so that it is protected in accordance with the SA parameters. It creates a new IP header that uses gateway A's IP address as the source IP address, and gateway B's IP address as the destination IP address. It sets the IP protocol to ESP and fills in the SPI number. It encrypts the original IP packet and includes this as the payload for this packet based on the encryption key of the SAD entry. It calculates and adds the integrity ICV to the ESP payload data based on the integrity key (or AEAD encryption key) of the SAD entry. Gateway A then sends the packet to Gateway B.
8. Meanwhile, Gateway B has also installed the IPsec SAs along with the SPD rules.
9. Gateway B receives the packet and uses the value in the unencrypted SPI field from the ESP header to determine which SA should be applied to the packet. After looking up the SA parameters (including the secret key(s) needed for integrity protection and decryption), gateway B decrypts and validates the packet. This includes removing the additional IP packet header, checking the integrity of the encrypted data, optionally

performing a replay check, and decrypting the original payload. Gateway B checks the SPD entry associated with the SAD entry to ensure that the decrypted IP packet complies to any source or destination restrictions, then sends the packet to its actual destination, endpoint B.

If endpoint B wishes to reply to the packet, steps 6 to 9 of this process are repeated, except the parties are switched. Endpoint B would send a packet to endpoint A; routing would direct it to gateway B. Gateway B would modify the packet appropriately and send it to gateway A. Gateway A would process and validate the packet to restore the original IP address, then send the packet to endpoint A.

Assuming that the IPsec connection between the gateways is sustained, eventually the IKE or IPsec SAs will approach one of the SA lifetime thresholds (maximum time or maximum bytes transmitted) as determined by the local policy on the respective gateways. The gateway with the shortest lifetime determines first that the maximum SA lifetime is approaching and initiates the rekeying process using the existing IKE SA. If the IPsec SA is being rekeyed, both ends install the new inbound and outbound IPsec SA before removing the old inbound and outbound IPsec SA. Once valid encrypted traffic is received on the new inbound IPsec SA, the old inbound IPsec SA will be deleted. This ensures that there is no interruption of the traffic flow during IPsec SA rekeying. If the IKE SA is being rekeyed, both ends replace the IKE SA, and all IPsec SAs belonging to the old IKE SA are attached to the new IKE SA.

5.3 Procurement Considerations for IPsec Products

IPsec VPN products vary in functionality, including protocol and algorithm support. They also vary in breadth, depth, and completeness of features and security services. Management features such as status reporting, logging, and auditing should provide adequate capabilities for the organization to effectively operate and manage the IPsec VPN and to extract detailed usage information. In the case of mesh encryption, too much logging can also be a concern. Traditionally, the management of IPsec products from different vendors has been problematic. Some recommendations and considerations include the following:

- Ensure that the cryptographic and networking capacity can accommodate the expected number of hosts and throughput.
- The Simple Network Management Protocol (SNMP) only provides a rudimentary and outdated interface for IKE and IPsec management. The IETF is working on a replacement management protocol using the YANG [52] data model language with ZEROCONF[39], which should provide a non-proprietary management interface that can be used across all vendors.
- AEAD algorithms such as AES-GCM for IPsec (ESP) significantly improve the performance of any IPsec product.

[39] A good history and summary of ZEROCONF can be found at http://www.zeroconf.org/.

- The IPsec VPN high availability, scalability, and redirection features should support the organization's requirements for automatic failover, where a secondary IPsec server is used as a spare that will automatically take over the IPsec services of a failing IPsec primary server. Or alternatively, support a deployment scenario where two IPsec servers perform load balancing for one logical IPsec service. State and information sharing are recommended to keep the IPsec server deployment process transparent to the user.
- IPsec VPN authentication should provide the necessary support for the organization's current and future authentication methods and leverage existing authentication databases. IPsec VPN authentication should also be tested to ensure interoperability with existing authentication methods. For remote access VPNs, support for EAP-TLS is an important consideration. For host-to-host and mesh encryption deployments, public key and certificate-based authentication is important.
- IPsec support within virtual machines or containers is usually provided by the operating system or container technology. This may require a different management system from physical IPsec gateway products. IPsec hardware offload needs careful consideration to ensure that the hardware offload capability is available within the virtualization technology without a performance penalty. In multi-tenant virtualization deployments, it might not be appropriate to use the hardware acceleration support, and support to disable hardware support should be available.
- Many IoT devices are severely resource constrained, requiring a very small footprint of supported algorithms and random-access memory (RAM) usage. These devices tend to not support certificate authentication, and usually support one or a few encryption and integrity algorithms, such as only AES-CCM. IPsec gateways that will be used to connect IoT devices should be selected carefully to ensure algorithm compatibility.
- IPsec products should be evaluated to ensure that they provide the level of granularity needed for access controls. Access controls should be capable of applying permissions to users, groups, and resources, as well as integrating with endpoint security controls. These considerations vary depending on the architecture that the IPsec product will be used for. Remote access VPNs need granularity at the user or device level, whereas host-to-host deployments could require access controls based on the IP address before accepting a connection based on proof of identity to prevent exposure to denial of service attacks.

6 Troubleshooting IPsec VPNs

This section provides information on troubleshooting IPsec VPNs.

6.1 IKE Policy Exceptions

A few IKE and IPsec interactions need some careful attention to prevent the two subsystems from interfering with each other. Usually these are handled by the IKE implementation. If an IPsec implementation insisted that all communication between two hosts be encrypted with IPsec, those two hosts would never be able to send non-IPsec packets, including IKE packets. And without allowing IKE packets, no IPsec SA can be negotiated and installed, and the two hosts would never be able to communicate. Similarly, if one host crashes and restarts, it needs to be able to send IKE packets that are not IPsec encrypted, yet the remote endpoint still has a policy that only allows encrypted traffic to be received.

To work around this, IPsec implements a policy exception for UDP port 500 and 4500 packets and will skip processing these via the regular SPD processing. If the kernel does not override IKE packets for IPsec processing, the IKE daemon needs to have a policy specifically for the IKE ports used with the highest preference, higher than the IPsec SA processing policy preference. Besides UDP port 500 and 4500, if TCP is used, those ports also need to have such a policy exception. Practically all IKE daemons perform this task on startup.

6.2 IPv6 Neighbor Discovery Policy Exception

A more subtle requirement is the need to exclude IPv6 neighbor discovery. If two hosts in the same subnet have established an IPsec SA over IPv6, and one of these hosts crashes and reboots, that host will send an unencrypted neighbor host discovery ICMP packet in an attempt to find the other host on the local network. If the host that did not crash drops the unencrypted ICMP packet, the two hosts will not be able to set up a new IPsec SA. If the host that did not crash performs DPD, it might find out in a few minutes that it needs to renegotiate the IPsec SA, otherwise communication will be blocked until the IPsec SA rekey or expiry timer runs out. This could be an outage that lasts anywhere between one and eight hours. Unfortunately, not all IKE daemons and IPsec implementations install the IPv6 neighbor discovery policy exception. It is recommended to test this scenario when using a new IKE/IPsec implementation.[40]

If a kernel receives a packet with an SPI for which it has no IPsec SA, it can send a message to the IKE process containing the IP address of the host that sent the IPsec packet. Such an IKE process may be able to recognize the peer based on its (static) IP address, and initiate a new IKE exchange to try and set up a new IPsec SA that replaces the obsoleted IPsec SA on the host that did not crash. Not all kernels implement this mechanism to inform the IKE process.

---

[40] To emulate, rather than actually crash a host, it is enough to send the IKE daemon a KILL signal, preventing it from telling the other side that it is shutting down, and then restart the IKE service.

6.3 Debugging IKE Configurations

The method for debugging IKE and IPsec configurations depends on the specific implementation. For new configurations that are not working properly, the first step should be for both endpoint administrators to verify the configuration options they believe they have agreed upon. A checklist with the most common options to check can be found in Appendix A. A mismatch between basic IKE or IPsec parameters is most often the cause for new IPsec configurations not establishing properly.

Using a network monitoring tool such as tcpdump is not very useful because only information from the first IKE_SA_INIT exchange can be inspected, and it only contains the DH groups, so it is unlikely that a misconfiguration can be detected at this point. All further captured IKE packets are encrypted, so they will not provide any additional information to diagnose the problem. It will be more helpful to enable additional logging or debugging. Remember to disable these settings again after the problem is resolved, otherwise large amounts of logs will continuously be produced.

If an administrator controls both endpoints that will be configured for IPsec, it is often the case that this administrator is sitting behind one of the gateways and is using a secure remote login tool, such as a web interface or SSH connection, to configure the remote endpoint. If a configuration mistake is made or a partial configuration is accidentally activated, the IPsec hosts will drop all non-IPsec traffic and lock out the administrator's remote session. To prevent this problem, use a third host to indirectly log in to the remote IPsec endpoint for configuration.

6.4 Common Configuration Mistakes

The HMAC integrity algorithm may be implemented with three different hash functions: SHA-256, SHA-384, and SHA-512. Different implementations use a different hash function for the "SHA2" indication that does not specify a specific hash function.

Care should be taken with sending DPD/liveness probes too often. If the remote client is a device that might enter sleep mode, it may not be able to respond to such probes. Another issue is when the device's link is congested while the IPsec connection is idle. This will trigger DPD/liveness probes that could be dropped due to traffic congestion. If repeatedly dropped, these packets will trigger a false positive warning about the remote IPsec endpoint connection being lost, causing the server to terminate the IKE and IPsec SA, resulting in more packets to re-establish the VPN on an already congested link. Do not set DPD/liveness probes to values under one minute, which matches the recommendation in [18].

PFS and DH group negotiation issues can be tricky to diagnose. In IKEv2, the first IPsec SA is established with the IKE SA establishment, and it does not really use a separate DH key exchange for PFS (unlike IKEv1). Any mismatch in DH group will only become apparent during a rekey message exchange hours later.

VPN gateways commonly are also used as NAT devices. If packets from the internal network are NAT'ed to the VPN server's public IP before being considered for IPsec protection, the source

IP no longer matches the IPsec policy, and the packet will not be sent out via IPsec. Instead, it could leak onto the internet without encryption, or be caught by the firewall subsystem running on the VPN gateway.

In an IPv4-based network, machines within the same subnet use the Address Resolution Protocol (ARP) to find the Ethernet address belonging to a local IP address. If remote access clients are being assigned IP addresses from the remote LAN, the VPN server needs to be configured to answer for all IP addresses that are reachable via the IPsec VPN, since those remote VPN clients do not receive the local network ARP requests. This service is often called *proxy ARP*. Some IPsec implementations detect this automatically. For IPv6, this process is handled via IPv6 neighbor discovery, which would also need to be performed by the VPN server if the local IPv6 range would be used for remote access clients.

The responder authenticates the initiator first, and fully establishes the IPsec SA before the initiator receives the IKE_AUTH response packet. If the initiator determines that the responder failed to authenticate itself, it can only notify the responder of this by immediately deleting the IKE SA, as the responder believes this is a fully established IKE SA and IPsec SA. This sometimes confuses administrators when debugging a problem, because from the responder's point of view, this was a successful—but very short—IPsec connection.

6.5 Routing-Based VPNs Versus Policy-Based VPNs

IPsec implementations need to inspect packet streams to determine when a packet should be encrypted and when it should be transmitted unencrypted. One method is to use the routing table. If a route is pointing to a specific IPsec device, the IPsec implementation processes the packet based on its SPD/SAD rules. However, using routes can be fragile. Another subsystem could change the routing to accidentally or maliciously bypass the IPsec device, thus bypassing all encryption policies.

Another issue of routing-based policies is that administrators often use a single IPsec policy from all possible IPv4 addresses (0.0.0.0/0) to all possible IPv4 addresses (0.0.0.0/0). Once the tunnel is established, routing is used to determine which packets to send over the IPsec connection. If a remote branch extends its network to use another subnet, say, 192.0.2.0/24, the only change needed is for the local branch to add a route for that IP range into the IPsec device. Firewall rules to limit the subnets allowed are omitted to allow this easy type of deployment, but this introduces a security problem as well as a compatibility problem. If the routes into the IPsec devices on both ends do not match, traffic will be encrypted in one direction but not in the other. At best, the IPsec gateway expecting encrypted packets will drop the unencrypted packets, and network connectivity fails. Or worse, the IPsec gateway will mistakenly route the unencrypted (and possibly modified) packets onto its local network.

Policy-based VPNs covering only specific subnets and not every address (0.0.0.0/0) are a better solution and recommended over routing-based VPNs, despite the additional management overhead required. Depending on the implementation, policy-based VPNs can be a bit harder to debug, since it might not be obvious to the administrator where in the IP stack a packet is taken to be processed by the IPsec subsystem. This can lead to unexpected issues in hub-spoke deployments. For example, if a host with LAN IP address 10.0.2.1 and public IP 192.0.2.1 creates an IPsec tunnel to a remote host on IP 192.0.2.2 to cover traffic between 10.0.2.0/24 and 10.0.0.0/8, such an IPsec gateway might lose access to its own LAN, since a packet with destination 10.0.2.13 will be sent over the IPsec tunnel because it falls within the destination IPsec policy range of 10.0.0.0/8. Routing-based VPNs do not have this issue, as LAN packets do not pass through the routing table and instead find the target host to send the packet to via ARP.

One common implementation processes the packets for IPsec after the network monitoring hooks are consulted. This leads to debugging tools such as the tcpdump tool seeing the packet as leaving the host unencrypted, while in fact the packet is encrypted after it is shown to the network debugging tool.

6.6 Firewall Settings

The most common network issue when setting up IPsec is that a firewall on the VPN server or on the network is blocking the IKE ports, UDP 500 and 4500. If an IPsec connection works for simple ping commands, but not when an application is trying to use the IPsec connection, the cause is most likely due to broken path MTU discovery. While this problem is not directly related to IPsec, it is often triggered because of the extra overhead of the ESP header making each 1500-byte original packet larger than 1500 bytes after the ESP header is added. The ESP packets would fragment and, too often, some stateful router or firewall mistakenly drops these packets.

If the ESP packet contains a TCP packet, it can also cause problems with the Maximum Segment Size (MSS). For TCP to work properly, it needs to be able to send ICMP packets (Packet too big), but ICMP is often blocked. Some IPsec policies might only allow TCP packets and prohibit ICMP packets. This also commonly manifests itself as an administrator who can log in over the IPsec connection using the SSH protocol, but as soon as they try to actually use this session, their screen freezes. Decreasing the MTU of the IPsec interface can work around this issue. For TCP, a common workaround is to use TCP MSS clamping to the path MTU or to a fixed value (e.g., 1380).

7 IPsec Planning and Implementation

This section focuses on the planning and implementation of IPsec in an enterprise. As with any new technology deployment, IPsec planning and implementation should be addressed in a phased approach. A successful deployment of IPsec can be achieved by following a clear, step-by-step planning and implementation process. The use of a phased approach for deployment can minimize unforeseen issues and identify potential pitfalls early in the process. This model also allows for the incorporation of advances in new technology, as well as adapting IPsec to the ever-changing enterprise. This section explores each of the IPsec planning and implementation phases in depth, as follows:

1. **Identify Needs.** The first phase of the process involves identifying the need to protect network communications, determining which computers, networks, and data are part of the communications, and identifying related requirements (e.g., minimum performance). This phase also involves determining how that need can best be met (e.g., IPsec, TLS, SSH) and deciding where and how the security should be implemented.
2. **Design the Solution.** The second phase involves all facets of designing the IPsec solution. For simplicity, the design elements are grouped into four categories: architectural considerations, authentication methods, cryptography policy, and packet filters.
3. **Implement and Test a Prototype.** The next phase involves implementing and testing a prototype of the designed solution in a lab or test environment. The primary goals of the testing are to evaluate the functionality, performance, scalability, and security of the solution, and to identify any issues with the components, such as interoperability issues.
4. **Deploy the Solution.** Once the testing is completed and all issues are resolved, the next phase includes the gradual deployment of IPsec throughout the enterprise.
5. **Manage the Solution.** After the IPsec solution has been deployed, it is managed throughout its lifecycle. Management includes maintenance of the IPsec components and support for operational issues. The lifecycle process is repeated when enhancements or significant changes need to be incorporated into the solution.

Organizations should also implement other measures that support and complement IPsec implementations. These measures help to ensure that IPsec is implemented in an environment with the technical, management, and operational controls necessary to provide adequate security for the IPsec implementation. Examples of supporting measures are as follows:

- Establish and maintain control over all entry and exit points for the protected network, which helps to ensure its integrity.
- Ensure that all IPsec endpoints (gateways and hosts) are secured and maintained properly, which should reduce the risk of IPsec compromise or misuse.
- Revise organizational policies as needed to incorporate appropriate usage of the IPsec solution. Policies should provide the foundation for the planning and implementation of

IPsec. Appendix B contains an extensive discussion of IPsec-related policy considerations.

7.1 Identify Needs

The purpose of this phase is to identify the need to protect communications and determine how that need can best be met. The first step is to determine which communications need to be protected (e.g., all communications between two networks, certain applications involving a particular server). The next step is to determine what protection measures (e.g., providing confidentiality, assuring integrity, authenticating the source) are needed for each type of communication. It is also important to identify other general and application-specific requirements, such as performance, and to think about future needs. For example, if it is likely that other types of communications will need protection in a year, those needs should also be considered.

After identifying all the relevant needs, the organization should consider the possible technical solutions and select the one that best meets the identified needs. Although IPsec is typically a reasonable choice, other protocols such as TLS or SSH may be equally good or better in some cases. See Section 8 for descriptions of such protocols and guidance on when a particular protocol may be a viable alternative to IPsec. In some cases, IPsec is the only option—for example, if a gateway-to-gateway VPN is being established with a business partner that has already purchased and deployed an IPsec gateway for the connection. Another possibility is that the solution may need to support a protocol that is only provided by IPsec.

Assuming that IPsec is chosen as the solution's protocol, the Identify Needs phase should result in the following:

- Identification of all communications that need to be protected (e.g., servers, client hosts, networks, applications, data), and the protection that each type of communication needs (preferably encryption, integrity protection, and peer authentication)
- Selection of an IPsec architecture (e.g., gateway-to-gateway, remote access VPN, host-to-host, mesh encryption)
- Specification of performance requirements (normal and peak loads).

7.2 Design the Solution

Once the needs have been identified, and it has been determined that IPsec is the best solution, the next phase is to design a solution that meets the needs. This involves four major components, which are described in more detail in Sections 7.2.1 through 7.2.5:

- **Architecture.** Designing the architecture of the IPsec implementation includes host placement (for host-to-host architectures)[41] and gateway placement (for remote access

[41] In most cases, the hosts are already placed on the network; the architectural considerations are focused on identifying intermediate devices between the hosts, such as firewalls performing NAT.

and gateway-to-gateway architectures), IPsec client software selection (for host-to-host and remote access architectures), and host address space management considerations (for host-to-host and remote access architectures).

- **Cryptography for Authentication.** The IPsec implementation must have an authentication method selected, such as the use of a digital signature or PSK. Only NIST-approved methods and algorithms shall be used. See NIST SP 800-131A [47].
- **Cryptography for Key Exchange, Confidentiality and Integrity.** The algorithms for DH key exchange, encryption, and integrity protection must be selected, as well as the key lengths for algorithms that support multiple key lengths. Only NIST-approved methods and algorithms shall be used. See NIST SP 800-131A [47].
- **Packet Filter.** The packet filter determines which types of traffic should be permitted and which should be denied, and what protection and compression measures (if any) should be applied to each type of permitted traffic (e.g., ESP tunnel using AES for encryption and HMAC-SHA-256 for integrity protection; Lempel-Ziv-Stac (LZS) for compression).

The decisions made regarding cryptography and packet filters are all documented in the IPsec policy. In its simplest form, an IPsec policy is a set of rules that govern the use of the IPsec protocol. It specifies the data to secure and the security method to use to secure that data. An IPsec policy determines the type of traffic that is allowed through IPsec endpoints, and generally consists of a packet filter and a set of security parameters for traffic that matches the packet filter. Those parameters include the authentication and encryption scheme and tunnel settings. When communications occur, each packet filter can result in the establishment of one or more IPsec SAs that enable protected communications satisfying the security policy for that packet filter.

Other decisions should also be made during the design phase, such as setting IKE and IPsec SA lifetimes and identifying which DH group number is best. Besides meeting the organization's cryptographic requirements of NIST SP 800-131A [47] and FIPS 140 [13], design decisions should incorporate the organization's logging and data management strategies, incident response and recovery plans, resource replication and failover needs, and current and future network characteristics, such as the use of wireless, NAT, and IPv6. Section 7.2.6 covers these considerations and design decisions in more detail.

7.2.1 Architecture

The architecture of the IPsec implementation refers to the selection of devices and software to provide IPsec services and the placement of IPsec endpoints within the existing network infrastructure. These two considerations are often closely tied together; for example, a decision could be made to use the existing Internet firewall as the IPsec gateway. This section will explore three particular aspects of IPsec architecture: gateway placement, IPsec client software for hosts, and host address space management.

7.2.1.1 Gateway Placement

Due to the layered defense strategy used to protect enterprise networks, IPsec gateway placement is often a challenging task. As described later in this section, the gateway's placement has security, functionality, and performance implications. Also, the gateway's placement may have an effect on other network devices, such as firewalls, routers, and switches. Incorporating an IPsec gateway into a network architecture requires strong overall knowledge of the network and security policy. The following are major factors to consider for IPsec gateway placement:

- **Device Performance.** IPsec can be computationally intensive, primarily because of encryption and decryption. Providing IPsec services from another device (e.g., a firewall, router) may put too high of a load on the device during peak usage, causing service disruptions. A possible alternative is to offload the cryptographic operations to a specialized hardware device, such as a network card with built-in cryptographic functions. Organizations should also review their network architecture to determine if bottlenecks are likely to occur due to network devices (e.g., routers, firewalls) that cannot sustain the processing of peak volumes of network traffic that includes IPsec-encapsulated packets.[42] For remote access architectures, the choice of DH group is important because it is the most computationally demanding part of IKE.
- **Traffic Examination.** If IPsec-encrypted traffic passes through a firewall, the firewall cannot determine what protocols the packets' payloads contain, so it cannot filter the traffic based on those protocols. Intrusion detection systems encounter the same issue; they cannot examine encrypted traffic for attacks. However, it is generally recommended to design the IPsec architecture so that a firewall and intrusion detection software can examine the unencrypted traffic. Organizations most commonly address this by using their Internet firewalls as VPN gateways or placing VPN gateway devices just outside their Internet firewalls. A full mesh encryption bypasses all network-based firewalls and intrusion detection systems because those systems can only accept or reject the encrypted stream without being able to inspect the data that has been encrypted. This could mean a reduction of security. This is discussed in greater detail in [54].
- **Traffic Not Protected by IPsec.** Organizations should consider carefully the threats against network traffic after it has been processed by the receiving IPsec gateway and sent without IPsec protection across additional network segments. For example, an organization that wants to place its VPN gateway outside its Internet firewalls should ensure that the traffic passing between the IPsec gateway and the Internet firewalls has sufficient protection against breaches of confidentiality and integrity.
- **Gateway Outages.** The architecture should take into consideration the effects of IPsec gateway outages, including planned maintenance outages and unplanned outages caused by failures or attacks. For example, if the IPsec gateway is placed inline near the Internet connection point, meaning that all network traffic passes through it, a gateway failure could cause a loss of all Internet connectivity for the organization. Also, larger IPsec

---

[42] The network architecture review is also beneficial in identifying intermediate network devices that may need to be reconfigured to permit IPsec traffic to pass through.

implementations may use a gateway management server; a server failure could severely impact the management of all gateways. Generally, if the network is designed to be redundant, the IPsec gateways and management servers should also be designed to be redundant.

- **NAT.** NAT provides a mechanism to use private addresses on the internal network while using public addresses to connect to external networks. NAT can map each private address to a different public address, while the network address port translation (NAPT) variant of NAT can map many private addresses to a single public address, differentiating the original addresses by assigning different public address ports.[43] NAT is often used by enterprises, small offices, and residential users that do not want to pay for more IP addresses than necessary or wish to take advantage of the security benefits and flexibility of having private addresses assigned to internal hosts. Unfortunately, as described in Section 4, there are known incompatibilities between IPsec and NAT because NAT modifies the IP addresses in the packet, which directly violates the packet integrity assurance provided by IPsec. However, there are a few solutions to this issue, as follows:
    - **Perform NAT before applying IPsec.** This can be accomplished by arranging the devices in a particular order, or by using an IPsec gateway that also performs NAT. For example, the gateway can perform NAT first and then IPsec for outbound packets. This is sometimes required because an IPsec service provider with multiple customers cannot build tunnels to each customer using the same internal IP addresses, and thus requires their customers to use specific RFC 1918 [35] IP addresses.
    - **Use UDP or TCP encapsulation of ESP packets.** Encapsulation requires tunnel mode. Encapsulation adds a UDP or TCP header to each packet, which provides an IP address and UDP/TCP port that can be used by NAT (including NAPT). This removes conflicts between IPsec and NAT in most environments.[44] IKE negotiates the use of encapsulation. During the IKE initial exchanges, both endpoints perform NAT discovery to determine if NAT services are running between the two IPsec endpoints. NAT discovery involves each endpoint sending a hash of its original source address(es) and port to the other endpoint, which compares the original values to the actual values to determine if NAT was applied. IKE then moves its communications from UDP port 500 to port 4500 in order to avoid inadvertent interference from NAT devices that perform proprietary alterations of IPsec-related activity. Detection of NAT and the use of encapsulation can also cause the host behind the NAT device to send keepalive packets to the other endpoint, which should keep the NAPT port-to-address mapping from being lost. Although all IKEv2 implementations must support UDP

[43] Additional information on NAT and NAPT is available from [53].

[44] In some cases, either the network architecture or the type of traffic may require additional measures to allow IPsec traffic to negotiate NAT successfully. For example, protocols such as Session Initiation Protocol (SIP) for Voice over IP (VoIP) and File Transfer Protocol (FTP) have IP addresses embedded in the application data. Handling such traffic correctly in NAT environments may require the use of application layer gateways (ALGs).

encapsulation, TCP encapsulation is a recent addition that has not yet reached universal support in IPsec devices.

7.2.1.2 Third-Party IPsec Client Software for Hosts

In IPsec host-to-host and remote access architectures, each host must have an IPsec-compliant implementation installed and configured. Most operating systems on computers and mobile devices have built-in support for IPsec and only require configuration or an enterprise provisioning system that provides and installs the required configurations. However, some mobile devices or embedded devices do not have a built-in IPsec implementation. Also, some built-in clients might be lacking a feature required for a certain deployment or might not support an enterprise provisioning system. In such cases, a third-party client might need to be deployed instead. Third-party clients must be distributed and installed, then configured or provisioned.[45]

Features that may be of interest when evaluating IPsec client software include support for the following:

- IKEv2
- IKEv1 (if communicating to legacy equipment)
- IKEv2 fragmentation
- IKEv2 encapsulation (UDP, TCP, or TCP-TLS)
- IKEv2 PPK
- Particular encryption, integrity protection, and compression algorithms
- Particular authentication methods such as EAP-TLS, RSA, and ECDSA
- Multiple simultaneous tunnels[46]
- Authentication support for hardware tokens utilizing Open Authorization (OAuth), OTP, or Fast Identity Online (FIDO)
- Flexible X.509 certificates and optional IPsec Extended Key Usage (EKU) restrictions
- CRL and/or OCSP support
- Certificate uniform resource indicator (URI) and raw keys for embedded clients
- DNSSEC provisioning of enterprise trust anchors

Another important IPsec client feature is the ability to allow or prevent split tunneling. Split tunneling occurs when an IPsec client on an external network is not configured to send all its

---

[45] Organizations deploying third-party clients should pay particular attention to mobile devices and application stores. On some mobile phone platforms, many questionable VPN implementations are being made available where the goal of the VPN service is to monitor and/or modify the user's traffic before it is protected by IPsec.

[46] In some cases, it may be desirable to permit a host to establish multiple tunnels simultaneously. For example, the host may perform two types of communications that each need different protective measures from IPsec.

traffic to the organization's IPsec gateway. Requests with a destination on the organization's network are sent to the IPsec gateway, and all other requests are sent directly to their destination without going through the IPsec tunnel. The client host is effectively communicating directly and simultaneously with the organization's internal network and another network (typically the Internet). If the client host were compromised, a remote attacker could connect to the host surreptitiously and use its IPsec tunnel to gain unauthorized access to the organization's network. This would not be possible if the IPsec client software had been configured to prohibit split tunneling. However, any compromise of an IPsec client host is problematic, because an attacker could install utilities on the host that capture data, passwords, and other valuable information.

Prohibiting split tunneling can limit the potential impact of a compromise by preventing the attacker from taking advantage of the IPsec connection to enter the organization's network; the attacker could only connect to the compromised system when it is not using IPsec. However, many hosts have multiple methods of connectivity, such as mobile data, wired LAN, and wireless LAN; if an attacker can connect to a network interface other than the one used for IPsec, it may be possible to use the IPsec tunnel even if split tunneling is prohibited. This can allow access to a more trusted network—the network protected by IPsec—from a less trusted network, such as an improperly secured wireless LAN. Accordingly, hosts should support being configured so that only the network interface used for IPsec is enabled when IPsec is in use. Some VPN clients can be configured to disable other network interfaces automatically. An alternative is to configure a personal firewall on the host so that it blocks unnecessary and unauthorized network traffic on all interfaces. Due to its security complications/risks, split tunneling is strongly discouraged.

As described in Section 7.2.6, not allowing split tunneling is also helpful in preventing IPsec clients' hosts from being compromised. If a user mistakenly tries to connect to a malicious site, the traffic would be forced to go through the VPN where an enterprise firewall or proxy server could filter malicious traffic. Some organizations prefer split tunneling because it prevents non-enterprise traffic from reaching the enterprise. It also reduces the internet bandwidth capacity needed by the enterprise to support its remote VPN clients. There might also be legal reasons why an enterprise prefers not to handle traffic unrelated to its organization.

There are other factors that may differentiate IPsec clients. For example, one client may provide substantially better performance than another client or consume less of the host's resources. Another consideration is the security of the client software itself, such as how frequently vulnerabilities are identified, and how quickly patches are available. Client interoperability with other IPsec implementations is also a key concern; some client implementations only interoperate with their own vendor's gateway implementation or with a limited number of other vendors' gateway implementations. It is critical to ensure that the selected client will interoperate with each gateway implementation it might encounter. Section 7.3.1 discusses this topic in more detail.

Organizations should also carefully consider how clients can be provisioned with IPsec client software and configuration settings, including policies. Many clients offer different features that can make client deployment, configuration, and management easier. For example, an administrator might be able to set policy for clients remotely, instead of manually visiting each host. Some clients offer administrators the ability to lock out or disable certain configuration options or functionality so that users cannot inadvertently or intentionally circumvent the intended security. If administrators cannot distribute pre-configured IPsec clients or remotely control IPsec configuration settings, the administrators might need to manually configure each IPsec client or rely on users to follow instructions and configure the clients themselves. The latter approach is often challenging for non-technical users.

7.2.1.3 Host Address Space Management

In remote access VPN architectures where the hosts are outside the organization (e.g., mobile devices, remote workers), the VPN client will receive an additional IP address from the organization's address space assigned as a virtual IP address to each external IPsec host. In the latter case, the client then establishes an IPsec connection that uses its real IP address in the external packet headers (so the IPsec-encapsulated packets can be routed across public networks) and its virtual IP address in the internal packet headers (so the packets can be routed across the organization's internal networks and treated as internally generated).

Virtual addresses can be assigned from an address pool that resides on the VPN server. The VPN server can also use the DHCP Relay protocol or use an AAA service such as RADIUS or Diameter to obtain an IP address. A local pool can provide an easier indication that the IP address accessing a local resource is originating from a VPN client or is a client connecting from a certain region.

It is important to ensure that any addresses the IPsec gateway manages are excluded from the ranges that other internal DHCP servers can assign to avoid address conflicts. Some vendors provide internal address assignment and authentication using proprietary functionality. This may present compatibility issues depending on the products being used.

When deploying a remote access VPN in a data center or cloud where the only service offered is the VPN server without any other local resources, non-routable IP addresses such as those defined in RFC 1918 [35] can be used for the address pool of virtual IPs for the VPN clients. The VPN server then uses NAT to translate these IP addresses to its own public IP address. One potential issue with such a deployment is that some websites limit the number of users or connections coming from a single IP address. If dozens or hundreds of website users appear to all come from the one VPN server public IP address, the website might block the IP address because it assumes it is a malicious entity that obtained the credentials of many users. Using multiple public IP addresses on such a VPN server deployment could mitigate this problem.

7.2.2 IKE Authentication

The endpoints of a host-to-host and gateway-to-gateway IPsec architecture typically use the same authentication method to validate each other. Validation for remote access VPNs tend to use different mechanisms to authenticate each other, where the server is authenticated using a machine certificate and clients are authenticated using EAP-TLS.

IPsec implementations typically support a number of authentication methods. The most common methods are certificate-based digital signatures or raw public keys, EAP, and PSK. When using IKEv1, a group PSK combined with a username and password is also common. This section discusses the primary advantages and disadvantages of these methods.

PSKs should only be used for gateway-to-gateway scenarios that cross an administrative domain and only when based on generating strong and sufficiently long random PSKs with at least 112 bits of entropy. Using a public-key key pair (with or without certificates) based on RSA, DSA or ECDSA is preferred over using PSKs, but if the implementations that need to interoperate do not share the same public key-based authentication method, PSKs are an appropriate alternative. Within an administrative domain, PSKs should not be used. For remote access VPN scenarios, EAP-TLS or machine certificate authentication should be used.

7.2.2.1 PSKs

To use PSKs, the IPsec administrator needs to create a strong random secret key or password string that is then configured in both IPsec devices (the end points) of an IPsec connection.[47] PSKs are the simplest authentication method to implement, but also by far the least secure. Administrators need to find IPsec products that provide key management capabilities for PSKs or implement their own key management mechanisms, such as generating, storing, deploying, auditing, and destroying keys; proper key management can be quite resource-intensive. Although it is easiest to create a single key that all endpoints share, this causes problems when a host should no longer have access—the key then needs to be changed on all other hosts. PSKs should also be updated periodically to reduce the potential impact of a compromised key. Another issue is that the key must be kept secret and transferred over secure channels. Individuals with access to an endpoint are almost always able to gain access to the PSK.[48] Depending on the key type, this could grant access from one, some, or all IP addresses. (A group shared key can only be used from addresses in a certain range, while a wildcard shared key can be used from any IP address.) Also, using the same key for a group of endpoints reduces accountability, as anyone within the group can impersonate another member of the group.

Because of scalability and security concerns, PSK authentication is generally an acceptable solution only for small-scale implementations with known IP addresses or small IP address ranges. The use of a single PSK for a group of hosts is strongly discouraged for all but the most highly-controlled environments, such as a group of secure routers. PSKs are also generally not recommended for remote access clients that have dynamic IP addresses, because the keys cannot be restricted to a particular IP address or small range of IP addresses. PSKs are also frequently used during initial IPsec testing and implementation because of their simplicity. After the IPsec implementation is operating properly, the authentication method can then be changed.

---

[47] Because PSKs are often long strings of random characters, manually typing them in to the endpoints can cause problems from typos.

[48] Some vendors protect stored PSKs using obfuscation, but since unattended access to these secrets is needed when booting up the system, this obfuscation is usually trivially broken.

7.2.2.2 Certificate-based digital signatures

Certificates are typically used in machine certificate and EAP-TLS based authentication. The certificate owner produces a digital signature of the IKE exchange that proves its possession of the certificate's private key and authenticates the IKE session.

A certificate identifies each device, and each device is configured to use certificates. User-specific certificates may be used instead of device-specific certificates, but some remote access VPN configurations do not allow a single user to log onto multiple devices simultaneously, so it is always better to generate a certificate per device rather than per user.

Two IPsec endpoints will trust each other if a CA they both trust has signed their certificates.[49] The certificates must be securely stored in the local certificate store on the IPsec hosts and gateways or on a secure hardware token. Using a certificate-based method allows much of the key administration to be offloaded to a central certificate server, but still requires IPsec administrators to perform some key management activities, such as provisioning hosts with credentials, either through IPsec vendor-provided features or IPsec administrator-created capabilities. Many organizations implement a public key infrastructure (PKI) for managing certificates for IPsec VPNs and other applications such as secure email and Web access.[50] Certificates can be issued to limit their use using EKU attributes. Some IPsec hosts insist on IPsec-specific EKUs, while others accept the TLS-based EKUs (serverAuth or clientAuth) and some ignore all EKUs. The IETF PKI standard for IKE EKUs is specified in RFC 4945 [55]. A certificate issued for secure email might not be usable for IPsec on some of the VPN gateways deployed in an organization. Issuing certificates per device instead of per user avoids this issue and has the additional advantage that if a device is lost or stolen, not all of the user's VPN access will need to be revoked.

Although the certificate authentication method scales well to large implementations and provides a much stronger security solution than PSKs, it does have some disadvantages. While certificates can be revoked and transmitted to the VPN servers via CRLs [57] in bulk, or on demand via OCSP) [58], typically these mechanisms provide no option for temporarily disabling a certificate. Additional complications can occur when the connection to the OCSP server itself is down, or worse, requires an IPsec tunnel to be negotiated that needs to use that OCSP server. Non-standard solutions using an AAA server or a Pluggable Authentication Module (pam authentication) are usually added for such use cases.

Another potential problem with the certificate authentication method involves packet fragmentation. Packets in an IKE negotiation are typically relatively small and do not need to be fragmented. By adding certificates to the negotiation, packets may become so large that they need to be fragmented, which is not supported by some IPsec implementations.

---

[49] This describes the most common CA model; other models, such as the Federal Bridge CA, function somewhat differently.

[50] PKI implementations require a considerable investment in time and resources. It is outside the scope of this document to discuss a PKI in detail. See NIST SP 800-32, *Introduction to Public Key Technology and the Federal PKI Infrastructure*, for more information [56].

7.2.2.3 Raw public key digital signatures

Raw public key digital signatures work the same as certificate-based digital signatures, except instead of trusting a certificate (directly or indirectly via a CA), the trust is placed in the public key itself. Keys are usually represented in base64 format or using just the SubjectPublicKeyInfo (SPKI) part of a certificate.

Public keys can be distributed to the endpoints via trusted provisioning software or can be fetched on demand from DNSSEC or a directory service (e.g., Lightweight Directory Access Protocol [LDAP]) based on the ID presented during the IKE exchange. Instead of specifying the validity period in a certificate, these publishing services can simply remove the key when it is no longer needed. The public key for a particular ID specified in IKE resides in the DNS or directory service under that ID name. Revocation is accomplished by removing the public key from the publishing service's database.

For resource-constrained embedded devices that authenticate using a single hard-coded public key, a certificate by itself can be too large to be contained or operated on and serves no purpose since certificate validation is not performed.

One disadvantage of raw public keys is that there are not as many tools that support these, because most IKE implementations have been written to be used with certificates or PSKs.

7.2.2.4 EAP

EAP support is included in IKEv2. Both older and newer EAP methods are supported. EAP can be used as the only authentication method, or as a second authentication method. Often, different authentication methods are used: the server is authenticated using certificate-based authentication, and the client (typically a laptop or mobile device) is authenticated using an EAP method. EAP authentication allows additional types of authentications to be used, such as a username with a password (EAP-MSCHAPv2), a user (not host) certificate (EAP-TLS), or an EAP method supporting two-factor authentication. EAP authentication is mostly used for laptops and mobile phones.

7.2.3 Cryptography for Confidentiality Protection, Integrity Protection and Key Exchange

Setting the cryptographic policy for confidentiality and integrity protection and key exchange involves choosing encryption and integrity protection algorithms, key lengths,[51] DH groups for key exchange, and IKE and ESP lifetimes. For up-to-date policies and advice on these settings, see NIST SP 800-131A [47] and FIPS 140 [13] as well as the recommendations of the IETF for IKE [20] and ESP [59]. Note that these documents will be updated over time or be obsoleted for newer publications.

[51] Only FIPS-validated implementations of NIST-approved algorithms shall be used.

The IKE protocol sends just a few packets per hour, so it makes sense to be extra cautious and pick strong algorithms with large enough keys, and specifically a strong DH group. Approved DH groups are identified in NIST SP 800-56A [62]. The bulk of the CPU power of an IPsec host will be spent on IPsec, not IKE. In IKE, the most CPU-intensive operation is the DH calculation. When an IPsec host has hundreds or thousands of IKE (re)connections, choosing the right DH group becomes very important.

It is recommended to use strong key sizes for IKE. The performance impact of larger key sizes is minimal because IKE traffic is negligible compared to IPsec traffic. For IPsec (ESP), the key size can have a significant impact on performance. In general, use larger key sizes for IPsec if performance is not an issue. For ESP, the choice of algorithms for confidentiality and integrity protection should also take performance into account. Using an AEAD algorithm such as AES-GCM that can provide both confidentiality and integrity protection in a single operation will give better performance than using non-AEAD algorithms that require separate operations (e.g., AES-CBC for encryption and HMAC for integrity protection). It is important to estimate the processing resources that the cryptographic computations will require during peak usage.

It is uncommon to use 192-bit AES keys, and this key length is optional in [20]. It is worth mentioning as well that in the future, an adversary with a quantum computer may be able to reduce the key strength of an AES key by a factor of two, in which case a 256-bit AES key may effectively provide around 128 bits of security in the quantum computer world (note that this level of security strength is a magnitude stronger than the current level of 128 bits for classical security).

AES-GCM (an AEAD algorithm) is often offloaded to hardware, making it significantly faster than AES-CBC (a non-AEAD algorithm). The CPU is typically the hardware component most affected by cryptographic operations. In some cases, a hardware-based cryptographic engine with customized CPUs, also known as a cryptographic accelerator, may be needed for greater throughput, but this may limit the algorithm options. Another potential issue is export restrictions involving the use of encryption algorithms in certain countries.[52] In addition, some IPsec components may not provide support for a particular algorithm or key size.

For integrity checking of non-AEAD algorithms, most IPsec implementations offer HMAC-SHA-1 or the HMAC with the SHA-2 hashing algorithms[53] (referred to as the HMAC-SHA2s). Even though HMAC-SHA1 is still a NIST-approved option, the HMAC-SHA2s are recommended due to the fact that the HMAC-SHA2s have stronger security than HMAC-SHA1. HMAC-MD5 has never been a NIST-approved algorithm and shall not be used.

In some implementations of IPsec, the cryptographic policy settings are not immediately apparent to administrators. The default settings for encryption and integrity protection, as well as the details of each setting, are often located down several levels of menus or are split among multiple locations. It is also challenging with some implementations to alter the settings once

[52] More information on export restrictions is available from the Bureau of Industry and Security, U.S. Department of Commerce, at https://www.bis.doc.gov/index.php/policy-guidance/encryption.

[53] HMAC-SHA256, HMAC-384 or HMAC-SHA-512.

they have been located. For example, by having portions of the settings in multiple locations, administrators may need to go back and forth between different configuration screens to ensure that the settings are correct and consistent.

7.2.4 High Speed and Large Server Considerations

While network devices such as routers and firewalls will already be optimized for network performance, generic operating systems will require tuning for optimized network performance. Enough RAM should be made available to the network stack. CPU power saving and throttling should be disabled and, on non-uniform memory access (NUMA) systems, further optimizations might be possible. Check with the hardware vendor for specific instructions.

Network card settings can also have a large impact on throughput. Check that the network card's transmit queue (txqueuelen) is set large enough to accommodate the amount of traffic. Check the network card settings for TCP Segmentation Offload (TSO), Generic Segmentation Offload (GSO), checksum offloading, and virtual local area network (VLAN) settings. If using a network card with IPsec hardware acceleration support, follow the vendor's instructions on how to optimize the host.

When using virtualization, ensure that the virtualization layer is using as much direct hardware access as possible. For performance, it will be better to configure a hardware network card inside a virtual machine than to configure the virtual machine with a virtual network card. On some hardware, this needs to be enabled in the Basic Input/Output System (BIOS). For example, on Intel systems, ensure that Intel Virtualization Technology for Directed I/O (Intel VT-d) is enabled. Ensure that the virtualization is not emulating a slightly different CPU than the real hardware because it will not be able to use the hardware virtualization instructions of the CPU and instead will have to perform full emulation in software.[54]

Ideally, when not using IPsec, the system should be able to utilize line-speed unencrypted traffic. A popular network tool to perform network performance tests is *iperf*. Once the system is performing well without IPsec, IPsec can be enabled.

IPsec hosts that are busy will spend the bulk of their computational resources on encrypting and decrypting ESP traffic. The performance of the algorithms for IKE is less important, as there are far fewer IKE packets than ESP packets in most deployments of IPsec VPNs.

7.2.4.1 ESP performance considerations

If the host's CPU usage is the limiting factor, it is particularly important to use the right algorithms. Using an AEAD algorithm for encryption and integrity protection is much faster than using two non-AEAD algorithms. Likely the best algorithm choice will be AES-GCM because modern CPUs have hardware support for it. Both 256-bit and 128-bit AES keys currently

[54] This usually happens when a virtual machine configuration with a specific CPU sub-type is migrated to different hardware without the configuration being updated.

provide strong protection, so when CPU load becomes an issue, one could consider switching from 256-bit to 128-bit keys, provided that this is allowed by the deployment policy.

If the host is running a few high-speed IPsec SAs, it could be that multiple CPUs on the host are not utilized properly to spread the cryptographic load of a single IPsec SA over multiple CPUs. When multiple CPUs are used for a single IPsec SA, there will be an increase in out-of-order packets being sent, and the replay-window will need to be increased to accommodate this at both endpoints. IPsec replay-protection can be disabled to test if that is the limiting factor for the server performance. This is less of a concern on busy servers that act as a remote access VPN, since these will be serving many users' IPsec SAs per CPU. For high-speed IPsec SAs, it is also important to use ESNs to avoid excessive rekeying.

If the application is sending packets close to the MTU size, using ESP encryption (which adds a few bytes in size compared to the unencrypted packet size) might lead to fragmentation, which will reduce performance. If the IPsec SA is a connection within a data center or over a dedicated fiber cable, it might be possible to increase the MTU (e.g., to 9000 bytes) to prevent fragmentation. The MTU of the internal-facing network card can also be reduced to force the LAN to send packets that are smaller than 1500 bytes, so once the host encrypts the packet to send it out over the external interface, the ESP packet will not exceed an MTU of 1500 bytes. TCP MSS clamping can be used on both IPsec endpoints to ensure that TCP sessions will use a lower MTU that prevents fragmentation.

7.2.4.2 IKE performance considerations

While IKE performance in most cases does not matter, it does matter for remote access VPN servers that have a continuous stream of clients connecting and disconnecting. If IKE uses too much of the CPU resources, this will impact ESP processing times as well. If a remote access VPN server is too busy and has degraded to the point where an IKE session takes more than a few seconds to establish, the server will completely collapse under the load. IKE clients usually timeout after five to ten seconds and will start a new IKE attempt. This will put even more load on the already loaded server. That is, the load based on the number of IKE clients connecting will slowly go up until it hits a breaking point. If the IKE REDIRECT [38] extension is supported, the server can be configured to start redirecting clients to another server before it becomes too busy. See Section 3.8 for more information.

The most computationally expensive part of IKE is the DH calculation performed during a key exchange. DH implemented using ECP groups (elliptic curve group modulo a prime) take less resources than the use of finite field groups (modular exponential, or MODP groups) such as DH group 14. The DH 19, DH 20, and DH 21 ECP groups are also considered to be more secure [61]. DH groups 1, 2, 5, and 22 are not NIST-approved because these groups do not supply the minimum of 112 bits of security. See NIST SP 800-56A [62] for further information about approved DH groups.

MOBIKE should be enabled on remote access VPN servers. Mobile devices will switch between WiFi and mobile data, and without MOBIKE ,this requires a new IKE session for each network switch. This will increase the number of DH calculations that need to be supported. IKE clients on unreliable WiFi can end up restarting IKE many times. When MOBIKE is used, an encrypted informational exchange message is sent to modify the existing IKE and ESP sessions to use the new IP address of the other interface and avoid starting new sessions with new expensive DH group calculations.

Liveness[55] probes can be used by a server to detect remote clients that have vanished without sending a delete notification. The timer for these probes should not be set too short, or else the server will need to send frequent IKE packets with DPD probes for idle IKE clients. If the timeout value is set very short (in the order of a few seconds), there is the additional risk of IKE clients on unreliable networks not receiving the DPD probes. The server will disconnect the IKE client when a response to the probe is not returned. That client will experience packet loss and declare the IPsec connection dead. This will lead to the creation of another new IKE session and an increased load on the VPN server. In general, keeping a few IKE and IPsec states alive for vanished VPN clients is cheap. It takes very little memory and no CPU resources. A reasonable DPD timeout value is in the range of 10 to 60 minutes.

The IKE SA and IPsec SA lifetimes are not negotiated. Each endpoint decides when it wants to rekey or expire an existing SA. Using longer IKE SA and IPsec SA lifetimes can reduce the amount of IKE rekeying required. IKE rekeying and IPsec rekeying with PFS require a new DH calculation as well, so extending the IKE and IPsec lifetimes can help reduce the server load.

Another option on busy servers with many remote access users is to support IKE session resumption [63]. A mobile device that is going to sleep can send the server a sleep notification to prevent DPD-based disconnections. The server and client keep the cryptographic state of the IKE session. When the device wakes up, it can send an encrypted session resumption request. This avoids the need for a new IKE session with the expensive DH calculation to establish a new connection; the server is triggered via a DPD timeout to delete the IKE and IPsec SA if the sleep period exceeds the timeout period.

If a provisioning system is used to generate and install configurations for the IKE clients, optimized settings could be pushed automatically to all IKE clients to ensure optimal performance. This would avoid manual configurations that, when performed by inexperienced users, could result in less optimized settings because the user did not enable or disable certain features.

Enabling IKE debugging can cause a lot of logging data to be generated. That in itself can cause a significant performance impact on the system. Always check to see if debugging has accidentally been left enabled on systems experiencing a high work load.

7.2.4.3 IKE denial of service attack considerations

DDoS attacks are a separate issue of concern. Such attacks also put an additional load on the server, but the characteristics are different from a legitimate user load.

---

55 This was formerly called Dead Peer Detection (DPD).

An attack from an authenticated user with valid credentials is assumed to be a readily solvable problem—simply revoke such users' access to the VPN infrastructure. One exception to this is when anonymous IPsec is in use, because in that case, the connection cannot be terminated or prevented based on the user credentials. Vendors of IPsec equipment supporting anonymous IPsec connections should take countermeasures, for example by limiting the number of IPsec SA requests that are accepted or by limiting the number of rekeys or anonymous connections allowed based on an IP address.

IKEv2 has built-in protection against DDoS attacks, but IKEv1 does not. When the number of incomplete IKE sessions (sometimes called half-open IKE SAs) reaches a threshold, indicating a possible DDoS attack, IKEv2 can enable DDoS COOKIES. Each new IKE_SA_INIT request will be answered with a reply that only contains a COOKIE based on a local secret[56] and the client's IP address and port. The client will have to resend its original IKE_SA_INIT request with the COOKIE added to the request. The server can calculate the value of the COOKIE without needing to store any state in memory for the original IKE_SA_INIT request. The IKE server will only perform the expensive DH calculations after the client has retransmitted its IKE_SA_INIT packet with the COOKIE, proving to the server that the client was not simply a spoofed IP packet.

Additionally, IKEv1 can be coerced into an amplification attack. With IKEv1, the responder and initiator are each responsible for retransmission when a packet is lost. A malicious user can send a single spoofed IKEv1 packet to an IKEv1 server and cause that IKEv1 server to send several retransmit packets to the spoofed IP address. Some IKEv1 implementations defend against this by never responding more than once to an initial IKEv1 request, but this can break legitimate IKEv1 clients using Aggressive Mode when there is actual packet loss happening.

7.2.5 Packet Filter

The purpose of the packet filter is to specify how each type of incoming and outgoing traffic should be handled—whether the traffic should be permitted or denied (usually based on IP addresses, protocols, and ports), and how permitted traffic should be protected (if at all). By default, IPsec implementations typically provide protection for all traffic. In some cases, this may not be advisable for performance reasons. Encrypting traffic that does not need protection or is already protected (e.g., encrypted by another application) can be a significant waste of resources. For such traffic, the packet filter could specify the use of the null encryption algorithm for ESP, which would provide integrity checks and anti-replay protection, or the packet filter could simply pass along the traffic without any additional protection. One caveat is that the more complex the packet filter becomes, the more likely it is that a configuration error may occur, which could permit traffic to traverse networks without sufficient protection.

---

56 The secret is usually a random value refreshed every hour to prevent attackers from attempting to guess the secret by trying different possibilities until the correct value is found. The server needs to remember the current and previous secret and to perform two calculations so that clients caught at a secret refresh will not be locked out. [rephrase "caught"]

An issue related to packet filters is that certain types of traffic are incompatible with IPsec. For example, IPsec cannot negotiate security for multicast and broadcast traffic.[57] This means that some types of applications, such as multicast-based video conferencing, may not be compatible with IPsec. Attempting to use IPsec to secure such traffic often causes communication problems or impairs or breaks application functionality. Other traffic such as multicast DNS (mDNS) and DNS Service Discovery (DNS-SD) broadcast requests should not be forwarded to other networks because they have no meaning or relevance beyond the local network. For example, ICMP error messages are often generated by an intermediate host such as a router, not a tunnel endpoint; because the source IP address of the error message is the intermediate host's address, these ICMP packets do not have confidentiality or integrity protection, and the receiving host cannot make security policy decisions based on unprotected packets. Packet filters should be configured to not apply IPsec protection to types of traffic that are incompatible with IPsec—they should let the traffic pass through unprotected if that does not compromise security. If the IPsec gateway cannot block broadcasts and other traffic that should not be passed through it, it may also be effective to configure firewalls or routers near the IPsec gateway to block that particular type of traffic.

7.2.6 Other Design Considerations

A particularly important consideration in design decisions is the identification and implementation of other security controls. Organizations should have other security controls in place that support and complement the IPsec implementation. For example, organizations should configure packet filtering devices (e.g., firewalls, routers) to restrict direct access to IPsec gateways. Organizations should have policies in place regarding the acceptable usage of IPsec connections and software. Organizations may also set minimum security standards for IPsec endpoints, such as mandatory host hardening measures and patch levels, and specify security controls that must be employed by every endpoint.

For endpoints outside the organization's control, such as systems belonging to business partners, users' home computers, and public internet access networks, organizations should recognize that some of the endpoints might violate the organization's minimum security standards. For example, some of these external endpoints might be compromised by malware and other threats occasionally; malicious activity could then enter the organization's networks from the endpoints through their IPsec connections. To minimize risk, organizations should restrict the access provided to external endpoints as much as possible, and also ensure that policies, processes, and technologies are in place to detect and respond to suspicious activity. Organizations should be prepared to identify users or endpoint devices of interest and disable their IPsec access rapidly as needed.

IPsec packet filters can be helpful in limiting external IPsec endpoints' accesses to the organization. Using packet filters to limit acceptable traffic to the minimum necessary for untrusted hosts, along with other network security measures (e.g., firewall rulesets, router access control lists), should be effective in preventing certain types of malicious activity from reaching

[57] Section 10.1 contains information on current research efforts to create IPsec solutions for multicast traffic.

their targets. Administrators may also need to suspend access temporarily for infected hosts until appropriate host security measures (e.g., antivirus software update, patch deployment) have resolved the infection-related issues. Another option in some environments is automatically quarantining each remote host that establishes an IPsec connection, checking its host security control settings, and then deciding if it should be permitted to use the organization's networks and resources. It is advisable to perform these checks not only for hosts connecting to the organization's VPN from external locations, but also for mobile systems connecting to the organization's internal network that are also sometimes connected to external networks.

In addition to endpoint security, there are many other possible design considerations. The following items describe specific IPsec settings not addressed earlier in this section:

- **SA Lifetimes.** The IPsec endpoints should be configured with lifetimes that balance security and overhead.[58] In general, shorter SA lifetimes tend to support better security, but every SA creation involves additional overhead. In IKEv1, the appropriate lifetime is somewhat dependent on the authentication method—for example, a short lifetime may be disruptive to users in a remote access architecture that requires users to authenticate manually, but not disruptive in a gateway-to-gateway architecture with automatic authentication. IKEv2 also decouples rekeying from reauthentication, so rekeying can be performed more frequently without affecting the user. During testing, administrators should set short lifetimes (perhaps 5 to 10 minutes) so the rekeying process can be tested more quickly. In operational implementations, IPsec SA lifetimes should generally be set to a few hours, with IKE SA lifetimes set somewhat higher. A common default setting for IKE SAs is a lifetime of 24 hours (86400 seconds), and for IPsec SAs a lifetime of 8 hours (28800 seconds). It is important to ensure that the peers are configured with compatible lifetimes; some configurations will terminate an IKE negotiation if the peer uses a longer lifetime than its configured value. Some IKEv2 implementations, especially minimum IKEv2 implementations used with embedded devices, might not support the CREATE_CHILD_SA exchange, and therefore do not support rekeying without reauthentication.
- **IKE Version.** IKEv2 should be used instead of IKEv1 where possible. If using IKEv1, the aggressive mode (see RFC 2409 [94] for detail) should be avoided because it provides much weaker security compared to main mode.
- **Diffie-Hellman Group Number.** DH group numbers 14, 15, 16, 17, and 18 [64], 19, 20, and 21 [61] are NIST-approved groups. The DH group 22 is not a NIST-approved option because it provides less than 112 bits of security; see [47]. The ECP DH groups 19, 20, and 21 are preferred for security and performance reasons. The DH group used to establish the secret keying material for IKE and IPsec should be consistent with current security requirements for the strength of the encryption keys generated by the IKE KDF.

[58] In most cases, lifetimes should be specified by both time and bytes of traffic so that all SAs, regardless of the volume of traffic, have a limited lifetime. Organizations should not specify a lifetime by bytes of traffic only, because an SA that is not used or used lightly might exist indefinitely.

- **Extra Padding.** As described in Section 4.1.5, ESP packets can contain optional padding that alters the size of the packet to conceal how many bytes of actual data the packet contains, which is helpful in deterring traffic analysis. Having larger packets increases bandwidth usage and the endpoints' processing load for encrypting and decrypting packets, so organizations should only use extra padding if traffic analysis is a significant threat (in most cases, it is not) and costs are not an important factor.
- **Perfect Forward Secrecy (PFS).** Because the PFS option provides stronger security, it should be used unless the additional computational requirements of the additional DH key exchanged would pose a problem. For IPsec servers with permanent IPsec tunnels, this is usually not a problem, but a remote access VPN with thousands of users might experience additional work load if PFS is enabled on all VPN clients.

Design decisions should incorporate several other considerations, as described below:

- **Current and Future Network Characteristics.** This document has already described issues involving the use of NAT. Organizations should also be mindful of other network characteristics, such as the use of IPv6 and wireless networking, when designing an IPsec implementation. For example, if the organization is planning on deploying IPv6 technologies in the near future, it may be desirable to deploy an IPsec solution that supports IPv4 in IPv6 and IPv6 in IPv4 configurations as well as an IPv6-only mode.
- **Incident Response.** Organizations should consider how IPsec components may be affected by incidents and create a design that supports effective and efficient incident response activities. For example, if an IPsec user's system is compromised, this should necessitate canceling existing credentials used for IPsec authentication, such as revoking a digital certificate or deleting a PSK.
- **Log Management.** IPsec should be configured so it logs sufficient details regarding successful and failed IPsec connection attempts to support troubleshooting and incident response activities. IPsec logging should adhere to the organization's policies on log management, such as requiring copies of all log entries to be sent through a secure mechanism to centralized log servers and preserving IPsec gateway log entries for a certain number of days.
- **Redundancy**. Organizations should carefully consider the need for a robust IPsec solution that can survive the failure of one or more components. If IPsec is supporting critical functions within the organization, the IPsec implementation should probably have some duplicate or redundant components. For example, an organization could have two IPsec gateways configured so that when one gateway fails, users automatically switch over to the other gateway (assuming that the gateways support such a failover capability). Redundancy and failover capabilities should be considered not only for the core IPsec components, but also for supporting systems such as authentication servers and directory servers.

7.2.7 Summary of Design Decisions

Table 2 provides a checklist that summarizes the major design decisions made during the first two phases of the IPsec planning and implementation process.

Table 2: Design Decisions Checklist

Completed	Design Decision
Identify Needs (Section 7.1)	
	Determine which communications need to be protected
	Determine what protective measures are needed for each type of communication
	Select an IPsec architecture
	Identify other current and future requirements
	Consider the possible technical solutions and select the one that best meets the identified needs
Design the Solution—Architecture (Section 7.2.1)	
	Determine where IPsec hosts and gateways should be located within the network architecture
	Select appropriate IPsec client software for hosts
	Determine whether split tunneling should be permitted
	Determine whether IPsec hosts should be issued virtual IP addresses
Design the Solution—IKE Authentication (Section 7.2.2)	
	Decide which authentication methods should be supported
Design the Solution—Cryptography (Section 7.2.3)	
	Set the cryptographic policy
Design the Solution—High Speed and Large Server Considerations (Section 7.2.4)	
	Tune the operating system for optimized network performance
Design the Solution—Packet Filter (Section 7.2.5)	
	Determine which types of traffic should be permitted and denied
	Determine what protection and compression measures (if any) should be applied to traffic
Design the Solution—Other Design Considerations (Section 7.2.6)	
	Select maximum lifetimes for IKE and IPsec SAs
	Choose IKEv2 or IKEv1. If using IKEv1, choose between main or aggressive mode
	Select an appropriate DH group number for each chosen encryption algorithm and key size
	Determine whether extra padding should be used to thwart traffic analysis
	Enable PFS if it would not negatively impact performance too much

7.3 Implement and Test Prototype

After the solution has been designed, the next step is to implement and test a prototype of the design. This could be done in one or more environments, including lab, test, and production networks.[59] Aspects of the solution to evaluate include the following:

[59] Ideally, implementation and testing should first be performed with a lab network, then a test network. Only implementations

- **Connectivity.** Users can establish and maintain connections that use IPsec for all types of traffic that are intended to be protected by IPsec and cannot establish connections for traffic that IPsec is intended to block. It is important to verify that all of the protocols that need to flow through the connection can do so. This should be tested after initial SA negotiation as well as after the original SAs have expired and new IKE and IPsec SAs have been negotiated. (During testing, it may be helpful to temporarily shorten the SA lifetimes so that renegotiation occurs more quickly.) Connectivity testing should also evaluate possible fragmentation-related issues for IKE (e.g., certificates) and ESP (e.g., TCP flow issues).
- **Protection.** Each traffic flow should be protected in accordance with the information gathered during the Identify Needs phase. This should be verified by monitoring network traffic and checking IPsec endpoint logs to confirm that the packet filter rules are ensuring that the proper protection is provided for each type of traffic.
- **Authentication.** Performing robust testing of IKE authentication is important because if authentication services are lost, IPsec services may be lost as well. Authentication solutions such as using digital signatures may be complex and could fail in various ways. See Section 7.2.2 for more information on IKE authentication.
- **Application Compatibility.** The solution should not break or interfere with the use of existing software applications. This includes network communications between application components, as well as IPsec client software issues (e.g., a conflict with host-based firewall or intrusion detection software).
- **Management.** Administrators should be able to configure and manage the solution effectively and securely. This includes all components, including gateways, management servers, and client software. For remote access architectures, it is particularly important to evaluate the ease of deployment and configuration. For example, most implementations do not have fully automated client configuration; in many cases, administrators manually configure each client. Another concern is the ability of users to alter IPsec settings, causing connections to fail and requiring administrators to manually reconfigure the client, or causing a security breach.
- **Logging.** The logging and data management functions should function properly in accordance with the organization's policies and strategies.
- **Performance.** The solution should be able to provide adequate performance during normal and peak usage. Performance issues are among the most common IPsec-related problems. It is important to consider not only the performance of the primary IPsec components, but also that of intermediate devices, such as routers and firewalls. Encrypted traffic often consumes more processing power than unencrypted traffic, so it may cause bottlenecks.[60] Also, because IPsec headers and tunneling increase the packet

in final testing should be placed onto a production network. The nature of IPsec allows a phased introduction on the production network as well.

60 The additional resources necessitated by IPsec vary widely based on several factors, including the IPsec mode (tunnel or

length, intermediate network devices might need to fragment them, possibly slowing network activity.[61] In many cases, the best way to test the performance under load of a prototype implementation is to use simulated traffic generators on a live test network to mimic the actual characteristics of expected traffic as closely as possible. Testing should incorporate a variety of applications that will be used with IPsec, especially those that are most likely to be affected by network throughput or latency issues, such as Voice Over IP.[62] Addressing performance problems generally involves upgrading or replacing hardware, offloading cryptographic calculations from software-based cryptographic modules to hardware-based cryptographic modules, or reducing processing needs (e.g., using a more efficient encryption algorithm or only encrypting sensitive traffic).

- **Security of the Implementation.** The IPsec implementation itself may contain vulnerabilities and weaknesses that attackers could exploit. Organizations with high security needs may want to perform extensive vulnerability assessments against the IPsec components. At a minimum, the testers should update all components with the latest patches and configure the components following sound security practices. Section 7.3.2 presents some common IPsec security concerns.
- **Component Interoperability.** The components of the IPsec solution must function together properly. This is of the greatest concern when a variety of components from different vendors may be used. Section 7.3.1 contains more information on interoperability concerns.
- **Default Settings.** Besides the IPsec settings described in Section 7.2, IPsec implementations may have other configuration settings. IPsec implementers should carefully review the default values for each setting and alter the settings as necessary to support their design goals. They should also ensure that the implementation does not unexpectedly "drop back" to default settings for interoperability or other reasons.

7.3.1 Component Interoperability

Another facet of testing to consider is the compatibility and interoperability of the IPsec components. Although there have been improvements in the industry, especially with IKEv2-based IPsec implementations, some vendors make it difficult to interoperate with, or manage, other IPsec devices. Because many vendors offer IPsec clients and gateways, implementation differences among products and the inclusion of proprietary solutions can lead to interoperability problems. Although IPsec vendors use the term "IPsec compliant" to state that they meet the current IETF IPsec standards, they may implement the standards differently, which can cause subtle and hard-to-diagnose problems. Also, some products provide support for components (e.g., encryption algorithms) that are not part of the IPsec standards; this is done for various

---

transport), the encryption algorithm, and the use of IPComp, UDP encapsulation, or optional padding.

61 Similar problems can occur when tunnels are within other tunnels, so that packets are encapsulated multiple times. Typically, the solution for these types of problems is to reduce the size of the MTU value on the host originating the network traffic. The MTU is the maximum allowable packet size. The MTU can be lowered so the IPsec-encapsulated packets are not large enough to require fragmentation.

62 For more information on Voice Over IP, see [66].

reasons, including enhancing ease-of-use, providing additional functionality, and addressing weak or missing parts of the standards. Examples of compatibility issues are as follows:

- The endpoints support different encryption algorithms, compression algorithms, or authentication methods.
- One endpoint requires the usage of a proprietary feature for proper operation.
- The endpoints may encode or interpret certain digital certificate fields or data differently.
- The endpoints default to different parameters, such as DH group 14 versus DH group 19.
- The endpoints implement different interpretations of ambiguous or vaguely worded standards, such as performing SA rekeying in different ways.
- Most gateway implementations interoperate with other vendors' implementations, but many client implementations only interoperate with their own vendor's gateway implementation.

The following are some IKE-related interoperability issues:

- **Certificate Contents.** Different implementations may encode or interpret certificate data fields (e.g., peer identity) differently, or handle certificate extensions such as EKU extensions in conflicting ways. Some vendors have also implemented sending intermediary certificates in a non-standard way.
- **Rekeying Behavior.** When implementations re-negotiate IKE or IPsec SAs, different rekeying behavior can result in lost traffic. One potential area of difficulty is timing-related: when to start using the new SA and when to delete the old SA. In addition, when an IKEv1 SA expires, some implementations delete all IPsec SAs that were negotiated using that IKEv1 SA. Other implementations allow the IPsec SAs to continue until they, in turn, expire. This can also cause interoperability problems. In IKEv1, an expired IKE SA leaving an IPsec SA can also no longer send or respond to DPD packets. IKEv2 resolved these issues by specifying that the deletion of an IKE SA causes the deletion of all its IPsec SAs.
- **Initial Contact Messages.** Some implementations send an Initial Contact notification message when they begin an IKE negotiation with a peer for whom they have no current SAs. This can also be an indication that the sending implementation has rebooted and lost previously negotiated SAs. There can be incompatibility issues if one implementation sends and expects to receive this message, and the other one has not implemented this feature.
- **Dead Peer Detection (DPD).** DPD enables an endpoint to ensure that its peer is still able to communicate. This can help the endpoint to avoid a situation in which it expends processing resources to send IPsec-protected traffic to a peer that is no longer available. If no traffic is sent through an SA, some implementations will delete the SA, even if the negotiated lifetime has not elapsed. DPD messages can be sent to ensure that an otherwise unused SA is kept alive. This can avoid NAT mapping timeouts and the deletion of inactive SAs.

- **Vendor ID.** One endpoint may depend upon a proprietary custom Vendor ID IKE payload to enable a feature that is either absent or inconsistently implemented. This has led some vendors to include Vendor IDs of other vendors in their product to gain compatibility with the other vendor. This can lead to unexpected side effects when one vendor adds a different customization that is activated when the same Vendor ID value is seen.
- **Lifetimes.** Peers may be configured with different values for IKE or IPsec SA lifetimes. IKEv2 allows the sending of the maximum accepted authentication lifetime, so a client connecting to a server will be told within which period of time it is supposed to re-authenticate.

In IKEv1, a misconfiguration of the mode (transport or tunnel) or compression would lead to a failure in establishing the IPsec SA. With IKEv2, transport mode and compression can only be requested. If not confirmed, the IPsec SA must be established in tunnel mode or without compression.

The best way to determine interoperability between vendors is to actually test them in a lab environment. Another approach is to research issues with the products by using Web sites that provide interoperability testing configuration and results, as well as the ability to perform real-time testing.

7.3.2 Security of the Implementation

Another topic to keep in mind during testing is the security of the IPsec implementation itself. IPsec was built with careful thought and consideration for security; however, no protocol or software is completely bulletproof. Security concerns regarding IPsec include the following:

- Some IPsec implementations store PSKs in plain text on the system. This can be accessed by legitimate users and anyone else who gains access to the system. The use of such implementations should be avoided if unauthorized physical access to the system is a concern. However, if it is necessary to use such a product, be sure to apply the appropriate system hardening measures and deploy host-based firewalls and intrusion detection software.
- IPsec allows some traffic to pass unprotected, such as broadcast, multicast, IKE, and Kerberos. Attackers could potentially use this knowledge to their advantage to send unauthorized malicious traffic through the IPsec filters. Be sure to carefully monitor the traffic that is passing through the IPsec tunnel, as well as that which is bypassing it. For example, network-based intrusion detection system or intrusion prevention system devices can typically be configured to alert when non-tunneled traffic appears.
- Periodically, vulnerabilities are discovered in IPsec implementations. Organizations such as the United States Computer Emergency Readiness Team (US-CERT) notify vendors of new vulnerabilities and, at the appropriate time, also notify the public of the issues and the recommended resolutions, such as installing vendor-supplied patches. Information on known vulnerabilities is provided by various online databases, including the National

Vulnerability Database (NVD)[63] and the Common Vulnerabilities and Exposures (CVE) database.[64]

7.4 Deploy the Solution

Once testing is complete and any issues have been resolved, the next phase of the IPsec planning and implementation model involves deploying the solution. A prudent strategy is to gradually migrate existing network infrastructure, applications, and users to the new IPsec solution. The phased deployment provides administrators an opportunity to evaluate the impact of the IPsec solution and resolve issues prior to enterprise wide deployment. Most of the issues that can occur during IPsec deployment are the same types of issues that occur during any large IT deployment. Typical issues that are IPsec-specific are as follows:

- Encrypted traffic can negatively affect services such as firewalls, intrusion detection, QoS, remote monitoring (RMON) probes, and congestion control protocols.
- Unexpected performance issues may arise, either with the IPsec components themselves (e.g., gateways) or with intermediate devices, such as routers.
- IPsec may not work properly on some production networks because of firewalls, routers, and other intermediate packet filtering devices that block IPsec traffic. For example, the devices might have been misconfigured for IPsec traffic or not configured at all—for example, if the IPsec implementers were not aware of the existence of a device. Misconfigured devices are more likely to be an issue with organizations that use a wider variety of network devices or have decentralized network device administration and management. In such environments, the changes needed to permit IPsec could vary widely among devices.
- The environment may change during the deployment. For example, IPsec client software may be broken by a new operating system update. This issue can be handled rather easily in a managed environment, but it can pose a major problem if users have full control over their systems and can select their own client software.

7.5 Manage the Solution

The last phase of the IPsec planning and implementation model is the longest lasting. Managing the solution involves maintaining the IPsec architecture, policies, software, and other components of the deployed solution. Examples of typical maintenance actions are testing and applying patches to IPsec software, deploying IPsec to additional remote sites, configuring additional user laptops as IPsec clients, performing key management duties (e.g., issuing new credentials, revoking credentials for compromised systems or departing users) and adapting the policies as requirements change. It is also important to monitor the performance of the IPsec components so that potential resource issues can be identified and addressed before the components become overwhelmed. Another important task is to perform testing periodically to

---

63 https://nvd.nist.gov/
64 https://cve.mitre.org/

verify that the IPsec controls are functioning as expected. Any new hardware, software, or significant configuration changes starts the process again at the Identify Needs phase. This ensures that the IPsec solution lifecycle operates effectively and efficiently.

Another aspect of managing the IPsec solution is handling operational issues. For example, a common problem is poor performance caused by undesired fragmentation or by not utilizing enough resources (e.g., other available CPUs or sufficient memory) to perform networking tasks. When troubleshooting IPsec connections, a network sniffer such as tcpdump or Wireshark can be very helpful. A sniffer allows the administrator to analyze the communications as they take place and correct problems. IPsec gateway logs and client logs may also be valuable resources during troubleshooting; firewall and router logs may validate whether the IPsec traffic is reaching them, passing through them, or being blocked.

7.6 Summary

This section has described a phased approach to IPsec planning and implementation and highlighted various issues that may be of significance to implementers. The following summarizes the key points from the section:

- The use of a phased approach for IPsec planning and implementation can help to achieve successful IPsec deployments. The five phases of the approach are as follows:
  1. **Identify Needs**—Identify the need to protect network communications and determine how that need can best be met.
  2. **Design the Solution**—Make design decisions in four areas: architectural considerations, authentication methods, cryptographic policy, and packet filters.
  3. **Implement and Test a Prototype**—Test a prototype of the designed solution in a lab or test environment to identify any potential issues.
  4. **Deploy the Solution**—Gradually deploy IPsec throughout the enterprise.
  5. **Manage the Solution**—Maintain the IPsec components and resolve operational issues; repeat the planning and implementation process when significant changes need to be incorporated into the solution.
- The placement of an IPsec gateway has potential security, functionality, and performance implications. Specific factors to consider include device performance, traffic examination, gateway outages, and NAT.
- Although IPsec clients built into operating systems may be more convenient than deploying third-party client software, third-party clients may offer features that built-in clients do not.
- When IPsec hosts are located outside the organization's networks, it may be desirable to assign them virtual internal IP addresses to provide compatibility with existing IP address-based security controls.
- Authentication options include PSKs, digital signatures, and (in some implementations) external authentication services such as EAP and Generic Security Services Application

Program Interface (GSSAPI)/Kerberos. An authentication solution should be selected based primarily on ease of maintenance, scalability, and security.

- Cryptographic algorithms and key lengths that are considered secure for current practice should be used for encryption and integrity protection. AES-GCM with a 128-bit key or 256-bit key is recommended for encryption and integrity. DH ECP groups and the MODP group 14 (2048) are recommended. More than one algorithm can be specified to ease the transition to new updated algorithms.
- Packet filters should apply appropriate protections to traffic and not protect other types of traffic for performance or functionality reasons.
- Specific design decisions include IKE and IPsec SA lifetimes, DH group numbers, extra packet padding, and the use of PFS. When IPsec is going to be used with third parties, design decisions should take the capabilities of those third parties into account, as long as their capabilities are using NIST-approved algorithms and methods. Additional design considerations include current and future network characteristics, incident response, log management, redundancy, and other security controls already in place.
- Testing of the prototype implementation should evaluate several factors, including connectivity, protection, IKE authentication, application compatibility, management, logging, performance, the security of the implementation, component interoperability, and default settings.
- Existing network infrastructure, applications, and users should gradually be migrated to the new IPsec solution. This provides administrators an opportunity to evaluate the impact of the IPsec solution and resolve issues prior to enterprise wide deployment.
- After implementation, the IPsec solution needs to be maintained, such as applying patches and deploying IPsec to additional networks and hosts. Operational issues also need to be addressed and resolved.
- Organizations should implement technical, operational, and management controls that support and complement IPsec implementations. Examples include having control over all entry and exit points for the protected networks, ensuring the security of all IPsec endpoints, and incorporating IPsec considerations into organizational policies.

8 Alternatives to IPsec

This section lists several VPN protocols that are used as alternatives to IPsec and groups them by the layer of the IP model (as shown in Figure 16)[65] at which they function, although the distinction between layers is not always clear. For each VPN protocol, a brief description is provided, along with a description of the circumstances under which it may be more advantageous than IPsec. Some alternatives have specifications and implementations, but some of the alternatives are implementations with some documentation that does not provide a full specification.

Application Layer. This layer sends and receives data for particular applications, such as Domain Name System (DNS), web traffic via Hypertext Transfer Protocol (HTTP) and HTTP Secure (HTTPS), and email via Simple Mail Transfer Protocol (SMTP) and the Internet Message Access Protocol (IMAP).
Transport Layer. This layer provides connection-oriented or connectionless services for transporting application layer services between networks. The transport layer can optionally assure the reliability of communications. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are commonly used transport layer protocols.
Network Layer. This layer routes packets across networks. Internet Protocol (IP) is the fundamental network layer protocol for TCP/IP. Other commonly used protocols at the network layer are Internet Control Message Protocol (ICMP) and Internet Group Management Protocol (IGMP).
Data Link Layer. This layer handles communications on the physical network components. The best-known data link layer protocols are Ethernet and the various WiFi standards such as the Institute of Electrical and Electronics Engineers (IEEE) 802.11.

Figure 16: IP Model

If only one or two applications need protection, a network layer control may be excessive. Transport layer protocols such as TLS are most commonly used to provide security for communications with individual HTTP-based applications, although they are also used to provide protection for communication sessions of other types of applications such as SMTP, Post Office Protocol (POP), IMAP, and FTP. Because all major web browsers include support for TLS, users who wish to use web-based applications that are protected by TLS normally do not need to install any client software or reconfigure their systems. Web-based systems have gained considerable integration support that reaches outside the browser. One common example is the virtual network drive, where the browser takes on the role of a file manager application to securely transmit files.

8.1 Data Link Layer VPN Protocols

Data link layer VPN protocols function below the network layer in the TCP/IP model. These types of VPNs are also known as layer 2 VPNs (L2VPN). This means non-IP network protocols can also be used with a data link layer VPN. Most VPN protocols (including IPsec) only support IP, so data link layer VPN protocols may provide a viable option for protecting networks running

65 Figure 16 repeats Figure 1 for additional clarity.

non-IP protocols. (As the name implies, IPsec is designed to provide security for IP traffic only.). Protection at the link layer means that the security added is limited to the devices that share this link layer, such as an Ethernet-based LAN or WiFi network. However, various virtual link layers now exist to facilitate network virtualization, allowing a link layer VPN protocol to secure nodes in different physical (and virtual) locations. Since confidentiality and integrity happen at the link layer, deploying a link layer VPN protocol requires no specific support in the application. However, this also means that the application is generally not aware of the link layer protection and cannot make decisions based on whether the communication is secure or not.

8.1.1 WiFi Data Link Protection

All devices that support WiFi technology support a number of link layer protocols that provide confidentiality and integrity protection. Wireless connections broadcast their data, so from the start there has been a push to send data using confidentiality and integrity protection. The initial security protocol was Wired Equivalent Privacy (WEP), deprecated in 2004 for Wi-Fi Protected Access (WPA). WEP uses 40-bit or 128-bit RC4 PSKs and is easily broken, whereas WPA2[66] uses AES-CCM. The Enterprise versions of WPA use IEEE 802.1X for authentication instead of a PSK. WPA supports a number of EAP extensions, such as EAP-TLS, EAP-MSCHAPv2, and EAP-Subscriber Identity Module (EAP-SIM). In WPA3, the PSK is replaced by Password Authenticated Key Exchange (PAKE) which offers more protection against the use of weak passwords. WPA3 also offers PFS.[67]

The strength of the link layer protection for WiFi depends strongly on the configuration and the implementation of the various 802.11 standards. WiFi encryption only protects the data from the wireless device to the wireless access point. It is good practice to consider WiFi encryption to be insufficient and to not trust the access point. Devices on a WiFi network should use a remote access VPN like IPsec to communicate with resources on the wired network. This is especially true for WiFi access points belonging to third parties, such as restaurants and hotels.

8.1.2 Media Access Control Security (MACsec)

MACsec is an industry standard defined in IEEE 802.1AE. It creates point-to-point security associations within an Ethernet network. MACsec is the Ethernet version of WiFi WPA security. It uses AES-GCM with 128-bit keys for confidentiality and integrity. It protects regular IP traffic, as well as ARP, IPv6 Neighbor Discovery (ND), and DHCP. For key exchange and mutual authentication, MACsec uses the IEEE 802.1X extension MACsec Key Agreement (MKA) protocol. New devices have to authenticate themselves to the authentication server before being able to join the network, and communication with other hosts on the network are encrypted between each pair of hosts. This allows MACsec to be used with virtual network

---

[66] WPA version 1 was designed as a compromise between security and being able to run on old hardware that implemented WEP. It uses the Temporal Key Integrity Protocol (TKIP) which was a stopgap replacement for the broken WEP protocol, but TKIP is also no longer considered secure. WPA2 mandated the support for the Counter Mode with Cipher Block Chaining Message Authentication Code Protocol (CCMP), which uses AES-CCM.

[67] See also NIST SP 800-153, *Guidelines for Securing Wireless Local Area Networks (WLANs)* [67].

technologies such as Virtual eXtensible LAN (VXLAN) and GEneric NEtwork Virtualization Encapsulation (GENEVE).

MACsec can protect two machines via a switch even if the switch itself does not support MACsec. However, if the switch supports MACsec, each individual Ethernet port of the switch can become a node in the MACsec network for devices connected to those ports that do not support MACsec natively. In that case, all traffic between this device and the LAN is encrypted, except from the Ethernet port to the actual device.

The Ethernet packet change to support MACsec is similar to the change of an IP packet to support IPsec. The Ethernet header is extended with the SecTAG header, which contains the equivalent to the ESP SPI number and Sequence Number. This is followed by the (now encrypted) original payload, followed by the ICV.[68] To a switch that does not support MACsec, the SecTAG and ICV look like just part of the regular Ethernet frame payload.

Similar to IPsec, MACsec can be configured to use manual keying. It suffers from all the same problems as IPsec manual keying: no PFS, and no protection from reusing the same counters as nonces for AES-GCM.

8.2 Transport Layer VPN Protocols (SSL VPNs)

Transport layer VPNs are what people usually think of when describing a VPN. The host obtains a new virtual interface configured with one or more IP addresses. Packets to and from this virtual interface use a transport protocol to encapsulate the packets securely to the remote endpoint of the VPN. The packets are then further routed, just like packets that arrived on a physical network interface. The most common IPsec alternative is the Secure Sockets Layer (SSL) VPN. Although these are still called SSL VPNs, most actually use the TLS protocol and not the older SSL protocol. This can be TLS [16] based on TCP or DTLS [68] based on UDP. The advantage is that SSL VPNs' traffic is much harder to be blocked, as it can run on any (preconfigured) port number. Usually, it is run over port 443 (HTTPS) since most networks pass on this traffic without attempting any kind of deep packet inspection. When using TCP, it can suffer from severe performance degrading due to dueling TCP layers when there is congestion or packet loss; DTLS does not have this problem. SSL VPNs are usually implemented as an application, resulting in significantly lower performance compared to kernel-based VPNs such as IPsec or WireGuard.

NIST provides specific guidance for SSL VPN deployments in NIST SP 800-113, *Guide to SSL VPNs* [69].

8.2.1 Secure Socket Tunneling Protocol (SSTP)

Secure Socket Tunneling Protocol (SSTP) is the Microsoft version of an SSL VPN. It uses SSL/TLS over port 443 and can use TCP or UDP as the underlying protocol. It uses the SSTP

68 In ESP, the ICV is only used for non-AEAD protocols. For AEAD protocols such as AES-GCM, the ICV is implicit and generated from the IKE session and not transmitted over the wire.

protocol to run a Point-to-Point Protocol (PPP) session that handles the IP assignment and IP encapsulation. Microsoft calls this a Point-to-Site VPN, which is another name for remote access VPN. It supports the standard encryption and integrity algorithms that SSL/TLS support.

8.2.2 OpenConnect

OpenConnect originated as an open source replacement implementation for the Cisco AnyConnect SSL VPN client using the Cisco proprietary AnyConnect protocol. OpenConnect is now a protocol specification and a client and server implementation. While it remains backwards compatible with Cisco AnyConnect, it has added its own features and has been submitted to the IETF as a draft to become an Informational RFC [70]. It uses DTLS but can fall back to TLS over TCP when needed. The server is authenticated via a machine certificate. Clients can authenticate using a user/password, certificate, or Kerberos (GSSAPI). The OpenConnect client also supports other proprietary SSL VPN protocols that are similar to Cisco AnyConnect, such as Palo Alto GlobalProtect and Juniper SSL-VPN. OpenConnect is a relatively new SSL VPN and has not been deployed as much as other SSL VPNs.

8.2.3 OpenVPN

OpenVPN is a popular SSL VPN protocol/implementation that was originally written in 2001. It uses SSL or TLS over any preconfigured port and can use TCP or UDP as the transport protocol. The supported algorithms are the common SSL/TLS algorithms. For authentication, it supports certificates, PSKs, and user/password. It can act as a link layer VPN or as a transport layer VPN. The server can send the client commands to be executed, which can be dangerous. OpenVPN has a larger attack surface because the entire protocol runs as a user process and has had vulnerabilities in the past. It is one of the more widely used SSL VPNs.

8.3 WireGuard

WireGuard[69] is a fairly new VPN implementation originally written for the Linux kernel. It is a minimalistic VPN implementation that is less complex than IPsec, but as a result is also not as flexible as IPsec. There is no formal protocol specification or publication in static form, which makes it harder to find compatibility issues between different versions, although it does provide extensive documentation of the current implementation. The code base is small compared to other VPN implementations. It combines the control and data plane over a single preconfigured UDP port.

WireGuard uses the Noise Protocol Framework[70] for its key exchange and the HMAC-Based Key Derivation Function (HKDF) [71] to generate symmetric encryption keys. It uses Curve25519 [72] as its DH group and supports authentication only via public keys. It uses CHACHA20POLY1305 [73] as its encryption and integrity algorithm. None of these algorithms

[69] https://www.wireguard.com
[70] https://www.noiseprotocol.org/

are NIST-approved at the moment. However, NIST plans to allow Edwards Curve DSA (EdDSA) digital signatures [74].

There are many similarities with IPsec and IKE. WireGuard uses IKEv2-style DDoS COOKIES and DPD/Keepalives. The data packet looks very similar to ESP in tunnel mode. Transport mode is not supported. Its replay attack protection is the same as IPsec, using a replay window of 2000 (continuous packet ids). It supports PPK and has the same seamless reconnection properties as MOBIKE where a device can switch network interfaces without losing the VPN connection. WireGuard takes advantage of multiple CPUs when present, unlike typical SSL VPNs that are bound to one CPU.

The protocol does not allow for DHCP-style IP address allocation, and IP addresses are hard-coded in its configuration file on the client and server. DNS configuration has to be conveyed via a provisioning protocol. WireGuard lacks authentication support using certificates or PSKs. It does not support a transport mode configuration, making it less suitable for mesh encryption. It does not support AES-GCM.

WireGuard is mostly intended as a remote access VPN. As such, it does a much better job compared to SSL VPNs and SSH. While it can be used in a gateway-to-gateway or host-to-host architecture, it misses the optimizations and flexibility of IPsec in these architectures.

8.4 Secure Shell (SSH)

SSH is a commonly used application layer protocol suite. While it is commonly used as a secure remote login application and a secure file transfer application, it can also be used to tunnel specific ports via an SSH connection to allow either a local connection to access a remote resource, or a remote connection to access a local resource. SSH is often used on intermediary hosts (also called bastion hosts) to jump to other hosts, but that jump does not need to be to the remote login (SSH) host itself. For instance, port 25 on localhost (127.0.0.1) could be made available to locally running mail clients, with SSH tunneling this traffic over the SSH VPN to the bastion host, where the SSH client running will forward the traffic to a remote mail server's port 25. Because a single SSH tunnel can provide protection for several applications at once, it is technically a transport layer VPN protocol, not an application layer protocol.

While SSH could be used to start a PPP daemon to create a more traditional VPN with an interface, recent versions of OpenSSH have added native functionality for binding the SSH protocol to tun interfaces on the hosts. An SSH tunnel creates a tun interface on the local and remote host, and these tun interfaces can be configured with other IP addresses, providing a true remote access VPN.

As with SSL VPNs, SSH VPNs perform badly if there is packet loss, due to multiple TCP layers independently retransmitting packets.

SSH tunnel-based VPNs are resource-intensive and complex to set up. They require the installation and configuration of SSH client software on each user's machine, as well as the reconfiguration of client applications to use the tunnel. Each user must also have login privileges on a server within the organization; because this server typically needs to be directly accessible from the Internet, it is susceptible to attack. Generally, users need to have solid technical skills so that they can configure systems and applications themselves, as well as troubleshoot problems that occur. The most common users of SSH tunnel-based VPNs are small groups of IT administrators.

8.5 Obsoleted and Deprecated VPN Protocols

A number of commonly used VPN protocols are no longer suitable for use. Some of these were designed for dial-up internet connections. Some used encryption techniques that were broken or have become too weak to withstand current computational attacks. Early VPN protocols were implemented on top of PPP [75]. These solutions were built as extensions to secure modem-based connections and are no longer appropriate to deploy, both from an architectural point of view and from a cryptographic point of view. The protocols listed in this section must not be used.

8.5.1 Point-to-Point Tunneling Protocol (PPTP)

The Point-to-Point Tunneling Protocol (PPTP) [76] uses Generic Routing Encapsulation (GRE, IP protocol 47) as its transport protocol. The GRE tunnel is used to send PPP packets. Similar to the ESP protocol, NAT routers often do not forward this protocol. PPTP uses TCP port 1723 as its control plane. It uses the Microsoft Point-to-Point Encryption (MPPE) mechanism at the PPP layer for encryption. MPPE uses the deprecated RSA RC4 algorithm with 40-bit or 128-bit keys [77]. For authentication it can use the Password Authentication Protocol (PAP) [78] or Challenge Handshake Authentication Protocol (CHAP) [79]. Microsoft created MS-CHAPv1 and MS-CHAPv2 to provide stronger forms of authentication, but researchers have found serious weaknesses in MS-CHAP.[71] The original version of PPTP contained serious security flaws. PPTP version 2 addressed many of these issues, but researchers have identified weaknesses with this version as well (in addition to the MS-CHAP issues).[72] PPTP should not be used, and if it is used regardless, it should be considered as a plaintext protocol with no functional confidentiality or integrity protection.

8.5.2 Layer 2 Tunneling Protocol (L2TP)

The Layer 2 Tunneling Protocol (L2TP) [80] is the successor to PPTP. Instead of using the GRE protocol, it encapsulates PPP packets inside UDP on port 1701. For confidentiality and integrity of the data plane, it depends on IPsec. Some implementations support encryption at the PPP layer, meaning that to enable IPsec support, one has to (confusingly) disable “L2TP encryption”. L2TP without IPsec is used by some ISPs as the replacement of PPTP connections, but this usage is not a VPN. L2TP VPNs all use IPsec in transport mode, commonly referred to as L2TP/IPsec. In addition to the PPP-provided authentication methods, L2TP can also use other

---

71 One paper discussing MS-CHAP weaknesses is “Exploiting Known Security Holes in Microsoft’s PPTP Authentication Extensions (MS-CHAPv2)” by Jochen Eisinger (http://www2.informatik.uni-freiburg.de/~eisinger/paper/pptp_mschapv2.pdf).

72 For more information on PPTP security issues, see Bruce Schneier’s “Analysis of Microsoft PPTP Version 2” page, located at https://www.schneier.com/academic/pptp/.

methods, such as RADIUS [81], although it commonly uses the PPP-based MS-CHAPv2 for authentication of the PPP layer. IPsec is established using IKEv1, often using a weak group PSK, but it can be deployed using X.509 certificates as well. Even when deployed securely, L2TP/IPsec offers no advantage over IKEv2-based IPsec VPNs. It adds a number of unnecessary encapsulation layers that reduce the effective MTU and increase network issues related to packet fragmentation. Additionally, because it uses IPsec in transport mode, it works poorly behind NAT. Some vendors switch to tunnel mode when behind NAT, but not all L2TP/IPsec servers are configured to support tunnel mode.

One advantage of L2TP/IPsec used to be that it was shipped as part of popular operating systems, which meant no separate VPN software needed to be purchased and installed. Up-to-date versions of those operating systems now support IKEv2-based IPsec VPNs. Additionally, L2TP/IPsec VPNs usually do not support AEAD algorithms such as AES-GCM, which increases the CPU usage compared to IKEv2-based IPsec VPNs. On mobile devices this means using more battery power. L2TP/IPsec deployments should be migrated to IKEv2-based IPsec VPNs.

8.6 Summary

Section 8 describes the main alternatives to IPsec. SSL VPNs are popular because they are not as easily blocked as IPsec VPNs, although this advantage will be negated once IKEv2-based IPsec implementations add support for TCP and TLS encapsulation as specified in [49]. Traditionally, SSL VPNs were easier to set up and use than IPsec VPNs, but IKEv2 configurations and provisioning systems have improved considerably making IPsec VPNs as easy to set up and use as SSL VPNs. WireGuard is an interesting upcoming remote access VPN protocol, but at the moment has no support for NIST-approved algorithms.

9 Planning and Implementation Case Studies

This section presents a few typical IPsec solution planning and implementation case studies. Each case study begins by describing a real-world security requirement scenario, such as protecting network communications between two offices. The case study then discusses possible solutions for the security requirement and explains why IPsec was selected over the alternatives. The next section of each case study discusses the design of the solution and includes a simple network diagram that shows the primary components of the solution (e.g., IPsec gateways and hosts, routers, switches). Each case study also provides some details of the implementation of the solution prototype, which include examples of configuring the solution using commonly available equipment and software, based on an implementation performed in a lab or production environment. Each case study ends with a brief discussion that points out noteworthy aspects of the implementation, indicates when another case study model may be more effective, and discusses variants on the case study scenario that might be of interest to readers.

The case studies are not meant to endorse the use of particular products, nor are any products being recommended over other products. Several common products were chosen so the case studies would demonstrate a variety of solutions. **Organizations and individuals should not replicate and deploy the sample configuration files or entries.** They are intended to illustrate the decisions and actions involved in configuring the solutions, not to be deployed as-is onto systems.

The case studies presented in this section are as follows:

- Protecting communications between two local area networks (remote office, main office)
- Protecting wireless communications in a small office/home office environment
- Protecting communications between remote users (e.g., telecommuters, road warriors) and the main office's network
- Protecting a datacenter or cloud network using mesh encryption

9.1 Connecting a Remote Office to the Main Office

An organization with a single office location is planning the creation of a small remote office, which includes identifying any needs to protect network communications. To perform various job functions, most users at the remote office will need to access several information technology (IT) resources located at the main office, including the organization's email, intranet web server, databases, and file servers, as well as several business applications. Currently, email is the only one of these resources that can be accessed from outside the main office (it is available through the Internet using a web-based email client). Communications with most of the IT resources will involve transferring sensitive data (such as financial information) between systems. To support its mission, the organization needs to maintain the confidentiality and integrity of the data in a cost-effective manner. (At this time, the need is to protect communications initiated by remote office hosts to the main office network only; in the future, the solution might be extended to protect communications initiated by main office hosts to the remote office network.) The following sections describe how the organization evaluates its options, identifies a viable solution, creates a design, and implements a prototype.

9.1.1 Identifying Needs and Evaluating Options

As described below, the organization considers a few options for providing access from the remote office to IT resources at the main office and protecting the data:

- **Data Link Layer Solution: Leased Line.** The organization could establish a dedicated leased line between the remote office and the main office. This would provide a private communications mechanism for all the network traffic between the offices. (If the organization were concerned about security breaches of the leased line, additional protection measures such as a data link layer VPN protocol could be used to provide another layer of security.) Unfortunately, because the remote office is geographically distant from the main office, a leased line would be prohibitively expensive.
- **Network Layer Solution: Network Layer VPN.** The organization could establish a network layer VPN between the remote office and main office. Connecting the remote office to the Internet and establishing a VPN tunnel over the Internet between the offices could provide access to the resources and protect the communications. The VPN could have a remote access architecture, which would reduce hardware costs (only one gateway needed) but increase labor costs (deploying and configuring clients on each remote office system). A gateway-to-gateway architecture would increase hardware costs and decrease labor costs; in effect, the VPN would be invisible to users. The two models also differ in terms of authentication. In a gateway-to-gateway VPN, the gateways would authenticate with each other; in a remote access VPN, each user would need to authenticate before using the VPN. A gateway-to-gateway VPN could also be configured to permit authorized users from the main office to access resources on the remote office's network. Although this is not a current need, it could be in the future.
- **Transport Layer Solution: Web-Based Applications.** The organization could provide web-based access to all required IT resources. This could be done either by creating or acquiring web-based clients for each resource, or by deploying a terminal server that provides access to the resource and providing a web-based terminal server client to employees. All web-based applications would use the TLS protocol over HTTP (transport layer security controls) to protect the confidentiality and integrity of data and authentication credentials. By connecting the remote office to the Internet and making the web-based applications available from the Internet, users at the remote office could use the required IT resources, and the communications would be protected. The main office's network perimeter could be configured to permit external access to the resources only from the remote office's IP address range, which would reduce the risk of external parties gaining unauthorized access to the resources. Users would need to be authenticated by the terminal server, the individual applications, or both the server and the applications.
- **Application Layer Solution: Application Modification.** The organization could purchase add-on software and modify existing applications to provide protection for data within each application. However, a brief review of the required IT resources shows that

several of them are off-the-shelf applications that cannot be modified and cannot be protected by third-party application add-ons. Even if the applications could be deployed to protect their own communications, the applications would have to be directly accessible by remote users, which would significantly increase their exposure to threats. The organization is also concerned about the effectiveness of application layer controls in protecting data. Application layer controls may also conceal information from network layer security controls such as network-based intrusion detection systems, necessitating the use of additional host-based security controls that can monitor application layer activity. Having separate controls for each application also complicates or precludes centralized enforcement of security policies across multiple applications, as well as centralized authentication (unless each application supports the use of a third-party authentication server.)

The organization considers the network layer and transport layer options to be the most feasible for meeting its remote access needs. The data link layer and application layer solutions are too expensive, compared to the network and transport layer solutions. Further investigation of the transport layer solution determines that it is not possible or practical to provide web-based interfaces for several of the desired IT resources. For example, some of the desired applications are off-the-shelf products that offer no web-based client. A terminal server solution could provide access, but this would require users to connect to the terminal server and authenticate before accessing any applications. Also, each host would need the terminal server client to be installed and configured.

After comparing the three remaining solutions (remote access network layer VPN, gateway-to-gateway network layer VPN, and terminal server transport layer VPN) and considering how each solution would be deployed in the organization's environment, the organization chooses the gateway-to-gateway network layer VPN. Its primary advantages are that it should be relatively easy for the organization to deploy and maintain, and it will be transparent to users. The organization expects to be able to configure the Internet routers at the main office and remote office to act as VPN gateways, so no additional hardware will be needed. Also, each office already routes internally generated network traffic designated for another office's network to its Internet router, so routing changes should need to be made only on the Internet routers themselves. Another advantage of the gateway-to-gateway VPN is that in the future, users at the main office could use it to access resources at the remote office. There is no current need for this, but it is likely that as the remote office matures, this may become a necessity.

9.1.2 Designing the Solution

The organization hopes to use its Internet routers as endpoints for the VPN solution, see Figure 17 below. Both routers support IPsec, and IPsec should be able to protect confidentiality and integrity adequately for the organization's needs, so the plan is to configure the routers to provide an IPsec tunnel. Based on the organization's performance requirements, the routers should be able to handle any additional load because they are currently lightly utilized.[73] Figure 17 illustrates the planned design for the VPN architecture. The main office and remote office networks are on separate private networks, each with an IPv4 network. Each private network is connected to the Internet through a router that provides NAT services. The plan is to establish an IPsec tunnel between the external interfaces of the two routers. Desktop computers on the remote office network will send unencrypted information to the office's Internet router. The router acts as a VPN gateway, encrypting the traffic and forwarding it to the destination router at the main office, which also acts as a VPN gateway. The main office router decrypts the traffic and forwards it to its final destination, such as a file server or email server. Responses from the servers to the desktops are returned through the tunnel between the gateways.

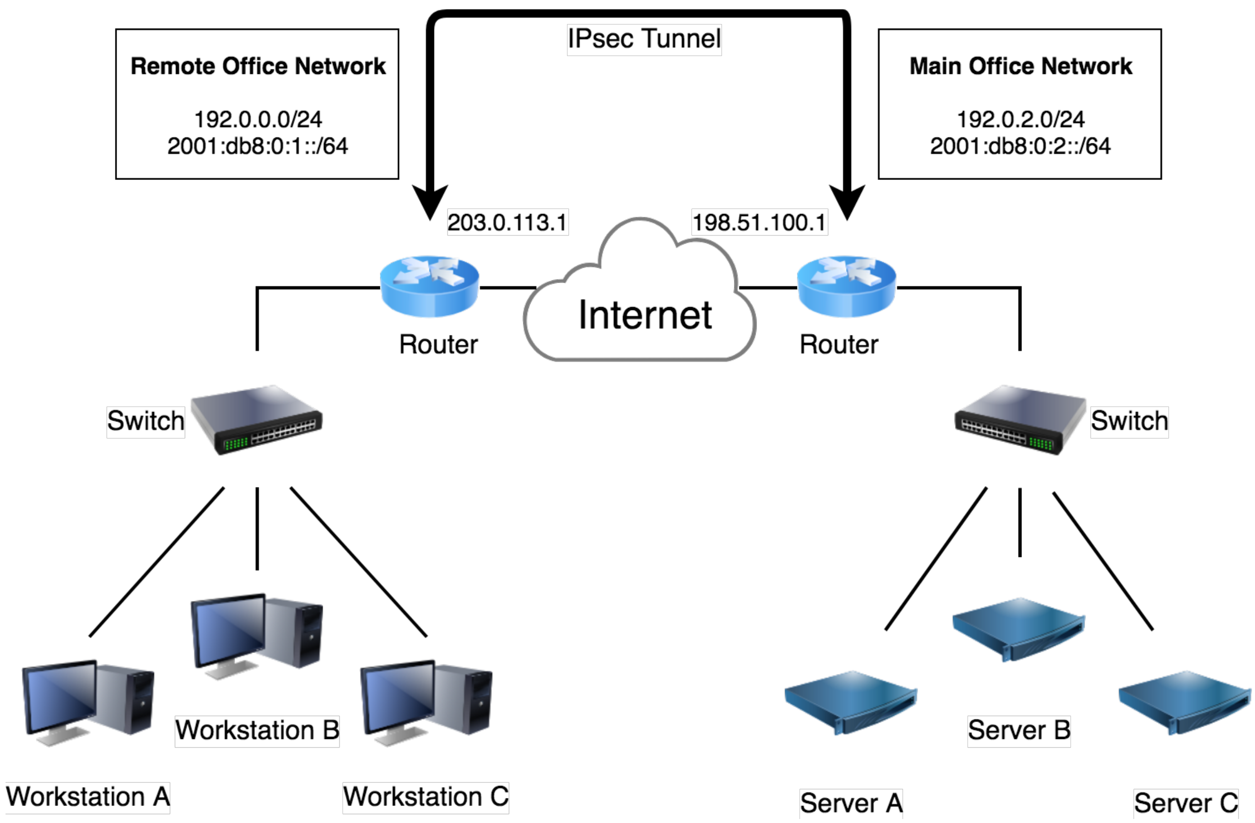

Figure 17: Gateway-to-Gateway VPN for Remote Office Connectivity

In this scenario, NAT is an important architectural consideration. If possible, the design should keep NAT services out of the IPsec tunnel path to avoid potential NAT-related incompatibilities and to simplify the design. This means that outgoing packets to the remote network needing to pass through the IPsec tunnel should be excluded from NAT.

---

[73] If the load on the routers increases significantly in the future, cryptographic accelerator cards possibly could be added to the routers. (Not all routers support the use of such cards.)

After designing the architecture, the network administrators next consider other elements of the design, including the following:

- **Authentication.** Because the VPN is being established between only two routers, a strong PSK with entropy of at least 112 bits should provide adequate authentication with minimal effort (as compared to alternatives such as digital certificates). The routers will encrypt the PSK in storage to protect it.
- **IKE and ESP Algorithms.** Since 128-bit AES provides sufficiently strong encryption, it is chosen initially for ESP to prevent potentially overloading the gateways. The AES-GCM algorithm is a good choice for IKE and ESP, because it is an AEAD algorithm providing encryption and integrity together in an efficient and more secure manner. It is preferred over the older combined algorithms with separate encryption and integrity algorithms, such as AES-CBC with HMAC-SHA-2. The PRF used is SHA-256-HMAC. If the DH group chosen is DH 19, a modern and strong ECP group that provides 128 bits of security strength. PFS is enabled to ensure that a compromise of one of the routers will not cause all previously captured encrypted traffic to be vulnerable to decryption. A fallback proposal using AES-CBC with HMAC-SHA-2 is added to ensure maximum interoperability with other devices, as not all devices support AES-GCM for IKE and ESP. The initiator must use a DH group that is also supported by the responder.
- **Packet Filters.** The network administrators work with the security staff to design packet filters that will permit only the necessary network traffic between the two networks and will require adequate protection for traffic. To make initial testing of the solution easier, the administrators decide that the packet filters should allow all IP-based communications from the remote office's hosts to the main office's hosts. Once initial testing has been completed, more restrictive packet filters will be added and tested. The packet filters should permit only the necessary communications and specify the appropriate protection for each type of communication.
- **MTU and Fragmentation.** Since the IPsec tunnel is using an ISP, and the network might not support packets larger than 1500 bytes, both routers are set to use TCP MSS clamping at 1440 bytes, as path MTU discovery might not work properly across the network.

9.1.3 Implementing a Prototype

Because the organization has limited network equipment and does not have a test lab, the IT staff decides the best option for validating the solution is to test it after hours using the production routers once the remote office network infrastructure is in place and Internet connectivity has been established. If the testing causes a connectivity outage, the impact should be minimal. The network administrators perform the following steps to configure and test a prototype of the IPsec solution:

1. **Back up the routers.** Backing up the router operating system and configuration files is a necessity since the prototype is being implemented on production equipment. Even in a test environment, performing a backup before making any changes is often very helpful because the routers can be restored quickly to a "clean" state.

2. **Update the firmware of the routers.** To ensure that no known bugs are left unfixed, the routers are updated to the latest firmware and assessed for regular operation without any other changes in configuration. One endpoint is updated and rebooted. Once the network is confirmed to be operating properly, the other endpoint's firmware is updated, and the router is rebooted. Once both routers are confirmed to be working properly on the latest firmware, the process of configuring the routers for IPsec can be started.

3. **Verify the security of the routers.** The network administrators should perform a vulnerability assessment to identify any existing security issues with the routers, such as unneeded user accounts or inadequate physical security controls. The administrators should then address all identified issues before proceeding, or the IPsec implementation may be compromised quickly.

4. **Update the endpoints to support IPsec.** This could involve patching the operating system, installing or enabling IPsec services, or making other changes to the endpoints so that they can support IPsec services. In this case, both endpoints happen to be Cisco routers, so the administrators double-check each router to confirm that it can support IPsec and the desired encryption algorithm.

5. **Specify the IKE cryptographic algorithms.** For our preferred proposal, use AES-GCM, since it is an AEAD algorithm; specify a PRF. For the fallback proposal, use AES-CBC with HMAC-SHA-256. It will use SHA-256 (in HMAC) for integrity protection as well. The following ECP DH group (19) is specified.

```
crypto ikev2 proposal 1
 encryption aes-gcm 256
 prf sha256
 group 19

crypto ikev2 proposal 2
 encryption aes-cbc-256
 integrity sha256[74]
 group 19[75]

crypto ikev2 policy default
 proposal 1
 proposal 2
 match fvfr any
```

6. **Specify the IKE authentication method.** In this case, each router needs to be configured to use a PSK, as illustrated by the following configuration entries[76]. Instead of IP

---

[74] For AEAD algorithms, a PRF needs to be specified. For non-AEAD algorithms, the PRF defaults to the integrity algorithm.

[75] Change this value to 14 and/or 15 if DH 19 is not supported by the other device.

[76] Secure transport for the PSK is provided by one of the network administrators, who physically carries a copy of the key from the main office to the remote office.

addresses as identifiers, Fully Qualified Domain Names (FQDNs) will be used. An easy way to create a strong random PSK is to use the openssl command: `openssl rand -base64 64`

```
crypto ikev2 profile default
 identity local fqdn west.example.gov
 match identity remote fqdn east.example.gov
 authentication local pre-share key XXXXXXXXX
 authentication remote pre-share key XXXXXXXXX
```

7. **Specify the IPsec mode and cryptographic algorithms.** The following configuration entry on each router specifies ESP tunnel mode, preferring AES-GSM instead of AES-CBC-128 encryption with HMAC-SHA-256 integrity protection:

```
crypto ipsec transform-set 1 esp-gcm-128[77]
 mode tunnel
crypto ipsec transform-set 2 esp-cbc-128
 mode tunnel
```

8. **Define the packet filters.** The following configuration entry tells the routers which packets should be permitted to use IPsec:

```
ip access-list extended 100
 permit ip 192.0.0.0 0.0.0.255 192.0.2.0 0.0.0.255
 permit ipv6 2001:db8:0:1::/64 2001:db8:0:2::/64
```

9. **Tie the IPsec settings together in a crypto map.** On Cisco routers, the settings created in steps 5, 6, and 7 need to be connected. This can be done through the following configuration settings, which create a crypto map called *west-east*:

```
crypto map west-east 1 ipsec-isakmp
set peer 203.0.113.1
set transform-set 1 2
set pfs group19[78]
set ikev2-profile default
match address 100
```

10. **Apply the IPsec settings to the external interface.** Because the external interface of the router will provide IPsec services, the crypto map created in the previous step must be applied to the external interface. This is done through the following commands:

```
interface g1/1
```

[77] The term *transform set* refers to the VPN algorithms and security protocols.
[78] For devices not supporting DH 19, use DH 14 and/or DH 15.

```
crypto map west-east
```

11. **Review the configuration.** After configuring both routers, the administrators review the routers' configurations to ensure that all the necessary settings are in place.[79] The following commands can be used to display the policies:

```
show crypto ikev2 policy
show crypto map
```

12. **Test the solution.** Administrators can test the solution by attempting to gain access to main office resources from a desktop at the remote office. The test should also include using packet sniffers to monitor the network traffic at both offices and confirm it is properly protected. If successful, the configuration could be updated to use 256-bit keys for ESP encryption. If the test is unsuccessful, the administrators should troubleshoot the problem, make any necessary corrections or changes, then test the solution again.[80] Additional test actions should include implementing the restrictive packet filters and verifying them, and verifying that the correct algorithms are used. For example, some IPsec implementations have a fallback policy that causes weaker algorithms to be used if the user-selected settings cannot be negotiated successfully; this could provide inadequate protection for communications.

9.1.4 Analysis

Setting up an IPsec tunnel between Internet routers can be effective in connecting remote offices with multiple users to another network. It can reduce costs because remote offices need only Internet connectivity instead of a leased line. In addition, all traffic from the remote office could be routed though the main corporate firewall, which could decrease the costs and risks associated with the administration of multiple firewalls. To set up this type of implementation, both routers need to have a static IP address because the addresses would have to be entered into the IPsec configurations. In most cases, this is not an issue for the router at the main office, but it may be a problem for locations such as home offices that often use DSL or cable modem services, which may offer only dynamic IP addresses. Remote access solutions may be more practical for such situations.

In this case study, a gateway-to-gateway VPN was established between a remote office and the main office. An interesting variant on this scenario is a gateway-to-gateway VPN between the main office and the network of a business partner. In such a case, more stringent security measures may be needed to satisfy each organization's requirements for communication. Also, the organizations should establish a formal interconnection agreement that specifies the technical and security requirements for establishing, operating, and maintaining the interconnection, as

---

[79] Appendix C.1 contains a sample configuration file from one of the routers.

[80] The **debug crypto ikev2**, **debug crypto ipsec**, and **debug crypto engine** commands cause the router to display any errors related to the crypto implementation in the terminal window. This can be useful in determining why a connection is failing. Also, the **clear crypto sa** command can be used to clear part or all of the SA database, which may clear some errors.

well as documenting the terms and conditions for sharing data and information resources in a secure manner. Appendix B contains more information on interconnection agreements.

In a gateway-to-gateway VPN between the organization and a business partner, each organization typically has control over its own VPN gateway. Accordingly, the organizations need to identify an acceptable out-of-band method for provisioning each other's gateways with the necessary authentication information, such as PSKs or digital certificates. Another possible difference from the original scenario is that in the business partner scenario, both organizations should configure their packet filters to be as restrictive as possible from the beginning of the implementation. The organizations also need to coordinate their testing efforts and determine how a prototype for the solution can best be tested.

9.1.4.1 Direct remote branch access versus hub-spoke

The solution for one remote location can be extended with additional remote office locations. If one remote office needs to be able to communicate to other remote offices, another design decision needs to be made. Either each remote office can build an IPsec tunnel to each other remote office and bypass the main office, or each remote office can contact other remote offices via the main office. This latter setup is called a *hub-spoke setup*.

The advantage of the hub-spoke architecture is that the main office is the central hub that can dictate policies and inspect all traffic. If a remote office wants to communicate with another remote office, it involves two separate IPsec tunnels. The hub server decrypts the traffic from the first remote office, performs network inspection and packet filter restrictions on the network traffic, and then re-encrypts the traffic to send it via the second IPsec tunnel to the second remote office. Adding a branch does not require any other branches to be reconfigured for the new branch.

The disadvantage of the hub-spoke architecture is the main office requires a lot more bandwidth to facilitate all the remote branches' traffic to each other. It might require an IPsec service with additional hardware acceleration network cards to be able to handle all the IPsec traffic. It also becomes a single point of failure. When the branches communicate via their own IPsec connections, the branches are more independent of the main office. It does require more management, since whenever a branch office is added or modified, all other branches need to have their IPsec configurations updated. Any network inspection configurations and packet filters can still be centrally managed but need to be pushed out to the branch locations.

9.2 Protecting Communications for Remote Users

A system administrator of a federal agency has been giving out SSH access to individual developers who sometimes work from home. While usable for remote logins via SSH, reaching various reporting servers required complicated port forwarding configurations for SSH that were prone to misconfiguration. It was decided that a proper remote access VPN should be deployed. It would allow the remote users to directly access the agency's servers from their browser once connected to the VPN, without needing SSH.

The system administrator had also learned that the WiFi at the office was using WPA2 security, which had seen a number of attacks and was no longer considered secure enough. However, the WiFi hardware vendor had no plans to support WPA3 for the hardware they used. The system administrator wanted to treat the office WiFi as insecure and require the remote access VPN to connect to the office network, even from the office WiFi network.

9.2.1 Identifying Needs and Evaluating Options

As described below, a federal agency may consider a few options for protecting the connections to their secure internal network for remote users as well as local WiFi users.

- **Network Layer Solution: Network Layer VPN.** The organization could establish network layer VPNs between the developers and the agency's main office. The VPN tunnels would provide access to the agency internal resources without the need for hopping through a number of servers via SSH. The organization considers each possible network layer VPN architecture, as follows:
    - A gateway-to-gateway VPN solution is not suitable because the developers work from a number of remote locations, such as co-sharing spaces, hotels, and coffee shops. The developers need access from their laptops and phones, not desktops at home.
    - The agency already has a flexible FreeBSD-based internet gateway. A remote access VPN solution for FreeBSD would allow the agency to use its existing gateway, eliminating additional hardware costs. Each remote device would need VPN client software installed, but their laptops and phones already support IKEv2 remote access VPNs, so additional labor would be limited to supporting the developers in performing the configuration and troubleshooting issues. The agency would not even need to pay for additional VPN client licenses.
- **Transport Layer Solution: Web-Based Access Solution.** The agency could provide web-based access to resources. This could be accomplished by deploying secured web-based services. This solution would meet the requirement to protect the data in transit, but it would require the agency to deploy, secure, and maintain a public web server connected to the internet. Additionally, all HTTPS services would need to be reconfigured to require a new kind of authentication system, as currently it is assumed that anyone who can reach the internal services is authorized to use the services.
- **Application Layer Solution: File Encryption.** Instead of encrypting communications, an application layer solution could encrypt the data itself, which could then be transferred through non-encrypted communications. Using a public key from the agency, the external developers could encrypt their data and then transfer the data to the server over public networks. The data on the server could be decrypted by the developers as needed. Although file encryption is a reasonable solution for transferring files to the agency's server, it is not well-suited for protecting reports and other files that may be downloaded from the server by the external organizations. Such files would need to be encrypted so the external organizations could decrypt them. As developers join or leave the agency, or other changes occur to the set of valid keys, all files would need to be encrypted using the

new set of keys. The agency could establish a shared key for all external developers, but this would increase the risk of unauthorized access, reduce accountability, and still require considerable maintenance effort, such as distributing new keys in an out-of-band manner.

After further investigations into security, ease of deployment, and cost, the agency selects the network layer VPN solution and chooses to use its existing remote access architecture. It is important to note that this solution protects traffic only between the external developers' laptops (at home or on the corporate WiFi) and the main office's VPN gateway; the traffic between the VPN gateway and the local servers is not encrypted, unless the developers use the SSH protocol to provide encryption.

9.2.2 Designing the Solution

The solution is based on the agency's existing FreeBSD Internet router and will only require installing the additional strong Swan IPsec software to become an IPsec VPN gateway. The router is lightly utilized, so an additional VPN device is not needed for the external developers' usage. The strongSwan IPsec implementation supports EAP-TLS for authentication, which can use the same AAA backend as the WiFi WPA2 solution. Certificates can be easily added and revoked when developers join or leave the agency. The VPN requirement for the internal WiFi network can be rolled out as optional first and made mandatory later by deploying a packet filter on the firewall that connects the WiFi access point to only allow IKE and ESP packets from the WiFi clients.

Figure 18 illustrates the planned design for the VPN architecture. The internal WiFi and the remote access clients are considered external (and insecure) networks and are on a different segment from the internal networks of the main office. The strategy is to establish an IPsec tunnel from the external devices to connect to the main office VPN router. Data sent between the developers' laptops and the VPN router will be encrypted, while data between the VPN router and the internal servers (A, B, and C) will not. The tunnel will stay intact until the external system or the VPN router manually terminates the tunnel, or the connection is inactive for a certain period of time. The VPN router and VPN client software on the developers' laptops support UDP encapsulation and MOBIKE, so remote clients that are on NAT networks or have multiple interfaces (WiFi and mobile data) can negotiate UDP encapsulation and MOBIKE to use the IPsec solution.

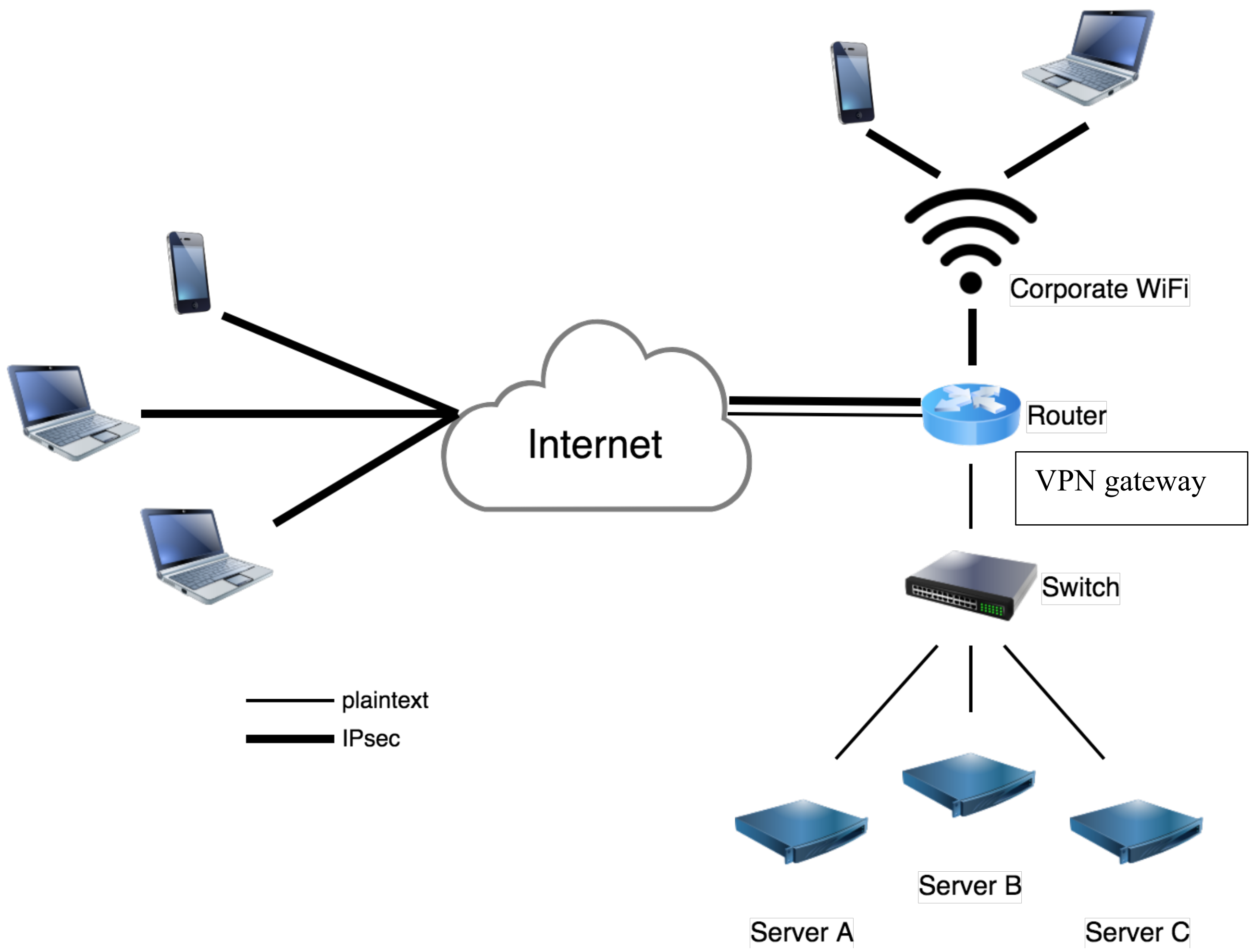

Figure 18: Remote Access VPN for Protecting Communications

After designing the architecture, the company next considers other elements of the design and makes several decisions, including the following:

- **Authentication.** In the actual deployment of the solution, the clients will be authenticated through digital certificates issued by the company's CA. The VPN router will be provisioned with a machine certificate. The certificates will be installed on the developers' laptops when these devices are locally present at the office. The IPsec client software will be configured to use the digital certificate as a user-based certificate, as this would not require any administrator privileges. When a tunnel needs to be established, the client will send its user certificate using EAP-TLS to the VPN gateway for authentication as part of the IKE exchange. The strongSwan IPsec software in the VPN gateway will act as a AAA server initially. When the company extends the solution to multiple VPN gateways for remote access to a number of remote access locations, a separate AAA backend will be set up to handle the EAP-TLS authentication. The VPN gateway will send its certificate via IKE to the remote clients as a machine certificate, so the clients do not need to contact the AAA server to authenticate the VPN's server certificate. Instead, the client uses the CA certificate to validate the VPN gateway certificate and that this certificate matches the IKE ID of the VPN gateway.

- **Encryption and Integrity Protection Algorithms.** The VPN gateway supports multiple encryption algorithms for IKE and ESP, including AES-CBC and AES-GCM. Since not all IKEv2 clients support AES-GCM for IKE, the gateway will also allow AES-CBC with HMAC-SHA-2 for IKE. However, since most IKEv2 clients support AES-GCM for ESP, the server normally does not permit AES-CBC with HMAC-SHA-2 as a default for ESP because that would put an additional load on the server.

- **Packet Filters.** To restrict the external developers' usage as much as possible, the IPsec packet filters should be configured to permit only access to the development network over the VPN tunnel. This would ensure that the agency's internal network is minimally impacted by the remote VPN clients.

- **Split Tunneling.** The IPsec client configuration could offer split tunnel configurations. Since the developers' laptops are issued for agency use only, their configurations do not allow split tunneling. The split tunnel configuration would also not make sense on the corporate WiFi, since all traffic will always first reach the corporate gateway regardless, so it makes sense to encrypt everything for the additional security it provides in case the native WiFi link layer security is compromised. For mobile phones, the IPsec configuration could allow split-tunnel configurations, as the network traffic generated by different applications on a phone are usually isolated from each other, and the VPN could be provisioned in such a way that only the corporate application is allowed to send traffic over the corporate VPN tunnel.

9.2.3 Implementing a Prototype

The VPN gateway administrator performs the following steps to configure and test a prototype of the IPsec solution between an external test system and the FreeBSD VPN gateway. Section 9.2.3.1 describes the configuration of the VPN gateway device, while Section 9.2.3.2 describes the external system's configuration. The testing of the whole solution is detailed in Section 9.2.3.3.

9.2.3.1 Configuring the Server

The administrator performs the following steps to configure the FreeBSD VPN gateway for use with strongSwan. It is assumed that there is an existing CA system that can issue certificates.

1. **Create a separate certificate for each device.** Device certificates use a subjectAltName (SAN) for the FQDN based on the user, a user-device@example.com like syntax, or a random globally unique identifier (GUID). For maximum compatibility, it will also set the EKU attribute for serverAuth.

2. **Create a VPN gateway machine certificate.** This certificate must have the full DNS hostname as SAN included with the certificate. Because the gateway has a static IP, a SAN for the IP address is added as well. For maximum compatibility, the EKU attribute for serverAuth is set as well.

3. **Configure global VPN server parameters.** The global parameters in the configuration files in the `/usr/local/etc/strongswan.d/` directory are reviewed. The system administrator decides to set logging to use a file instead of the default syslog.

4. **Configure the VPN server's IPsec connection and EAP-TLS RADIUS backend.** A new configuration file `remote-access.conf` is created in the `/usr/local/etc/swanctl/ipsec.d/` directory. It contains the server's IKEv2 parameters, such as the IKE ID, public IP address, local subnet (0.0.0.0/0 and/or ::0), configuration for DNS servers, lease IP addresses for clients, and tunnel. The radius server is located at IP address 10.10.10.10.

```
# /usr/local/etc/swanctl/ipsec.d/remote-access.conf
connections {
   remote-clients-eap {
      local_addrs = 192.0.2.1
      local {
         auth = pubkey
         certs = vpn.example.gov.pem
         id = vpn.example.gov
      }
      remote {
         auth = eap-tls
      }
      children {
          net {
              local_ts = 0.0.0.0/0
              updown = /usr/local/libexec/ipsec/_updown iptables
              esp_proposals = aes256gcm256-ecp256, aes256gcm256-
modp2048
          }
      }
      version = 2
      send_certreq = no
      proposals = aes256gcm256-prfsha2-ecp256, aes256-sha256-
modp2048
   }
}

pools {
  connections_pool {
      addrs = 10.11.0.0/16
  }
}
```

The EAP-TLS configuration is configured in strongswan.conf by editing the libtls{} and plugins{} section:

```
# /usr/local/etc/strongswan.conf
```

```
        plugins {
            eap-radius {
                secret = XXXXXXXXX
                server = 10.10.10.10
            }
        }

    libtls {
         suites = TLS_DHE_RSA_WITH_AES_128_GCM_SHA256,
                        TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
    }
```

5. **Ensure that the VPN service is started**. To ensure the strongSwan IKE daemon is started when booting the system, the file `/etc/rc.conf` is updated and the server is rebooted as a test.

6. **Create provisioning profiles for those IKEv2 clients that support it**. Using provisioning profiles can save a lot of time for the administrator and make it easier on the users to configure their system for IPsec. Unfortunately, not all common IKEv2 clients support this. The administrator uses the vendor enterprise tools from Apple, Microsoft, and others to generate profiles for easy installation.

7. **Update the firewall settings**. The firewall settings need to be updated to allow the IKE and IPsec traffic and to allow the decrypted traffic to be inspected and then forwarded to the right interfaces. The `/etc/rc.conf` file is updated to set `firewall_enable="YES"`, and the file `/etc/rc.firewall` is updated to allow protocol 50, UDP port 500, and UDP and TCP port 4500.

9.2.3.2 Configuring the Clients

After completing the VPN gateway configuration, the administrator configures an externally located test system to be an IPsec client. The steps performed to achieve this are as follows:

1. **If required, install IKEv2 software on the device.** On most phones and laptops, an IKEv2-based IPsec client comes pre-installed. Because some people inside the company use Android-based phones, and they do not have native support for IKEv2, the strongSwan IKEv2 client is installed on them.

2. **Configure the IPsec clients.** Each vendor's IPsec client has its own type of configuration. Clients that support provisioning can usually install a profile configuration file from universal serial bus (USB) media or an email attachment. Such profiles are usually encrypted by a password to ensure that the file can be sent over an insecure network. If provisioning is not supported, the configuration menu on the client will have an option to add a "VPN configuration". This configuration will then ask for the remote VPN server's DNS name, the type of configuration required, and some optional information. Some IPsec clients have an option to import a certificate bundle, while other IPsec clients require the user to import certificates separately from the VPN connection.

Certificates usually are transported using the PKCS#12 format, which consists of an encrypted bundle consisting of a certificate, private key, and CA certificate that are protected by symmetric key wrapping using a key derived from a strong password.

3. **Test the tunnel settings.** Once the parameters have been entered, the administrator starts the VPN connection.

9.2.3.3 Testing the Solution

After completing the configuration of the VPN router and the external test clients, the VPN gateway administrator tests the solution to ensure that the external system can successfully establish a secure tunnel to the VPN router and transfer encrypted traffic through the tunnel. While ping commands are a good initial test to see if things appear to be working, it is not enough, as these packets are unusually small and will give no indication whether a large TCP stream will work as well. Using a web browser to generate traffic is a better test. If the remote access server provides both IPv4 and IPv6 lease IP addresses to the VPN clients, both types should be verified to work properly. Traffic to both the corporate servers and the Internet should be tested to ensure proper functioning of the (lack or presence of) split tunnel configuration.

Tests should also ascertain that the VPN gateway will only negotiate IPsec tunnels for the approved algorithm and will block traffic that is not encrypted. The administrator should monitor the VPN gateway's logs for errors that indicate problems with the connection. The gateway's log report generation tool can be useful when troubleshooting issues because it can indicate where connections are failing or where traffic is being dropped. The administrator also deploys a packet sniffer on the gateway or an external test device to confirm that the traffic is being protected.

MOBIKE is tested by using a phone that has mobile data and WiFi connectivity. The phone establishes a VPN connection to the VPN server using the WiFi interface. The WiFi interface is then disabled. The VPN connection should still be working. Logs on the VPN server can be checked to see if the VPN client's public IP address changed through a MOBIKE message. Re-enabling WiFi should cause the VPN client to switch back to WiFi, since that is usually the preferred connection, as it will be faster and cheaper.

9.2.4 Analysis

IPsec tunnels established from external systems to a trusted gateway can be effective for protecting sensitive information from eavesdroppers. Providing secure remote access for laptops, phones, or industrial equipment can be done using standard IKEv2 and IPsec software. Using the existing IPsec client software and IPsec gateway eliminates the need to purchase additional hardware or software and greatly reduces design and implementation time.

Reusing the remote access VPN architecture to provide additional protection to the local WiFi network requires less reliance on the WiFi hardware manufacturers and WiFi security protocols. The WEP and WPA2 link layer security protocols have been cryptographically broken on a few occasions, requiring protocol updates that are not always possible on older hardware models. Using an IPsec solution provides confidence that the WiFi network cannot be abused or broken into to gain access to the corporate network, as the WiFi network is as untrusted as any other host on the internet. Visitors to the office can be given guest internet access to the WiFi network using the link layer credentials without endangering the corporate network, as access to the corporate network is not possible from the office WiFi network without using the IPsec remote access VPN.

9.3 Remote Access to a Cloud Server Instance

An agency has outsourced some of its public facing web pages to a cloud provider. A number of virtual machines are used to provide the service from the cloud. This private cloud uses private IP addresses. The agency has one public IP address that terminates at the cloud provider. The cloud provider allows the agency to forward specific protocols and ports to one of its virtual machines. The agency forwards TCP port 80 and TCP port 443 to one of the virtual machines running the haproxy software configured as a service that load balances these connections to a number of virtual machine web servers. These web servers connect to another set of virtual machines running a database server. During peak seasons for this agency, the number of database and web servers can be increased to match demand. To update the database content on these virtual machines from the agency internal network, a VPN connection is desired. This would allow the database servers to be replicated from the agency's network to the private cloud.

The virtual cloud is using the IPv4 private space IP network 10.0.2.0/24. The cloud provider runs a virtual router on the IP address 10.0.2.254. Traffic for the cloud uses one of the cloud provider's public IP addresses, 192.1.2.78. This is the IP address for the agency's cloud webserver at cloud.example.gov. Web traffic using ports 80 and 443 to the IP address 192.1.2.78 uses NAT and is sent to the internal IP 10.0.2.2 running the haproxy service. The agency itself uses the private space IP network 192.168.0.0/16, but only wants select parts of their network to have direct access to the private cloud—192.168.103.0/24 and 2001:db8:0:2::/64. While the agency could get public IPv6 addresses for its virtual private cloud, it decides it would be safer to use private space IPv6 addresses as well, similar to how it rolled out private space IPv6 at the agency network for its database servers and workstation machines. The IPv6 private cloud will use 2001:db8:0:1::/64.

9.3.1 Identifying Needs and Evaluating Options

As there is no dedicated link between the agency and the cloud provider, link-based VPNs cannot be used. The agency also wants to keep the ability to move to another cloud provider, so it does not want to use the cloud provider's VPN solution. An additional advantage of using a virtual VPN server inside the private cloud is that all traffic inside the cloud provider's network, but outside the private cloud itself, would be encrypted. Only the virtual machines of the agency would be able to see the unencrypted traffic.

Using a network layer VPN would allow the agency to extend the solution by adding IPsec VPN tunnels to other cloud providers or new physical locations. It could extend the solution to building more VPN tunnels to other physical locations or other cloud providers. A VPN tunnel could even be used to move a single server to another cloud provider without reconfiguration of any other virtual servers in the private cloud.

9.3.2 Designing the Solution

Since the agency is using Linux-based virtual machines at the cloud provider, it will also use a Linux-based virtual machine as its VPN server in the private cloud. It decides to use the libreswan IPsec software that comes with the Linux distribution it is using for its cloud instances. The agency already has an enterprise Linux-based server as its internet access and firewall server, so it is decided to extend that server to build an IPsec VPN to the private cloud network. This enterprise Linux server is also using libreswan. See Figure 19 for illustration of the network setting.

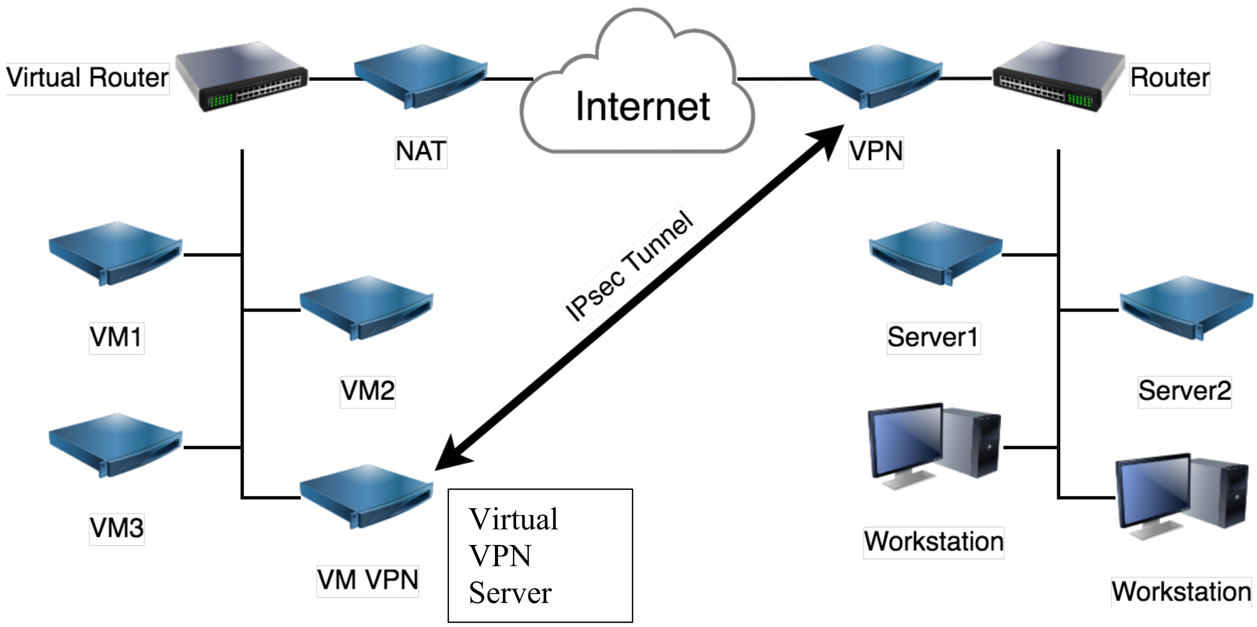

Figure 19: Remote Access to a Cloud-Based Virtual Network

After designing the architecture, the company next considers other elements of the design and makes several decisions, including the following:

- **Authentication.** Libreswan supports and defaults to using IKEv2. Since both VPN endpoints are controlled by the agency, it decides to use public keys for authentication without using certificates. This will prevent the situation where certificates would expire. Using public keys without a CA is also much simpler.
- **Encryption and Integrity Protection Algorithms.** Since both ends use the same enterprise Linux solution that supports libreswan running a cryptographic module operating in FIPS mode, it is decided to leave the IKE and ESP options with their default values. That means that the VPN will start out using AES-GCM with 256-bit keys for IKE and ESP, SHA-256 as the IKE PRF, and DH 14 with PFS. When NIST-approved algorithms change in the future, the Linux enterprise solution will update the libreswan software, and the configuration on the VPN servers will be automatically updated to use the new stronger algorithm requirements.

- **Packet Filters.** To restrict the VPN access to the cloud from the agency's internal network, it is decided that only workstations and servers at some specific IP addresses are allowed to have access to the private cloud, such as only two IPv4 networks and one IPv6 network for the developer workstations using 192.168.100.0/24 and the database servers using the IPv4 range 192.168.103.0/24 and the IPv6 range 2001:db8:0:2::/64.
- **MTU and TCP settings.** It is not known exactly how many layers of encapsulations are happening at the cloud provider and at the agency's Internet service provider (ISP) itself. It is known that a digital subscriber line (DSL) service adds at least one encapsulation using PPP at the data link layer. To prevent unnecessary fragmentation and possible flow issues on the database and remote SSH login connections that will use TCP, it is decided to use TCP MSS clamping and slightly reduce the MTU for packets across the VPN connection.

9.3.3 Implementing a Prototype

A new virtual machine instance is requested from the cloud provider. The cloud security policy is updated to temporarily allow SSH connections from port 2222 of the public IP to reach the SSH port 22 on the new VPN virtual machine. An administrative SSH public key is configured to be allowed to log in to the server, and password-based SSH logins are disabled.

Using SSH to remotely log in, the virtual machine is configured as a VPN gateway. The configuration options of libreswan uses the terms *left* and *right*. The left side of our diagram is the virtual machine VPN and the administrator uses left* options to refer to it. Similarly, the agency's office VPN is on the right side of the diagram and denoted by *right*.

9.3.3.1 Configuring the VPN gateways

The cloud instance and the office gateway are prepared to run libreswan by:

- Updating the operating system: yum update
- Installing Libreswan: yum install libreswan
- Initializing Libreswan's NSS database: ipsec initnss
- Generating a new host key: ipsec newhostkey --output /etc/ipsec.d/hostkey.secrets
- Using the host key's ckaid from the previous step to obtain the public key:
    - On the cloud instance: ipsec showhostkey --left --ckaid <ckaid>
    - On the office gateway: ipsec showhostkey --right --ckaid <ckaid>
- Creating the configuration file cloud-office.conf with a *conn* definition for the connection named cloud-office-ipv4 and cloud-office-ipv6, then uploading it to both VPN servers and placing it in the directory */etc/ipsec.d/*

- Customizing the left= entry on both servers, as indicated in the configuration file below
- Updating firewall rules to allow traffic from the subnets and exempt these IP destination ranges from being NAT'ed. Adding a firewall rule for TCP MSS clamping.[81]
- Enabling IP forwarding on the cloud instance. The built-in rp_filter is disabled to avoid false positives, otherwise the kernel will drop or try to redirect traffic due to the encrypted and decrypted traffic using the same (single) virtual ethernet card.

```
# /etc/ipsec.d/cloud-office.conf

conn cloud-office-base
    # On the cloud gateway, use left=%defaultroute to pick up its
    # internal IP address
    # left=%defaultroute
    # on the office gateway, use left=<IP of the cloud's public IP>
    left=192.1.2.78
    leftid=@cloud-vpn
    leftrsasigkey=<value from above ipsec showhostkey  --left command>
    right=office-gw.example.gov
    righted=@office-gw
    leftrsasigkey=<value from above ipsec showhostkey  --left command>
    ikev2=insist
    mtu=1440

conn cloud-office-ipv4
    also=cloud-office-base
    leftsubnets=10.0.2.0/24
    rightsubnets=192.168.100.0/24,192.168.103.0/24
    auto=add

conn cloud-office-ipv6
    also=cloud-office-base
    leftsubnet=2001:db8:0:1::/64
    rightsubnet=2001:db8:0:2::/64
    auto=add
```

9.3.4 Testing the Solution

The administrator is at the office, so they use SSH to log in to a third-party host that is neither behind the office VPN nor within the private cloud. From that machine, they use SSH to log in to the cloud instance VPN server. Now if the IPsec tunnels fail to come up due to a misconfiguration and drop all packets between the two locations, they are not locked out from fixing the configuration.

[81] Different Linux systems use different firewall management tools. These could be based on iptables, firewalld, or shorewall. Consult the vendor's documentation.

- On both ends, start libreswan: `systemctl start ipsec`
- On one end, start the IPv4 connection manually: `ipsec auto --up cloud-office-ipv4`
- If the connection fails, it should show what happened. Consult the libreswan documentation and Frequently Asked Questions (FAQ) if the error is unclear.
- Once the connection establishes, a ping from one of the workstations in the office can be used to test: ping 10.0.2.78.
- Once confirmed to work, a database replication is started to test performance.
- Byte counters on the tunnel are confirmed using the command `ipsec trafficstatus`
- Next, the IPv6 connection can be brought up and tested: `ipsec auto --up cloud-office-ipv6`

With the tunnels have been confirmed to be working correctly, the configuration is updated to automatically start the tunnels when the libreswan IPsec service starts by changing `auto=add` to `auto=start`. The ipsec service is enabled to start at bootup on both gateways using the command `systemctl enable ipsec`.

The port forwarding for SSH into the private cloud is disabled using the cloud management tools to prevent the virtual machines from being scanned by attackers from the internet. SSH access is still possible, as long as the connections are made from the office through the VPN connection.

9.3.5 Analysis

A private cloud can be safely accessed remotely by adding a virtual machine acting as a VPN gateway. The private cloud can be used and protected just like physical servers at a data center. Additionally, by requiring the use of the VPN, remote access control can be further limited to legitimate sources and prevent the cloud instances from being susceptible to port scanning attacks via port forwarding on the public IP through which the private cloud is reachable.

In the future, the VPN configuration can be extended to connect to other private clouds or other data centers. It can also be extended to act as a remote access VPN for developers so they can safely connect to the private cloud from their laptops even if not at the office.

Both IPv4 and IPv6 can be used, even if the cloud provider does not provide IPv6 themselves. This allows the agency to be proactive and compliant to regulations that mandate IPv6 readiness on all their equipment.

9.4 Cloud Encryption

A large enterprise has a number of data centers and is renting virtual machines from various cloud providers. While it has connected the different networks using a gateway-to-gateway architecture, it is concerned that traffic within these networks is not encrypted. Furthermore, its global size makes it hard to monitor and ensure that all fiber cables and satellite links it deploys use proper data link security. For example, the agency might be renting an inter-city fiber cable to create a VLAN network that uses MPLS to connect a number of physically separate locations. It might be using MPLS without any link security. As nodes would not be aware when traffic would be local or would be traversing a fiber cable, such a network is vulnerable to unauthorized wiretaps. The desire is to encrypt as much traffic as possible between all nodes worldwide without creating chokepoints or single point of failures for encryption.

9.4.1 Identifying Needs and Evaluating Options

The goal of the project is for all network traffic to be protected by network layer-based security to ensure that a compromised segment of its global data link security would not result in plaintext data being obtained by an attacker. As the goal is to encrypt all traffic, it is infeasible to perform this at the application layer. While part of the traffic can be protected by the application's use of the TLS protocol, this would not fulfill the requirement of ensuring that all traffic is encrypted at the network layer.

As a first step for encrypting traffic between any two nodes, each node needs to have an identity. With various cloud deployments using virtualization and container technologies, it means that nodes are created and destroyed continuously. A provisioning system will need to be able to create and revoke identities for authorization. Ideally, the existing provisioning system that creates virtual machines and containers will be extended to give these services their cryptographic identity.

To comply with legal requirements and corporate compliance policies, specific traffic between certain nodes must be monitored and stored. This traffic must be exempted from the network-wide encryption policy.

Due to the sheer size of the project, it is inevitable that individual exceptions to policies need to be accommodated. A phased approach will be required where individual network managers can prepare their data center or cloud deployments for participation in the network-wide mesh encryption solution.

9.4.2 Designing the Solution

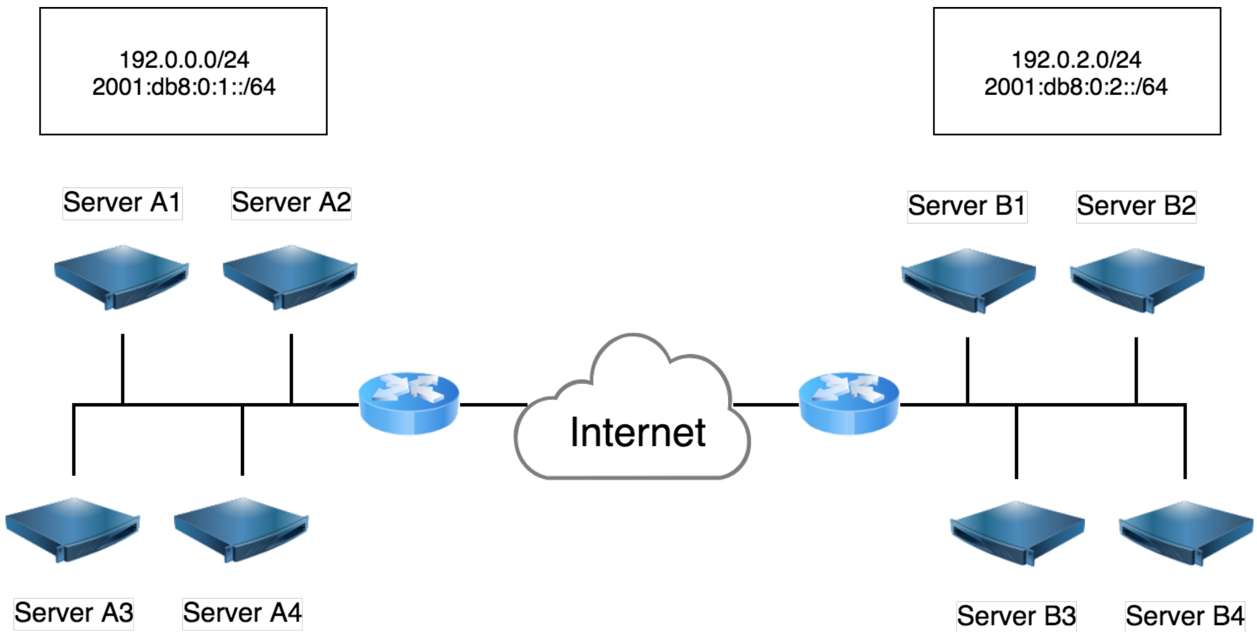

Figure 20: Mesh Encryption Using Opportunistic IPsec

Connection Establishment. A packet triggered IPsec based solution is chosen. Since IPsec can be easily added to physical servers, virtual servers, and container-based instances, the solution should work across most of the global infrastructure.

Authentication. As certificates are already used to identify many services, the IPsec nodes will be authenticated using machine certificates. At a later date, DNSSEC-based authentication using public keys will be evaluated, which will reduce the overhead of running a CA and remove the need for certificate renewal.

Confidentiality and Integrity. As it is expected that some nodes will have hundreds of IPsec connections, it is important to pick the most optimum cryptography. AES-GCM with 128-bit keys is used for IKE and IPsec. For DH, the DH group 19 is used to provide 128 bits of security strength for the key exchange.

Lifetime and Idletime. Standard IKE SA and IPsec SA lifetimes are used, although since these are not negotiated, individual managers can tune these later to optimum values depending on their traffic patterns. Similarly, idletimes are set to 15 minutes to prevent the accumulation of too many idle IKE and IPsec sessions per host, and idletimes can be tuned at a later stage as well.

IPsec Mode. As all networks are already connected via IPsec gateways, no NAT is deployed and the IPsec connections can use the transport mode, resulting in a larger effective MTU than if an IPsec tunnel mode was used. Transport mode also prevents a node from creating a custom policy covering more than itself.

9.4.3 Implementing a Prototype

To make a realistic deployment prototype, the company decides to use two networks normally reserved as staging servers that test new code before it is deployed into production. Two staging networks at different data centers are used. These two networks are already connected in a gateway-to-gateway architecture. In a first step, servers in network A and servers in network B will each be configured for mesh encryption to their local nodes only. Once the mesh IPsec encryption is functional in one network, and the mesh IPsec encryption is functional in the other network, the mesh will be extended to incorporate both networks in a single mesh configuration. This allows for further testing of IPsec-in-IPsec packets when a server from network A starts an IPsec connection to a server in network B.

- The opensource *ansible* software provisioning system is extended to create a PKCS#12 certificate for each new virtual machine that is created for network A and network B.
- An opportunistic IPsec configuration file is created and added to the ansible script to be installed on new virtual machines deployed in networks A and B.

```
# /etc/ipsec.d/mesh.conf
conn private-or-clear
       left=%defaultroute
       leftcert=provisioned-cert
       leftid=%fromcert
       rightid=%fromcert
       rightrsasigkey=%cert
       right=%opportunisticgroup
       type=transport
       failureshunt=passthrough
       auto=ondemand
```

- As part of the new virtual machine provisioning, libreswan is installed, and the generated file containing the PKCS#12 bundle with *friendly_name* “provisioned-cert” is imported into libreswan using the `ipsec import` command.
- Opportunistic IPsec is enabled using the “private-or-clear” connection by adding the IP network ranges of the participating networks to the file:

  `/etc/ipsec.d/policies/private-or-clear`:

```
# /etc/ipsec.d/policies/private-or-clear
192.0.0.0/24
192.0.2.0/24
2001:db8:0:1::/64
2001:db8:0:2::/64
```

9.4.4 Testing the Solution

Traffic is generated and nodes are inspected using the `ipsec trafficstatus` command. Once the basic mesh encryption is working, more advanced scenarios are tested.

- A single IP address is added to the exception policy `/etc/ipsec.d/policies/clear` to confirm communication only happens in cleartext.
- Both network A and network B add each other's IP ranges to the policy file for opportunistic IPsec in `/etc/ipsec.d/policies/private-or-clear` to test mesh encryption across the two networks.
- Some servers are tested with a policy in `/etc/ipsec.d/policies/private`, which mandates IPsec encryption.
- TCP streams are tested between network A and B to confirm that there are no issues with double encryption (a VPN over another VPN) and packet sizes.
- An IPsec mesh IP connection is triggered, and no more traffic is sent between the nodes. The connection is monitored to be expired due to idleness within the configured timeframe.
- To harden against attacks where one compromised server takes over the IKE identity of another server while using its non-matching certificate, the `dns-match-id` option is enabled. After testing that the mesh connections still work, one host is configured with another host's certificate, and a mesh connection is attempted again. The connection is tested for proper rejection.

9.4.5 Analysis

The additional provisioning to add IPsec to the virtual machines and containers are minimal and working. However, it was found that packet filters on the networks were no longer able to filter traffic because most of it was encrypted. This necessitated an extension of the provisioning system to push firewall rules to each virtual machine and container.

While the initial deployment of using certificates works, using raw keys in DNSSEC would work better for a large-scale deployment, but it would require a way to update DNS dynamically after generating host keys for newly generated virtual machines and containers. A follow-up project is planned for a DNSSEC-based deployment.

10 Work In Progress

This section briefly discusses some of the future directions of IPsec. At this time, the IETF is working on various IKE and IPsec extensions. This section provides a brief discussion of the new standards and pointers to additional information.

10.1 Support for Multicast and Group Authentication

Multicast traffic refers to sending a packet to an IP address that is designated as a multicast address; one or more hosts that are specifically interested in the communication then receive copies of that single packet. This differs from *broadcast traffic*, which causes packets to be distributed to all hosts on a subnet, because multicast traffic will only be sent to hosts that are interested in it. Multicasting is most often used to stream audio and video. For the sender, there are two primary advantages of using multicast. First, the sender only needs to create and send one packet, instead of creating and sending a different packet to each recipient. Second, the sender does not need to keep track of who the actual recipients are. Multicasting can also be advantageous from a network perspective, because it reduces network bandwidth usage.

RFC 4301 [40] describes IPsec processing for multicast traffic. RFC 5374 [82] extends the IKEv1 protocol to apply to groups and multicast traffic. It defines a new class of SAs (Group Security Associations, GSAs) and additional databases used to apply IPsec protection to multicast traffic [83]. The secret key to these GSAs is distributed to the group members. Once a member leaves the group, any secret key shared with other members has to be replaced with a new group key unknown to the group member that just left. For large groups that always have members joining and leaving, this can be complicated.

At the time of writing, IKEv2 does not support this, but a draft document is under development to add this support [84]. It defines a new G-IKEv2 extension that conforms with the Multicast Group (MEC) Security Architecture [83] and the Multicast Security (MSEC) Group Key Management Architecture [85]. G-IKEv2 replaces Group Domain of Interpretation (GDOI) [86], which defines a similar group key management protocol for IKEv1.

10.2 Labeled IPsec

Labeled IPsec is a mechanism to convey a security label or context that is associated with an IPsec stream. Both endpoints can apply further restrictions on the type of traffic allowed to be transmitted via the IPsec connection. Some vendors had a proprietary extension to IKEv1 to support labeled IPsec. The IETF is currently working on a draft to add this extension to IKEv2. The extension takes the form of an additional Traffic Selector with the security context that needs to be matched. This work is discussed in [87].

10.3 ESP Implicit IV

For IoT devices, as well as other battery-powered network devices, there is a desire to reduce the number of bytes sent over a network to save battery power. When IPsec is deployed using an AEAD such as AES-GCM, each packet contains an IV, also called a nonce. This value must be unique but may be predictable. The recommended implementation is to use a simple counter. However, the ESP protocol itself already has a counter, which is used to defend against replay attacks. A proposal is being developed by the IETF to define AES-GCM and AES-CCM variants that omit sending the AEAD IV and use the ESP replay counter instead. These variants are only defined for ESP algorithms, not the IKE algorithms. This work is discussed in [88].

10.4 The INTERMEDIATE Exchange

Classic DH key exchanges could become vulnerable to quantum computing attacks. There is a need to replace the DH key exchange with a quantum-safe key exchange. Current proposals for such algorithms all require the use of large public keys that need to be exchanged in IKE during the IKE_SA_INIT phase. During this phase of the exchange, IKEv2 fragmentation cannot yet be used, because a confidential channel that can identify fragments as legitimate has not yet been established. A new INTERMEDIATE exchange is placed between the IKE_SA_INIT and IKE_AUTH exchanges, which can support fragmentation. This work is discussed in [89].

10.5 IPv4 and IPv6 Support in Remote Access VPNs

The telecom networks (LTE/5G) can provide notifications about whether a network connection should be attempted with IPv4, IPv6, or both. However, IKEv2 does not offer a similar notification structure or rich enough error notification for clients to determine if they should attempt IPv4 or IPv6 only, or address both families (IPv4 and IPv6) for use with IPsec. A new draft is proposing to clarify this, for better integration of 3GPP standards with IKEv2. This work is discussed in [90].

10.6 Post Quantum Key Exchange

Once there are quantum-safe key exchange algorithms that can replace the classic DH key exchanges, the IKEv2 protocol will need to be extended to support this. One suggestion is to keep the existing (EC)DH exchange and add on one or more quantum-safe key exchanges to the protocol in such a way that the resulting hybrid key exchange is at least as strong as the strongest component. This guarantees that even if a quantum-safe algorithm candidate is used and later turns out to be unsafe, the security of the connection is still at least as strong as the known classical DH key exchange. This design also ensures that a NIST-approved IPsec implementation that adds a quantum-safe algorithm for protection still complies to all current NIST requirements. This work is discussed in [91].

Appendix A—Required Configuration Parameters for IKE and IPsec

The table below can be used as a checklist of information required to set up a gateway-to-gateway VPN tunnel. Example values are NIST approved and ranked from most preferred to least preferred. IKE and IPsec lifetimes and maximum bytes are local values only and not negotiated.

Information	Value(s)	
Local network name:		
Remote network name:		
IKE parameters:		
IKE version: (e.g., IKEv2, IKEv1)		
IKEv1 mode: (if applicable) (e.g., Main, Aggressive)		
Local ID: (type can be: IPv4, IPv6, FQDN, email or DN[82]. Default is often IPv4/IPv6)	type:	value:
Local Peer IP address or DNS name:		
Remote Peer ID: (type can be: IPv4, IPv6, FQDN, email or DN[83]. Default is often IPv4/IPv6)	type:	value:
Remote Peer IP address or DNS name:		
Encryption algorithm(s): (e.g., AES-GCM, AES-X**CBC, 3DES (deprecated))**		
Encryption key size(s): (e.g**., 128, 192, 256)**		
Integrity algorithm(s): (None when using an AEAD such as AES-GCM) (e.g., HMAC-SHA-2-512, HMAC-SHA-2-384, HMAC-SHA-2-256)		
Diffie-Hellman Group: (e.g., DH 19 (ecp256), DH 20 (ecp384), DH 21 (ecp512), DH 14 (modp2048), DH 15 (modp3072), DH 16 (modp4096), DH 17 (modp6144), DH 18 (8192), DH 23, DH 24, DH 25 (ecp192), DH 26 (ecp224)	group(s):	PFS (yes/no):
Authentication type: (e.g., ECDSA >=256, RSA-Probabilistic Signature Scheme (RSA-PSS) (>= 2048), RSA-v1.5 (legacy) **(>=2048), PSK)**		
If PSK: (minimum 32 random characters)		
IPsec parameters:		
DH Group for PFS: must be equal strength (or stronger) as IKE above		
Local network(s):		
Remote network(s):		
Encryption algorithm(s): (e.g., AES-GCM, AES-X**CBC, 3DES (deprecated)**		
Encryption key size(s): (e.g**., 128, 192, 256)**		
Integrity algorithm(s): (None when using an AEAD such as AES-GCM) (e.g., HMAC-SHA-2-512, HMAC-SHA-2-384, HMAC-SHA-2-256)		

[82] When using a certificate, instead of specifying its DN, it is often easier and more robust to use its SubjectAltName.

[83] When using a certificate, instead of specifying its DN, it is often easier and more robust to use its SubjectAltName.

Appendix B—Policy Considerations

As mentioned in Section 6, organizations should develop IPsec-related policies and use them as the foundation for their IPsec planning and implementation activities. This appendix presents examples of common IPsec-related policy considerations that address the confidentiality, integrity, and availability of the IPsec implementation, as well as the conditions constituting its acceptable use. The appendix focuses on policy considerations for three sample scenarios: a gateway-to-gateway VPN between two offices of a single organization, a gateway-to-gateway VPN between two business partners, and a remote access VPN for telecommuting employees of an organization.

The examples provided in this appendix are intended only to provide a starting point for developing IPsec-related policy. Each organization needs to develop its own policy based on its environment, requirements, and needs. Also, many of the policy considerations in this section might already be addressed through an organization's existing policies. The examples in this appendix are not comprehensive; organizations should identify additional IPsec-related considerations that apply to their environments.

B.1 Communications with a Remote Office Network

In this scenario, an organization wants to establish an IPsec VPN to protect communications between its main office's network and a remote office's network. This VPN would be created by having the organization deploy and manage an IPsec gateway on each network and configuring the gateways so that they protect communications between the networks through an IPsec tunnel as needed. This scenario assumes that the same policies apply to the main office and remote office networks. The policy consideration examples listed in this section are divided into two groups: items specific to the IPsec gateway devices and management servers, and items specific to the hosts and people using the IPsec tunnel.

B.1.1 IPsec Gateway Devices and Management Servers

Items that are typically part of VPN policy for gateway devices and management servers include the following:

- Roles and responsibilities related to IPsec gateway operations.
- Definition for where VPN tunnels should terminate (e.g., between the border router and firewall, on the firewall).
- Security controls that are required to monitor the unencrypted network traffic, such as network-based intrusion detection systems or antivirus software, and their acceptable placement in the network architecture relative to the IPsec gateways.
- Authentication requirements for IPsec gateway administrators (e.g., two-factor authentication). This could also include requirements to change all default manufacturer passwords on the gateways and management servers, to have a separate account for each administrator, to change administrator passwords on a regular basis, and to disable or delete an administrator account as soon as it is no longer needed.

- Authentication requirements for IPsec tunnel users, if any. This should include a requirement for how often user accounts are audited.
- Authentication requirements for the IPsec gateway devices.
- Security requirements for the IPsec gateway devices and IPsec management servers. For example, an organization might require a firewall to be deployed between an IPsec gateway device and its users and be configured to block all traffic not explicitly approved for use with the IPsec implementation. An organization might also require certain security controls on the IPsec gateway devices and management servers, such as host-based firewalls and antivirus software.
- What information should be kept in audit logs, how long it should be maintained, and how often it should be reviewed.
- Requirements for remediating vulnerabilities in the IPsec gateway devices and management servers.
- Which types of traffic should be protected by IPsec tunnels, and what types of protection should be applied to each type of traffic.
- What types of protection should be applied to communications between an IPsec gateway and an IPsec management server.

B.1.2 Hosts and People Using the IPsec Tunnel

Because the hosts and people using the IPsec tunnel are assumed to be using the organization's equipment and networks, existing policies regarding acceptable use of the organization's systems should already address most policy needs regarding IPsec tunnel use. Examples include host access requirements (e.g., authentication) and vulnerability mitigation requirements (e.g., patching OS and application vulnerabilities). Existing policy also typically specifies technical controls that must be used on each host, as well as the minimum acceptable configuration for the technical controls.

B.2 Communications with a Business Partner Network

In this scenario, an organization wants to establish an IPsec VPN to protect certain communications between a system on its network and a system on a business partner's network. This VPN would be created by having each organization deploy and manage an IPsec gateway on its own network and configuring the gateways so that they protect communications between the organizations through an IPsec tunnel. This section focuses on the formal agreements made between the two organizations, and also summarizes policy considerations related to the organization's IPsec gateway and management server, and the people and hosts within the organization using the IPsec tunnel.

B.2.1 Interconnection Agreement

Federal policy requires Federal agencies to establish interconnection agreements for connections with business partners.[84] Specifically, OMB Circular A-130, Appendix III, requires agencies to obtain written management authorization before connecting their IT systems to other systems, after determining that there is an acceptable level of risk of doing so. The written authorization should define the rules of behavior and controls that must be maintained for the system interconnection and should be included in the organization's system security plan. It is critical that the organization and the business partner establish an agreement between themselves regarding the management, operation, and use of the interconnection, and that they formally document this agreement. The agreement should be reviewed and approved by appropriate senior staff from each organization.

An interconnection agreement is typically composed of two documents: an Interconnection Security Agreement (ISA) and a Memorandum of Understanding or Agreement (MOU/A).[85] The ISA is a security document that specifies the technical and security requirements for establishing, operating, and maintaining the interconnection. It also supports the MOU/A between the organizations. Specifically, the ISA documents the requirements for connecting the systems, describes the security controls that will be used to protect the systems and data, contains a topological drawing of the interconnection, and provides a signature line. The MOU/A documents the terms and conditions for sharing data and information resources in a secure manner. Specifically, the MOU/A defines the purpose of the interconnection; identifies relevant authorities; specifies the responsibilities of both organizations; and defines the terms of agreement, including the apportionment of costs and the timeline for terminating or reauthorizing the interconnection. The MOU/A should not include technical details on how the interconnection is established or maintained; that is the function of the ISA.

Items that are typically part of the ISA include the following:

- The information and data that will be made available, exchanged, or passed in only one direction between the systems through the IPsec gateways, and the sensitivity of that information
- The services offered over the VPN by each organization, if any
- The user community that will be served by the VPN
- A description of all system security technical services pertinent to the secure exchange of data between the systems; examples include the use of NIST-approved encryption

[84] NIST SP 800-47, *Security Guide for Interconnecting Information Technology Systems*, contains information on interconnection agreements, as well as extensive guidance on planning, establishing, maintaining, and disconnecting system interconnections, and developing an interconnection agreement [92].

[85] Appendices A and B of NIST SP 800-47 [92] contain detailed guidance on developing an ISA and an MOU/A as well as a sample of each. Rather than develop an ISA and MOU/A, organizations may choose to incorporate this information into a formal contract, especially if the interconnection is to be established between a Federal agency and a commercial organization. Also, in some cases, organizations may decide to use established organizational procedures for documenting the agreement, in lieu of an ISA and MOU/A.

mechanisms to protect communications, and the use of physical security controls to restrict access to the IPsec gateway devices and the systems

- A summary of the behavior expected from users who will have access to the interconnection; for example, each system is expected to protect information belonging to the other through the implementation of security controls that protect against intrusion, tampering, and viruses, among others
- The titles of formal security policies that govern each system
- A description of the agreements made regarding the reporting of and response to information security incidents for both organizations
- An explanation of how the audit trail responsibility will be shared by the organizations and what events each organization will log; this should include the length of time that audit logs will be retained.

Items that are typically part of the MOU/A include the following:

- A description of the systems communicating through the VPN
- A discussion of the types of formal communications that should occur among the owners and the technical leads for the systems
- A statement regarding the security of the systems, including an assertion that each system is designed, managed, and operated in compliance with all relevant federal laws, regulations, and policies.

As a foundation for the interconnection agreement, the organization should have general policy statements regarding the appropriate and necessary use of IPsec, so that it is clear when and how IPsec should be used to protect an interconnection.

B.2.2 IPsec Gateway Devices and Management Servers

Each organization should have policy statements that apply to the security and acceptable use of its IPsec gateway devices and management servers, as described in Appendix B.1.1.

B.2.3 Hosts and People Using the IPsec Tunnel

As described in Appendix B.1.2, existing policies regarding the acceptable use and security of the organization's systems should already address most or all policy needs regarding IPsec tunnel use by hosts and people within the organization.

B.3 Communications for Individual Remote Hosts

In this scenario, an organization wants to establish an IPsec VPN to protect communications between individual remote hosts used by telecommuting employees and its main network. This VPN would be created by having the organization deploy and manage an IPsec gateway on its main network. Employees' computers would be configured with IPsec clients that would establish tunnels with the IPsec gateway as needed to protect communications between the laptops and the organization's main network. This section presents policy consideration examples for remote hosts and the organization's IPsec gateway and management server.[86]

B.3.1 Remote Access Policy

The organization should have a remote access policy that includes IPsec usage by employees from both organization-controlled and other systems. The organization might also choose to have each employee that will use the IPsec implementation sign a remote access agreement or a copy of the remote access policy before being permitted to use the systems.[87]

IPsec-related items that are typically in a remote access policy include the following:

- A description of appropriate and inappropriate usage of the IPsec connection (e.g., forbidding personal use and forbidding use by other individuals)
- Pointers to other organization policies that apply to remote access, such as an acceptable use policy or a VPN policy
- Remote access authentication requirements, such as two-factor authentication or strong passwords
- Requirements for the networking profile of remote hosts; for example, the policy might forbid a host from being connected to the organization's network and another network at the same time, as well as forbidding split tunneling
- Minimum hardware and software requirements for remote hosts, including acceptable operating systems and patch levels
- Required security controls for remote hosts; this could also include required configuration settings for the controls, such as scanning all files before placing them onto the host

Organizations might also wish to require remote hosts to be checked automatically for vulnerabilities, malware, or other security problems immediately after establishing an IPsec connection. This should be stated in the remote access policy.

B.3.2 IPsec Gateway Devices and Management Servers

The organization should have policy statements that apply to the security and acceptable use of its IPsec gateway devices and management servers, as described in Appendix B.1.1. In addition, the organization might add policy statements specific to IPsec usage by remote hosts, such as the following:

- An automatic termination and disconnection of idle connections after X minutes

---

[86] Additional guidance on policy and security considerations for remote access users is available from NIST SP 800-46 [93].

[87] The policy and agreement could also be utilized for the use of the IPsec implementation by non-employees. Depending on the details of the policy and agreement, some changes might be needed to make them suitable for addressing non-employee use.

- A requirement for creating and maintaining a list of authorized users, disabling access for individual users as soon as it is no longer needed, and auditing the list of authorized users periodically.

Appendix C—Case Study Configuration Files

This section contains configuration files that are referenced in the Section 9 case studies.

C.1 Section 9.1 Case Study Cisco Configuration

The following lists the contents of one of the Cisco router configuration files used in the Section 9.1 gateway-to-gateway case study.

```
!
version 12.0
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname west.example.gov
!
enable secret 5 $1$rMk2$5fPj5s3CvYE35OSW0qkLD.
!
ip subnet-zero
no ip finger
!
crypto ikev2 proposal 1
 encryption aes-gcm 256
 prf sha256
 group 19
!
crypto ikev2 proposal 2
 encryption aes-cbc-256
 integrity sha256
 group 19
!
crypto ikev2 policy default
proposal 1
proposal 2
match fvfr any
!
crypto ikev2 profile default
identity local fqdn west.example.gov
match identity remote fqdn east.example.gov
authentication local pre-share key XXXXXXXXX
authentication remote pre-share key XXXXXXXXX
!
crypto ipsec transform-set 1 esp-gcm-128
mode tunnel
crypto ipsec transform-set 2 esp-cbc-128
mode tunnel
!
crypto map west-east 1 ipsec-isakmp
set peer 203.0.113.1
```

```
set transform-set 1 2
set pfs group19
set ikev2-profile default
match address 100
!
interface g1/1
 ip address 198.51.100.1  255.255.255.0
 no ip directed-broadcast
!
ip classless
ip route 0.0.0.0 0.0.0.0 20.20.20.20
no ip http server
!
ip access-list extended 100
 permit ip 192.0.0.0 0.0.0.255 192.0.2.0 0.0.0.255
 permit ipv6 2001:db8:0:1::/64 2001:db8:0:2::/64
!
line con 0
login
 transport input none
line aux 0
line vty 0 4
login
!
end
```

C.2 Section 9.1 Case Study Alternative Using strongSwan on FreeBSD

The following lists the contents of the same configuration as provided in Appendix C.1, but using strongSwan on FreeBSD:

```
# /usr/local/etc/swanctl/swanctl.conf
connections {
	west-east {
		local_addrs = 198.51.100.1
		remote_addrs = 203.0.113.1
		local {
			auth = psk
			id = west.example.gov
		}
		remote {
			auth = psk
			id = east.example.gov
		}
		children {
			net4-net4 {
				local_ts = 192.0.0.0/24
				remote_ts = 192.0.2.0/24
				esp_proposals = aes128gcm128-ecp256
			}
```

```
                net6-net6 {
                     local_ts = 2001:db8:0:1::/64
                     remote_ts = 2001:db8:0:2::/64
                     esp_proposals = aes128gcm128-ecp256
                }
           }
           version =2
           mobike = no
           proposals = aes128gcm128-prfsha256-ecp256
     }
}
secrets {
     ike-1 {
          id-1 = west.example.gov
          secret = XXXXXXXXXXXXXX
     }
     ike-2 {
          id-2 = east.example.gov
          secret = XXXXXXXXXXXXXX
     }
}
```

C.3 Section 9.1 Case Study Alternative Using libreswan on Linux

The following lists the contents of the same configuration as provided in Appendix C.1, but using libreswan on Linux:

```
# /etc/ipsec.d/west-east.conf
# left and right are arbitrary choices and auto-detected.
# The identical configuration can be used on both gateways
conn west-east
     left=198.51.100.1
     leftid=@west.example.gov
     right=203.0.113.1
     rightid=@east.example.gov
     ikev2=insist
     authby=secret
     auto=add
conn westnet-eastnet-ipv4
     also=west-east
     leftsubnet=192.0.0.0/24
     rightsubnet=192.0.2.0/24
     auto=start
conn westnet-eastnet-ipv6
     also-west-east
     leftsubnet=2001:db8:0:1::/64
     rightsubnet=2001:db8:0:2::/64
     auto=start
# /etc/ipsec.d/west-east.secrets
@west.example.gov @east.example.gov : PSK “XXXXXXXXXXX”
```

C.4 Section 9.1 Case Study Alternative Using iked on OpenBSD

The following lists the contents of the same configuration as was provided for Appendix C.1 but using OpenIKED on OpenBSD. Note that this IKE daemon does not support AES-GCM for IKE, only for ESP. The order of the keywords matter.

```
# /etc/iked.conf
ikev2 westnet-eastnet esp \
from 192.0.0.0/24 to 192.0.0.0/24 \
from 2001:db8:0:1::/64 to 2001:db8:0:2::/64 \
local 198.51.100.1 peer 203.0.113.1 \
ikesa enc aes-256 auth hmac-sha2-256 group ecp256 group modp2048 \
childsa enc aes-128-gcm \
childsa enc aes-128 auth hmac-sha2_512
srcid west.example.gov dstid east.example.gov \
psk XXXXXXXX \
tag west-east
```

Appendix D—Glossary

Selected terms used in the publication are defined below.

Asymmetric Cryptography	Cryptography that uses two separate keys to exchange data, one to encrypt or digitally sign the data and one for decrypting the data or verifying the digital signature. Also known as *public key cryptography*.
Authentication Header (AH)	A deprecated IPsec security protocol that provides integrity protection (but not confidentiality) for packet headers and data.
Encapsulating Security Payload (ESP)	The core IPsec security protocol; can provide encryption and/or integrity protection for packet headers and data.
Extensible Authentication Protocol (EAP)	A framework for adding arbitrary authentication methods in a standardized way to any protocol.
Internet Key Exchange (IKE)	A protocol used to negotiate, create, and manage its own (IKE) and IPsec security associations.
IP Payload Compression Protocol (IPComp)	A protocol used to perform lossless compression for packet payloads.
Keyed Hash Algorithm	An algorithm that creates a message authentication code based on both a message and a secret key shared by two endpoints. Also known as a *hash message authentication code algorithm*.
Mobile Internet Key Exchange (MOBIKE)	A form of IKE supporting the use of devices with multiple network interfaces that switch from one network to another while IPsec is in use.
Network Layer Security	Protecting network communications at the layer of the IP model that is responsible for routing packets across networks.
Perfect Forward Secrecy (PFS)	An option that causes a new secret key to be created and shared through a new Diffie-Hellman key exchange for each IPsec SA. This provides protection against the use of compromised old keys that could be used to attack the newer derived keys still in use for integrity and confidentiality protection.
Preshared Key (PSK)	A single secret key used by IPsec endpoints to authenticate endpoints to each other.
Security Association (SA)	A set of values that define the features and protections applied to a connection.

Security Association Database (SAD)	A list or table of all IPsec SAs, including those that are still being negotiated.
Security Parameters Index (SPI)	An arbitrarily chosen value that acts as a unique identifier for an IPsec connection.
Security Policy Database (SPD)	A prioritized list of all IPsec policies.
Symmetric Cryptography	A cryptographic algorithm that uses the same secret key for its operation and, if applicable, for reversing the effects of the operation (e.g., an AES key for encryption and decryption).
Traffic Flow Confidentiality (TFC) Padding	Dummy data added to real data in order to obfuscate the length and frequency of information sent over IPsec.
Transport Mode	An IPsec mode that does not create an additional IP header for each protected packet.
Tunnel Mode	An IPsec mode that creates an additional outer IP header for each protected packet.
Virtual Private Network (VPN)	A virtual network built on top of existing physical networks that can provide a secure communications mechanism for data and IP information transmitted between networks or between different nodes on the same network.

Appendix E—Acronyms and Abbreviations

Acronyms and abbreviations used in this publication are defined below.

3DES	Triple Data Encryption Standard
3GPP	3rd Generation Partnership Project
5G	5th Generation
6LowPAN	Low-Power Wireless Personal Area Network
AAA	Authentication, Authorization, and Accounting
AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
AES-CBC	Advanced Encryption Standard-Cipher Block Chaining
AES-CCM	Advanced Encryption Standard-Counter with CBC-MAC
AES-CMAC	Advanced Encryption Standard-Cipher-Based Message Authentication Code
AES-GCM	Advanced Encryption Standard-Galois Counter Mode
AES-GMAC	Advanced Encryption Standard-Galois Message Authentication Code
AES-SHA-2	Advanced Encryption Standard-Secure Hash Algorithm-2
AES-XCBC	Advanced Encryption Standard-eXtended Cipher Block Chaining
AH	Authentication Header
ALG	Application Layer Gateway
API	Application Programming Interface
ARP	Address Resolution Protocol
BGP	Border Gateway Protocol
BIOS	Basic Input/Output System
BMP	BGP Monitoring Protocol
CA	Certificate Authority
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CCMP	Counter Mode with Cipher Block Chaining Message Authentication Code Protocol
CGN	Carrier Grade NAT
CHAP	Challenge Handshake Authentication Protocol
CIDR	Classless Inter-Domain Routing
CMVP	Cryptographic Module Validation Program
CoAP	Constrained Application Protocol
COTS	Commercial-Off-the-Shelf
CP	Configuration Payload
CPU	Central Processing Unit
CRL	Certificate Revocation List
CSE	Communications Security Establishment
CVE	Common Vulnerabilities and Exposures
DDoS	Distributed Denial of Service
DES	Data Encryption Standard
DH	Diffie-Hellman

DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DNS-SD	Domain Name System Service Discovery
DNSSEC	Domain Name System Security Extensions
DPD	Dead Peer Detection
DSA	Digital Signature Algorithm
DSL	Digital Subscriber Line
DTLS	Datagram Transport Layer Security
EAP	Extensible Authentication Protocol
EAP-MSCHAPv2	Extensible Authentication Protocol-Microsoft Challenge Handshake Authentication Protocol version 2
EAP-SIM	Extensible Authentication Protocol-Subscriber Identity Module
EAP-TLS	Extensible Authentication Protocol-Transport Layer Security
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
ECP	Elliptic Curve Groups Modulo a Prime
EdDSA	Edwards Curve Digital Signature Algorithm
EKU	Extended Key Usage
ESN	Extended Sequence Number
ESP	Encapsulating Security Payload
ESPinUDP	ESP encapsulated in UDP
ESP-NULL	Encapsulating Security Payload without encryption
EVPN	Ethernet Virtual Private Network
FAQ	Frequently Asked Questions
FIDO	Fast Identity Online
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Modernization Act
FOIA	Freedom of Information Act
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
GDOI	Group Domain of Interpretation
GENEVE	Generic Network Virtualization Encapsulation
GMAC	Galois Message Authentication Code
GRE	Generic Routing Encapsulation
GSA	Group Security Association
GSO	Generic Segmentation Offload
GSSAPI	Generic Security Services Application Program Interface
GUID	Globally Unique Identifier
HKDF	HMAC Key Derivation Function
HMAC	Keyed-Hash Message Authentication Code
HMAC-MD5	Keyed-Hash Message Authentication Code-Message Digest
HMAC-SHA-1	Keyed-Hash Message Authentication Code-Secure Hash Algorithm
HMAC-SHA-2	Keyed-Hash Message Authentication Code-Secure Hash Algorithm
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure

ICMP	Internet Control Message Protocol
ICV	Integrity Check Value
IEEE	Institute of Electrical and Electronics Engineers
IGMP	Internet Group Management Protocol
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IMAP	Internet Message Access Protocol
Intel VT-d	Intel Virtualization Technology for Directed I/O
IoT	Internet of Things
IP	Internet Protocol
IPComp	IP Payload Compression Protocol
IPsec	Internet Protocol Security
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISA	Interconnection Security Agreement
ISAKMP	Internet Security Association and Key Management Protocol
ISP	Internet Service Provider
IT	Information Technology
ITL	Information Technology Laboratory
IV	Initialization Vector
KDF	Key Derivation Function
KE	Key Exchange
L2TP	Layer 2 Tunneling Protocol
L2VPN	Layer 2 VPN
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LTE	Long-Term Evolution
LZS	Lempel-Ziv-Stac
MAC	Message Authentication Code
MACsec	Media Access Control Security
MD	Message Digest
mDNS	Multicast Domain Name System
MEC	Multicast Group
MKA	MACsec Key Agreement
MOBIKE	Mobile Internet Key Exchange
MODP	Modular Exponential
MOU/A	Memorandum of Understanding or Agreement
MPLS	Multi-Protocol Label Switching
MPPE	Microsoft Point-to-Point Encryption
MS-CHAP	Microsoft Challenge-Handshake Authentication Protocol
MS-CHAPv1	Microsoft Challenge-Handshake Authentication Protocol version 1
MS-CHAPv2	Microsoft Challenge-Handshake Authentication Protocol version 2
MSEC	Multicast Security
MSS	Maximum Segment Size
MTU	Maximum Transmission Unit

NAPT	Network Address Port Translation
NAT	Network Address Translation
ND	Neighbor Discovery
NETCONF	Network Configuration Protocol
NIC	Network Interface Card
NIST	National Institute of Standards and Technology
NSA	National Security Agency
NUMA	Non-Uniform Memory Access
NVD	National Vulnerability Database
NVO3	Network Virtualization Overlay
OAuth	Open Authorization
OCSP	Online Certificate Status Protocol
OMB	Office of Management and Budget
OSPF	Open Shortest Path First
OTP	One-Time Password
P.L.	Public Law
PAKE	Password Authenticated Key Exchange
PAM	Pluggable Authentication Module
PAP	Password Authentication Protocol
PFS	Perfect Forward Secrecy
PKCS	Public Key Cryptography Standards
PKI	Public Key Infrastructure
POP	Post Office Protocol
PPK	Postquantum Preshared Key
PPP	Point-to-Point Protocol
PPTP	Point-to-Point Tunneling Protocol
PRF	Pseudo Random Function
PSK	Preshared Key
PSS	Probabilistic Signature Scheme
QoS	Quality of Service
RADIUS	Remote Authentication Dial In User Service
RAM	Random Access Memory
RFC	Request for Comment
RMON	Remote Monitoring
S/MIME	Secure/Multipurpose Internet Mail Extensions
SA	Security Association
SAD	Security Association Database
SAN	subjectAltName
SDN	Software-Defined Networking
SDWAN	Software-Defined Wide Area Network
SHA	Secure Hash Algorithm
SIP	Session Initiation Protocol
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SP	Special Publication

SPD	Security Policy Database
SPI	Security Parameters Index
SPKI	SubjectPublicKeyInfo
SSH	Secure Shell
SSL	Secure Sockets Layer
SSTP	Secure Socket Tunneling Protocol
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TCP-TLS	Transmission Control Protocol-Transport Layer Security
TFC	Traffic Flow Confidentiality
TKIP	Temporal Key Integrity Protocol
TLS	Transport Layer Security
TSi	Traffic Selector for Initiator
TSO	TCP Segmentation Offload
TSr	Traffic Selector for Responder
TTL	Time to Live
UDP	User Datagram Protocol
URI	Uniform Resource Indicator
USB	Universal Serial Bus
US-CERT	United States Computer Emergency Readiness Team
VLAN	Virtual Local Area Network
VM	Virtual Machine
VoIP	Voice over IP
VPN	Virtual Private Network
VXLAN	Virtual eXtensible Local Area Network
WEP	Wired Equivalent Privacy
WiFi	Wireless Fidelity
WPA	Wi-Fi Protected Access
WPA2	Wi-Fi Protected Access version 2
WPA3	Wi-Fi Protected Access version 3
XCBC	eXtended Cipher Block Chaining

Appendix F—References

This appendix contains the references for the document.

[1] Eggert L, Fairhurst G, Shepherd G (2017) *UDP Usage Guidelines*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8085 and Best Current Practice 145. https://doi.org/10.17487/RFC8085

[2] van Elburg J, Drage K, Ohsugi M, Schubert S, Arai K (2014) *The Session Initiation Protocol (SIP) P-Private-Network-Indication Private Header (P-Header)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 7316. https://doi.org/10.17487/RFC7316

[3] Saito M, Wing D, Toyama M (2011) *Media Description for the Internet Key Exchange Protocol (IKE) in the Session Description Protocol (SDP)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6193. https://doi.org/10.17487/RFC6193

[4] Mahalingam M, Dutt D, Duda K, Agarwal P, Kreeger L, Sridhar T, Bursell M, Wright C (2014) *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 7348. https://doi.org/10.17487/RFC7348

[5] Filsfils C, Previdi S, Ginsberg L, Decraene B, Litkowski S, Shakir R (2018) *Segment Routing Architecture*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8402. https://doi.org/10.17487/RFC8402

[6] Yong L, Dunbar L, Toy M, Isaac A, Manral V (2017) *Use Cases for Data Center Network Virtualization Overlay Networks*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8151. https://doi.org/10.17487/RFC8151

[7] Gross J, Ganga I, Sridhar T (2019) *Geneve: Generic Network Virtualization Encapsulation*. (Internet Engineering Task Force (IETF) Network Working Group). Internet-Draft draft-ietf-nvo3-geneve-13. https://datatracker.ietf.org/doc/draft-ietf-nvo3-geneve/

[8] Baker F, Meyer D (2011) *Internet Protocols for the Smart Grid*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6272. https://doi.org/10.17487/RFC6272

[9] Pauly T, Touati S, Mantha R (2017) *TCP Encapsulation of IKE and IPsec Packets*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8229. https://doi.org/10.17487/RFC8229

[10] Bhatia M, Manral V (2011) *Summary of Cryptographic Authentication Algorithm Implementation Requirements for Routing Protocols*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6094. https://doi.org/10.17487/RFC6094

[11] Scudder J, Fernando R, Stuart S (2016) *BGP Monitoring Protocol (BMP)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 7854. https://doi.org/10.17487/RFC7854

[12] Gupta M, Melam N (2006) *Authentication/Confidentiality for OSPFv3*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4552. https://doi.org/10.17487/RFC4552

[13] National Institute of Standards and Technology (2019) *Security Requirements for Cryptographic Modules*. Federal Information Processing Standard (FIPS) 140-3. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf

[14] Ramsdell B, Turner S (2010) *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 5751. https://doi.org/10.17487/RFC5751

[15] Ylonen T, Lonvick C (2006) *The Secure Shell (SSH) Transport Layer Protocol*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4253. https://doi.org/10.17487/RFC4253

[16] Rescorla E (2018) *The Transport Layer Security (TLS) Protocol Version 1.3*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8446. https://doi.org/10.17487/RFC8446

[17] Rescorla E, Tschofenig H, Modadugu N (2019) *The Datagram Transport Layer Security (DTLS) Protocol Version 1.3*. (Internet Engineering Task Force (IETF)). Internet-Draft draft-ietf-tls-dtls13-31. https://datatracker.ietf.org/doc/draft-ietf-tls-dtls13/

[18] Kaufman C, Hoffman P, Nir Y, Eronen P, Kivinen T (2014) *Internet Key Exchange Protocol Version 2 (IKEv2)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 7296 and Internet Standard 79. https://doi.org/10.17487/RFC7296

[19] Kent S (2005) *IP Encapsulating Security Payload (ESP)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 4303. https://doi.org/10.17487/RFC4303

[20] Nir Y, Kivinen T, Wouters P, Migault D (2017) *Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8247. https://doi.org/10.17487/RFC8247

[21] Wouters P, Migault D, Mattsson J, Nir Y, Kivinen T (2017) *Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8221. https://doi.org/10.17487/RFC8221

[22] Frankel S, Krishnan S (2011) *IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6071. https://doi.org/10.17487/RFC6071

[23] Rosen E, Viswanathan A, Callon R (2001) *Multiprotocol Label Switching Architecture*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 3031. https://doi.org/10.17487/RFC3031

[24] National Institute of Standards and Technology (2008) *The Keyed-Hash Message Authentication Code (HMAC)*. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 198-1. https://doi.org/10.6028/NIST.FIPS.198-1

[25] National Institute of Standards and Technology (2015) *Secure Hash Standard (SHS)*. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 180-4. https://doi.org/10.6028/NIST.FIPS.180-4

[26] National Institute of Standards and Technology (2001) *Advanced Encryption Standard*. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 197. https://doi.org/10.6028/NIST.FIPS.197

[27] Dworkin M (2005) *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-38B, Includes updates as of October 6, 2016. https://doi.org/10.6028/NIST.SP.800-38B

[28] Dworkin M (2007) *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-38D. https://doi.org/10.6028/NIST.SP.800-38D

[29] National Institute of Standards and Technology (2013) *Digital Signature Standard (DSS)*. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 186-4. https://doi.org/10.6028/NIST.FIPS.186-4

[30] Enns R, Bjorklund M, Schoenwaelder J, Bierman A (2011) *Network Configuration Protocol (NETCONF)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6241. https://doi.org/10.17487/RFC6241

[31] Carrel D, Weis B (2019) *IPsec Key Exchange using a Controller.* (Internet Engineering Task Force (IETF) Network Working Group). Internet-Draft draft-carrel-ipsecme-controller-ike-01. https://datatracker.ietf.org/doc/draft-carrel-ipsecme-controller-ike

[32] Eronen P (2006) *IKEv2 Mobility and Multihoming Protocol (MOBIKE)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4555. https://doi.org/10.17487/RFC4555

[33] Aboba B, Calhoun P (2003) *RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 3579. https://doi.org/10.17487/RFC3579

[34] Eronen P, Hiller T, Zorn G (2005) *Diameter Extensible Authentication Protocol (EAP) Application*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4072. https://doi.org/10.17487/RFC4072

[35] Rekhter Y, Moskowitz B, Karrenberg D, de Groot G J, Lear E (1996) *Address Allocation for Private Internets*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 1918 and Best Current Practice (BCP) 5. https://doi.org/10.17487/RFC1918

[36] Smyslov V (2014) *Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 7383. https://doi.org/10.17487/RFC7383

[37] Fluhrer S, McGrew D, Kampanakis P, Smyslov V (2019) *Postquantum Preshared Keys for IKEv2*. (Internet Engineering Task Force (IETF)). Internet-Draft draft-ietf-ipsecme-qr-ikev2-08. https://datatracker.ietf.org/doc/draft-ietf-ipsecme-qr-ikev2/

[38] Devarapalli V, Weniger K (2009) *Redirect Mechanism for the Internet Key Exchange Protocol Version 2 (IKEv2)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 5685. https://doi.org/10.17487/RFC5685

[39] Kent S, Atkinson R (1998) *IP Encapsulating Security Payload (ESP)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 2406. https://doi.org/10.17487/RFC2406

[40] Kent S, Seo K (2005) *Security Architecture for the Internet Protocol*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 4301. https://doi.org/10.17487/RFC4301

[41] Krawczyk H, Bellare M, Canetti R (1997) *HMAC: Keyed-Hashing for Message Authentication*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 2104. https://doi.org/10.17487/RFC2104

[42] Frankel S, Herbert H (2003) *The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 3566. https://doi.org/10.17487/RFC3566

[43] Black D, McGrew D (2008) *Using Authenticated Encryption Algorithms with the Encrypted Payload of the Internet Key Exchange version 2 (IKEv2) Protocol*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 5282. https://doi.org/10.17487/RFC5282

[44] Viega J, McGrew D (2005) *The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4106. https://doi.org/10.17487/RFC4106

[45] Frankel S, Glenn R, Kelly S (2003) *The AES-CBC Cipher Algorithm and Its Use with IPsec*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 3602. https://doi.org/10.17487/RFC3602

[46] Kelly S, Frankel S (2007) *Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4868. https://doi.org/10.17487/RFC4868

[47] Barker E, Roginsky A (2019) *Transitioning the Use of Cryptographic Algorithms and Key Lengths*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-131A, Rev. 2. https://doi.org/10.6028/NIST.SP.800-131Ar2

[48] Kent S (2005) *Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 4304. https://doi.org/10.17487/RFC4304

[49] Pauly T, Touati S, Mantha R (2017) *TCP Encapsulation of IKE and IPsec Packets*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8229. https://doi.org/10.17487/RFC8229

[50] McDonald D, Metz C, Phan B (1998) *PF_KEY Key Management API, Version 2*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 2367. https://doi.org/10.17487/RFC2367

[51] Salim J, Khosravi H, Kleen A, Kuznetsov A (2003) *Linux Netlink as an IP Services Protocol*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 3549. https://doi.org/10.17487/RFC3549

[52] Shafer P (2011) *An Architecture for Network Management Using NETCONF and YANG*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6244. https://doi.org/10.17487/RFC6244

[53] Srisuresh P, Holdrege M (1999) *IP Network Address Translator (NAT) Terminology and Considerations*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 2663. https://doi.org/10.17487/RFC2663

[54] Moriarty K, Morton A (2018) *Effects of Pervasive Encryption on Operators*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8404. https://doi.org/10.17487/RFC8404

[55] Korver B (2007) *The Internet IP Security PKI Profile of IKEv1/ISAKMP, IKEv2, and PKIX*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4945. https://doi.org/10.17487/RFC4945

[56] Kuhn DR, Hu VC, Polk WT, Chang S-jH (2001) *Introduction to Public Key Technology and the Federal PKI Infrastructure*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-32. https://doi.org/10.6028/NIST.SP.800-32

[57] Cooper D, Santesson S, Farrell S, Boeyen S, Housley R, Polk W (2008) *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 5280. https://doi.org/10.17487/RFC5280

[58] Santesson S, Myers M, Ankney R, Malpani A, Galperin S, Adams C (2013) *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 6960. https://doi.org/10.17487/RFC6960

[59] Esale S, Torvi R, Jalil L, Chunduri U, Raza K (2017) *Application-Aware Targeted LDP*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8223. https://doi.org/10.17487/RFC8223

[60] Barker E, Chen L, Roginsky A, Vassilev A, Davis R, Simon S (2019) *Recommendation for Pair-Wise Key Establishment Using Integer Factorization Cryptography*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-56B, Rev. 2. https://doi.org/10.6028/NIST.SP.800-56Br2

[61] Fu D, Solinas J (2010) *Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 5903. https://doi.org/10.17487/RFC5903

[62] Barker E, Chen L, Roginsky A, Vassilev A, Davis R (2018) *Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-56A, Rev. 3. https://doi.org/10.6028/NIST.SP.800-56Ar3

[63] Sheffer Y, Tschofenig H (2010) *Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 5723. https://doi.org/10.17487/RFC5723

[64] Kivinen T, Kojo M (2003) *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 3526. https://doi.org/10.17487/RFC3526

[65] Lepinski M, Kent S (2008) *Additional Diffie-Hellman Groups for Use with IETF Standards*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 5114. https://doi.org/10.17487/RFC5114

[66] Kuhn D R, Walsh T, Fries S (2005) *Security Considerations for Voice Over IP Systems*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-58. https://doi.org/10.6028/NIST.SP.800-58

[67] Souppaya M, Scarfone K (2012) *Guidelines for Securing Wireless Local Area Networks (WLANs)*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-153. https://doi.org/10.6028/NIST.SP.800-153

[68] Rescorla E, Modadugu N (2012) *Datagram Transport Layer Security Version 1.2*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6347. https://doi.org/10.17487/RFC6347

[69] Frankel S, Hoffman P, Orebaugh A, Park R (2008) *Guide to SSL VPNs*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-113. https://doi.org/10.6028/NIST.SP.800-113

[70] Mavrogiannopoulos N (2018) *The OpenConnect VPN Protocol Version 1.1*. (Internet Engineering Task Force (IETF)). Internet-Draft draft-mavrogiannopoulos-openconnect-02. https://datatracker.ietf.org/doc/draft-mavrogiannopoulos-openconnect/

[71] Krawczyk H, Eronen P (2010) *HMAC-based Extract-and-Expand Key Derivation Function (HKDF)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 5869. https://doi.org/10.17487/RFC5869

[72] Nir Y, Josefsson S (2016) *Curve25519 and Curve448 for the Internet Key Exchange Protocol Version 2 (IKEv2) Key Agreement.* (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8031. https://doi.org/10.17487/RFC8031

[73] Nir Y, Langley A (2015) *ChaCha20 and Poly1305 for IETF Protocols*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 7539. https://doi.org/10.17487/RFC7539

[74] Nir Y (2018) *Using the Edwards-Curve Digital Signature Algorithm (EdDSA) in the Internet Key Exchange Protocol Version 2 (IKEv2)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 8420. https://doi.org/10.17487/RFC8420

[75] Simpson W (1994) *The Point-to-Point Protocol (PPP)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 1661 and Standard 51. https://doi.org/10.17487/RFC1661

[76] Hamzeh K, Pall G, Verthein W, Taarud J, Little W, Zorn G (1999) *Point-to-Point Tunneling Protocol (PPTP)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 2637. https://doi.org/10.17487/RFC2637

[77] Pall G, Zorn G (2001) *Microsoft Point-to-Point Encryption (MPPE) Protocol*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 3078. https://doi.org/10.17487/RFC3078

[78] Lloyd B, Simpson W (1992) *PPP Authentication Protocols*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 1334. https://doi.org/10.17487/RFC1334

[79] Simpson W (1996) *PPP Challenge Handshake Authentication Protocol (CHAP)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 1994. https://doi.org/10.17487/RFC1994

[80] Townsley W, Valencia A, Rubens A, Pall G, Zorn G, Palter B (1999) *Layer Two Tunneling Protocol "L2TP"*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 2661. https://doi.org/10.17487/RFC2661

[81] Rigney C, Willens S, Rubens A, Simpson W (2000) *Remote Authentication Dial In User Service (RADIUS)*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 2865. https://doi.org/10.17487/RFC2865

[82] Weis B, Gross G, Ignjatic D (2008) *Multicast Extensions to the Security Architecture for the Internet Protocol*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 5374. https://doi.org/10.17487/RFC5374

[83] Hardjono T, Weis B (2004) *The Multicast Group Security Architecture*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 3740. https://doi.org/10.17487/RFC3740

[84] Weis B, Smyslov V (2019) *Group Key Management using IKEv2*. (Internet Engineering Task Force (IETF) Network Working Group). Internet-Draft draft-yeung-g-ikev2-15. https://datatracker.ietf.org/doc/draft-yeung-g-ikev2/

[85] Baugher M, Canetti R, Dondeti L, Lindholm F (2005) *Multicast Security (MSEC) Group Key Management Architecture*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 4046. https://doi.org/10.17487/RFC4046

[86] Weis B, Rowles S, Hardjono T (2011) *The Group Domain of Interpretation*. (Internet Engineering Task Force (IETF)). Request for Comments (RFC) 6407. https://doi.org/10.17487/RFC6407

[87] Prasad S, Wouters P (2018) *Labeled IPsec Traffic Selector Support for IKEv2*. (Internet Engineering Task Force (IETF)). Internet-Draft draft-ipsecme-labeled-ipsec-00. https://datatracker.ietf.org/doc/draft-ietf-ipsecme-labeled-ipsec/

[88] Migault D, Guggemos T, Nir Y (2019) *Implicit IV for Counter-Based Ciphers in Encapsulating Security Payload (ESP).* (Internet Engineering Task Force (IETF)). Internet-Draft draft-ietf-ipsecme-implicit-iv-07. https://datatracker.ietf.org/doc/draft-ietf-ipsecme-implicit-iv/

[89] Smyslov V (2018) *Intermediate Exchange in the IKEv2 Protocol.* (Internet Engineering Task Force (IETF) Network Working Group). Internet-Draft draft-smyslov-ipsecme-ikev2-aux-02. https://datatracker.ietf.org/doc/draft-smyslov-ipsecme-ikev2-aux/

[90] Boucadair M (2018) *IKEv2 Notification Status Types for IPv4/IPv6 Coexistence.* (Internet Engineering Task Force (IETF) Network Working Group). Internet-Draft draft-ietf-ipsecme-ipv6-ipv4-codes-02. https://datatracker.ietf.org/doc/draft-ietf-ipsecme-ipv6-ipv4-codes/

[91] Tjhai C, Tomlinson M, Bartlett G, Flurher S, Van Geest D, Garcia-Morchon O, Smyslov V (2019) *Framework to Integrate Post-Quantum Key Exchanges Into Internet Key Exchange Protocol Version 2 (IKEv2)*. (Internet Engineering Task Force (IETF)). Internet-Draft draft-tjhai-ipsecme-hybrid-qske-ikev2-03. https://datatracker.ietf.org/doc/draft-tjhai-ipsecme-hybrid-qske-ikev2/

[92] Grance T, Hash J, Peck S, Smith J, Korow-Diks K (2002) *Security Guide for Interconnecting Information Technology Systems.* (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-47. https://doi.org/10.6028/NIST.SP.800-47

[93] Souppaya M, Scarfone K (2016) *Guide to Enterprise Telework, Remote Access, and Bring Your Own Device (BYOD) Security*. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-46, Rev. 2. https://doi.org/10.6028/NIST.SP.800-46r2

[94] Harkins D, Carrel D (1998) *The Internet Key Exchange (IKE)*. (Internet Engineering Task Force (IETF) Network Working Group). Request for Comments (RFC) 2409. https://doi.org/10.17487/RFC2409