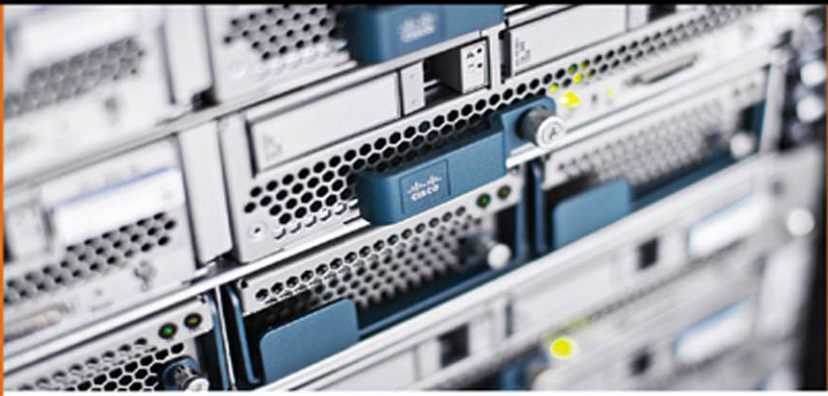


SECURITY

IKEv2 IPsec Virtual Private Networks

Understanding and Deploying IKEv2, IPsec VPNs, and FlexVPN in Cisco IOS

Graham Bartlett, CCIE No. 26709
Amjad Inamdar, CISSP No. 460898

ciscopress.com

IKEv2 IPsec Virtual Private Networks

Understanding and Deploying IKEv2, IPsec VPNs, and FlexVPN in Cisco IOS

Graham Bartlett, CCIE No. 26709

Amjad Inamdar, CISSP No. 460898

Cisco Press

800 East 96th Street

Indianapolis, Indiana, 46240 USA

IKEv2 IPsec Virtual Private Networks

Understanding and Deploying IKEv2, IPsec VPNs, and FlexVPN in Cisco IOS

Graham Bartlett, CCIE No. 26709
Amjad Inamdar, CISSP No. 460898

Published by:
Cisco Press
800 East 96th Street
Indianapolis, IN 46240 USA

Printed in the United States of America 1 2 3 4 5 6 7 8 9 0

First Printing August 2016

Library of Congress Cataloging-in-Publication Number: 2016947704

ISBN-13: 978-1-58714-460-8

ISBN-10: 1-58714-460-3

Warning and Disclaimer

This book is designed to provide information about IKEv2 and IPsec VPNs on Cisco IOS. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an "as-is" basis. The authors, Cisco Press, and Cisco Systems, Inc. shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Cisco Systems, Inc.

Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through email at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc. cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **www.cisco.com/go/offices.**

Editor in Chief: Mark Taub

Product Line Manager: Brett Bartow

Acquisitions Editor: Brett Bartow

Business Operation Manager, Cisco Press: Ronald Fligge

Managing Editor: Sandra Schroeder

Development Editor: Ellie Bru

Project Editor: Mandie Frank

Copy Editor: Lori Martinsek

Technical Editors: Alex Honore, Olivier Pelerin

Team Coordinator: Tammi Barnett

Cover Designer: Chuti Prasertsith

Composition: codeMantra

Indexer: Lisa Stumpf

About the Authors

Graham Bartlett, CCIE No. 26709, has designed a number of large scale Virtual Private Networks within the UK and worked with customers throughout the world using IKEv2 and Next Generation Encryption. Graham's interests include Security and Virtual Private Networks. Within this space he has discovered zero-day vulnerabilities, including the higest severity security advisory in the March 2015 Cisco IOS software and IOS XE software security advisory bundled publication. He has contributed to numerous IETF RFCs, and has intellectual property published as prior art. He is a CiscoLive speaker and has developed Cisco Security exam content (CCIE/CCNP). He is a CCP (Senior) IA Architect, CCP (Practitioner) Security & Information Risk Advisor, CCNP, CISSP, Cisco Security Ninja and holds a BSc(Hons) in Computer Systems and Networks.

Amjad Inamdar CISSP 460898, is a Senior Technical Leader with Cisco IOS Security Engineering, India. He has primarily worked on design, development and deployment of Cisco IOS secure connectivity solutions including the industry leading FlexVPN, DMVPN, GETVPN and EzVPN solutions and is currently working on the Cisco next generation SD-WAN solution. He has contributed to IETF drafts, holds a Cisco patent and has prior art publications. He holds many industry certifications including CISSP, CCSK, CCNP Security, CCDP, CCNP R/S, CCNA (SP, Data Center, Wireless, Voice), Cisco Security Ninja and has presented security at conferences, internal forums and to Cisco customers and partners. He holds a degree (B.E) in Electronics and Communication Engineering.

Note from the Authors

Alex Honore was originally an author of this book, but due to commitments opted to become a Technical Reviewer. Alex has been a fundamental member of the team, a number of chapters were originally written by Alex in addition to some of the diagrams.

About the Technical Reviewers

Alex Honore, (CCIE Security No. 19553) has been with Cisco since 2005 and currently works as a Technical Leader in Cisco's Security Business Group, specializing in leading-edge network and threat analytics. He was a senior engineer in Cisco Technical Services for 9 years, focusing on advanced troubleshooting and escalation support for VPN, Security, and Content Networking solutions in the Technical Assistance Center (TAC), as well as consulting for Cisco Advanced Services and speaking regularly at Cisco Live on the topic of IPsec VPNs. Alex holds a M.Sc. degree of Electrical Engineering and Telecommunications from the Faculty of Engineering in Mons, Belgium.

Olivier Pelerin (CCIE Security No. 20306) has more than 16 years of experience in networking and security. He joined Cisco TAC (Cisco Technical Assistance Center) as a customer support engineer back in 2005. He is still working for TAC as a Technical Leader, focusing on escalations around VPN solutions. He has been involved in FlexVPN since the start and is co-leading the development of the packet tracer on ISR-NG/ASR1000. Olivier is a distinguished speaker at CiscoLive. He holds a degree in Geography from Univesité Catholique de Louvain (Belgium).

Dedications

Graham Bartlett:

I dedicate this book to my loving family: Lorna, Edward, Rose, and my parents.

Amjad Inamdar:

I dedicate this book to my parents, Sayedrasul and Ayesha Inamdar, who have been my role models and inspiration, and to my entire family.

Acknowledgments

Graham Bartlett:

Alex Honore deserves a very special mention. Alex originally was an author, but due to work and family commitments became a reviewer. Alex has been a diligent reviewer due to his ability to break complex topics into many simple layers. I want to thank Alex for his patience and the help he has provided. We are honored to have him as a reviewer and he is a great asset to Cisco. I know if Alex ever writes a book himself, it will be, for me, the perfect book.

Olivier Pelerin has been a fantastic reviewer and brought a lot of issues and resolutions to our attention that otherwise we would have missed. Olivier is 100% focused on customers and many of the tips, tricks, and guidance contained within need to be credited to him. Many times I've had late-night chats regarding IKEv2 and our implementation. Olivier should also be credited for improving Cisco's IKEv2 usability, stability, and serviceability, which ultimately has a positive effect on you—the customer.

Thank you both—I treat you as good friends and could not have achieved this without your knowledge and skills.

Frederic Detienne: Fred—thank you for leading the creation of DMVPN, Flex, and IKEv2 on Cisco IOS. FlexVPN was my main motivation for joining Cisco, and it's now apparent that as VPN Architect you have developed a technology that is unique within the industry. It's a great pleasure and honor to work with you.

Raffaele Brancaleoni: Raff, thanks for all the help you provided with the development of this book. You've been a great mentor and a good friend.

The many great VPN engineers in Cisco TAC: Piotr and the fantastic Krakow crew, Wen, Atri, Jay, Frank, Marcin, Istvan, Nitin, Tom. Xavier and PSIRT. All of the IKEv2 developers that have helped: Sairam, Manish, Balaji, Tapesh. Crypto team: Anthony, David, Scott and Panos.

Richard Gallagher: A great engineer and friend.

All the customers that I work with, who are not just customers of Cisco, but friends. You know who you are.

Thanks for the support of my management team, Francois, Khalid, David and Ashley.

Thanks to all my good friends and colleagues within Cisco and within the industry; you know who you are.

Also thanks for the support and patience of the Cisco Press team, namely Ellie and Brett.

Amjad Inamdar:

First, I would like to thank God for this opportunity to write a book which I have always wanted to do. Special thanks to my wife, Huzefiya, for being a constant companion in writing this book and her encouragement and support; my children, Sobia, Aleena, Arfa, and Ammar, for their patience and understanding while I spent many of the weekends and holidays writing this book. Thanks to my brother, sisters and family members for their support.

Graham Bartlett—Thank you for launching me onto this journey by offering me the chance to be the co-author. I look forward to continuing this journey with you. Thanks for your patience and all your help, you are a great friend.

Frederic Detienne—Thank you for being my mentor. I have truly enjoyed working with you in designing and implementing the Cisco IOS IKEv2 and FlexVPN solution.

Alex Honore—Thank you for your excellent review. You have been an inspiration and a great friend and guide. I will always cherish our work together on IKEv2 and FlexVPN.

Olivier Pelerin—Thank you for your detailed review comments, most of which were customer focused. I value our work together on FlexVPN scalability.

Thanks to Raffaele Brancaleoni for his review comments and to Yeleshwarapu Sairam, Tapesh Maheshwari, Kalyani Garigipati from the IKEv2 and FlexVPN development team for their input.

Thanks to my management, colleagues, and friends at Cisco for their support.

Contents at a Glance

Contents

Foreword

Dear reader,

My name is Frederic Detienne and I had the chance to participate during the early days and eventually to lead cryptographic product development in Cisco, acting as Architect of DMVPN and FlexVPN. I started as a TAC Engineer and have evolved into network designer and consultant, inside Cisco and toward our customers. I work across functions with our Engineering, Advanced Services, Support, and Marketing departments.

It seems like yesterday or many eons ago … In the beginning were crypto maps.

A few dinosaurs (like myself) started their journey in cryptographic protocols and algorithms when Cisco released CET (Cisco Encryption Technology) on IOS 11.2 in August 2003.

My only exposure to cryptography had been strictly theoretical, as a student at the University of Liège 7 years before. I suppose I was lucky to have had such a background as around me, nobody seemed to have received any crypto education nor felt inclined toward that very obscure technology. Before that, cryptography was managed through very complex systems, mostly reserved to governments and militaries.

CET was commercial grade in the sense that it was a major simplification over the former systems. It allowed a mere mortal to configure a very regular and relatively cheap router (Cisco 2500) to encrypt data across a public Layer 3 network. The cryptographic algorithms were very good: DES, then 3-DES, Diffie-Hellman key exchange. At 160 Kbps, the throughput was acceptable in those days.

In the aggregation services, technology goes through 4 steps: make it work, make it work reliably, make it work at speed, make it work at scale. There are other timelines of interest, but this one mattered particularly for cryptographic VPNs.

CET evolved into IKE/ISAKMP + IPsec as the drafts matured into standards under the leadership of Dan Harkins.

Nobody really knew what we were going toward. The initial code inherited from CET which we still had to support for our early adopters. It was also modular and ready to accommodate future enhancements, optimizations and hardware architectures. In a word: it was messy.

The data-plane vs control-plane separation outlined into RFC2408 was both a blessing and a curse. On one hand it brought complexity, on the other, it brought good OSI and code separation without which we may not have survived.

At the control plane level, IKE itself and its rekey complexity, the differences in behavior between IKE SA rekeys and IPsec SA rekeys triggered many race conditions. Overall, we managed to stabilize the system and we “made it to work reliably.” Step 2 was complete.

In the data plane things were less rosy. Crypto maps quickly showed their limits:

–the combinatory explosion of source/destination pairs on ever larger and complex networks

–code complexity due to packets being stolen in OSI layer 2 and re-encapsulated into a new IP header (OSI layer 3)

The security policy size explosion made mesh networks totally unmanageable. Besides, the security policy was mostly a static transcription of information we already had in the dynamic routing table, which led to customer frustration.

A few site-to-site and hub-and-spoke configurations were possible, but manageability and scalability suffered badly. The TAC was recommending the use of GRE protected by IPsec in order to run routing protocols on top of the tunnels. This quickly became the preferred way to deploy complex meshes. Scalability was relatively limited due to hardware performance, but it really made everyone's life easier and the support of those network became very pleasant with more and more satisfied customers.

Meanwhile, EasyVPN had appeared and was offering a remote access solution. The clients were either PC software or small branch routers. The big advantage was that the hub configuration was very compact—a few lines would allow hundreds of remote branches to connect. Unfortunately, the underlying implementation relied on crypto maps and suffered from quality and supportability issues. While EasyVPN was very good, it was not stable enough compared to the GRE/IPsec solution we used in mesh.

Customers had to choose between easy of configuration for large but simple hub-and-spoke networks and a more complex configuration for mesh networks.

One day, someone showed me NHRP: a protocol to establish circuits on demand. The code was very crude and incomplete but the developer (who had left Cisco by then) had provisioned for GRE tunnels, very likely in order to test his code without expensive equipment. I had this light bulb moment and hacked together a prototype to encrypt those GRE tunnels as they were created.

DMVPN was born in a TAC lab in Brussels, demonstrated to our colleagues in San Jose, California, and developed into a product.

We now had something that worked well, was satisfactorily stable despite being a fresh feature, and offered an easy configuration for complex networks. It started as DMVPN phase 1 with hub-and-spoke only and followed quickly with DMVPN phase 2 allowing dynamic branch-to-branch tunnel creation.

Scale was not there yet though. The IGPs (OSPF and EIGRP mostly) caused significant burdens and deploying more than 350 nodes networks was still a burden. It may seem small today, but the bulk of the network sizes grew as technology permitted. A mesh network of 350 nodes was fantastic back then. Just that the market quickly got used to it and demand for more appeared.

The market demanded that we scale up both the tunnel density (the number of tunnels on a single given platform) and horizontally (the ability of a cluster of DMVPN hubs to collaborate to service). DMVPN phase 2 daisy chaining was a dreaded system to design and troubleshoot. Besides scalability problems, it also suffered from reconvergence time and convoluted configuration.

The workload reduced dramatically while market shares and revenue took off; we started work on scaling DMVPN before it become too stringent.

The semantic of the NHRP redirect and NHRP resolution forwarding appeared and helped us scale almost limitlessly across hubs. You could literally have dozens of hubs working in cluster mode. Also importantly, we could finally get away from the traditional IGPs and investigate lighter protocols such as RIP, OTV and even BGP (which is feature-rich and complex at large but out of which we only needed the simplest elements).
We could now scale to about 1500 peers per hub and a virtually limitless number of hubs. Each additional hub would linearly add its capability to the cluster. This was an important step forward in network design.

The biggest challenge was now to educate our customers and sales team about the various options and design. When you work on Crypto VPNs all day, every day, it is easy to forget that very few people actually understand the ins and outs of every feature and design. Also, many customers were satisfied with what they had and had no reason to investigate for more—or even suspect that something better could exist.

A metric of complexity could be seen in our 8 hours CiscoLive session going over our multiple Crypto VPN solutions and their use case:

- Crypto maps
- Easy VPN (client mode and network extension mode)
- Enhanced Easy VPN
- GRE/IPsec
- GET VPN
- DMVPN phase 1, 2 and 3

The pros and cons panned out as below:

- Crypto maps were still as limited and terrible as before but are necessary for third-party integration as they offer minimal compatibility with devices that have minimal functionality.
- EasyVPN supported remote access (especially the software client) compact but the feature had grown organically and the UI was terrible; it was also crypto map based, and its quality was poor.
- Enhanced Easy VPN solved the crypto map problem and was a major improvement over Easy VPN, but it did not enjoy proper marketing and remained poorly adopted. The UI was the same and hence difficult.

- GRE/IPsec was slowly disappearing at the benefit of DMVPN and tunnel protection in the site-to-site scenarios.
- GET VPN has lower security and limited scalability, but it is lighter on resources when used properly, if the use case is adequate. Notably, it allows native multicast.
- DMVPN was growing in both the hub-and-spoke and partial mesh cases, but the routing protocol was a deterrent for Security Operations who preferred using EasyVPN.

This really meant 8 hours during which we barely had the time to describe how a solution worked and what use case it was best for.

Customers who were successively shopping for a remote access solution, then a site-to-site, then a dynamic mesh … had to study and learn new ways of designing and troubleshooting for each feature, over and again.

The complexity we were witnessing in TAC on our fresh recruits was impacting our customers, partners, Advanced Services, and sales teams.

At the same time, as all things so far, after a few years, market demand slowly started to outgrow DMVPN. Tunnel density still had to increase and the routing protocols were not scaling anymore.

We decided to merge EasyVPN and DMVPN features into a single feature that would offer us the advantages of both under a single feature set: one time learning, applicable always. The characteristics had to be the following:

- clear, consistent, compact, and powerful CLI: simple things ought to be simple to configure, complex things ought to be possible.
- using routing protocols should be a customer choice, not mandatory.
- NHRP usage could decrease except for spoke-spoke tunnel creation
- increased scale to 10,000 tunnels per hub at least.
- all the remote access management features had to be applicable to site-to-site and hub-and-spoke (AAA authorization in particular to apply per user QoS, ACLs, and so on.)
- reduce the reliance on PKI and make pre-shared keys more manageable. Both had to be possible, at least for hub-and-spoke
- backup and load balancing scenario
- third-party interoperability
- high serviceability/troubleshootability
- reduced learning time by using consistent protocol and data flows
- state of the art security at the cryptographic and network level

Because we could not take the risk to break IKEv1 stability nor invest in a protocol that was slated to disappear, we used IKEv2 as an inflection point to do things right. Clean implementation, clean user interface.

Today, we are capable of offering combined training, including hands-on experience, covering remote-access, hub-and-spoke, dynamic mesh, AAA management, and some troubleshooting in 4 (fours) hours. The total training time has decreased by an order of magnitude.

FlexVPN is not perfect and is not the end of the road, but in terms of applicability and total cost of ownership, taking in account training time and supportability, this is the best we have ever had.

I hope you will have as much pleasure discovering FlexVPN in this book as we had developing those features, thinking about you, our users, our customers, our sponsors.

None of this would have happened without great individuals who went beyond the basic market analysis that a typical Product Management team performs and took it on themselves to listen to our customers' real demands.

Namely, it took the courage of one Senior Manager, Pratima Sethi, to sponsor and execute on the development of FlexVPN. She also made DMVPN and EasyVPN successful; she understood deeply the need of post-deployment capabilities such as monitoring and troubleshooting and made it all possible.

The authors of this book, Amjad Inamdar and Graham Bartlett, are long-time collaborators who also deeply impacted all our VPN solutions, and I am very proud to work with them.

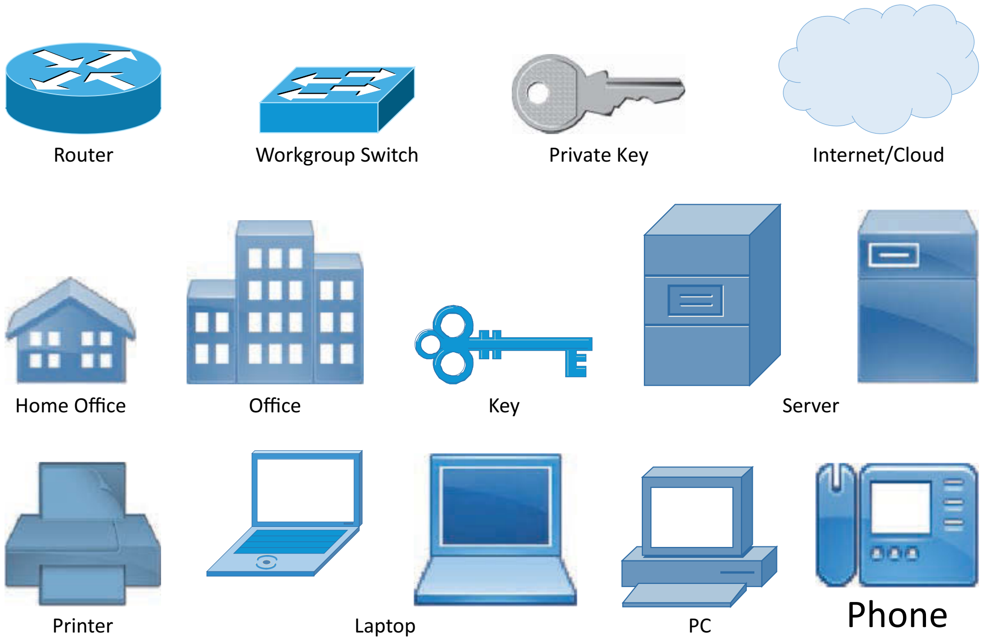
The teams prominent members included Alexandre Honore, Olivier Pelerin, Wen Zhang, Raffaele Brancaleoni, Sairam Yeleshwarapu, Saikrishna Adoni, Tapesh Maheshwari, Raghunandan P., and many others to whom I apologize for not citing.

Frederic Detienne

Distinguished Services Engineer

Cisco

Icons Used in This Book



Command Syntax Conventions

The conventions used to present command syntax in this book are the same conventions used in the IOS Command Reference. The Command Reference describes these conventions as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italics* indicate arguments for which you supply actual values.
- Vertical bars (|) separate alternative, mutually exclusive elements.
- Square brackets [ ] indicate optional elements.
- Braces { } indicate a required choice.
- Braces within brackets [{ }] indicate a required choice within an optional element.

Introduction

The motivation for writing this book was to educate users and customers about the benefits that FlexVPN and IKEv2 bring and provide an in-depth coverage of the building blocks and topics related to IPsec VPNs in general, in an easy-to-understand manner. FlexVPN was a breath of fresh air with regards to VPN technologies; for the first time all VPN technologies could be configured under a single CLI construct. We want to educate users so that secure, efficient VPN technologies can be implemented not only using Cisco IOS, but with third-party equipment also.

FlexVPN has allowed IKEv2 and IPsec VPNs on Cisco IOS to become a lot more user friendly; IKE, IPsec, concepts of cryptography, and VPNs can be hard subjects to understand. This book is intended to explain these topics and allow the reader to not only grasp the concepts, but master them.

The books explains how IPsec VPNs deal with NAT traversal, fragmentation, segmentation, IP dual stack, multicast, non-IP protocols and so on.

When VPNs are configured, there are a plethora of options; this book is intended to clarify these with ample illustrations and configuration examples so technologies are implemented in a secure and streamlined fashion.

When we talk to customers, many are unaware of what's happening under the hood and what impact a certain command will have. This book tries to clarify these points.

Goals and Methods

Provide a guide that will take the reader from knowing very little about VPN technologies to having an in-depth understanding.

Prevent customers from making mistakes that lead to network down scenarios or put the overall architecture at risk.

Give architects an understanding of the technology, allowing them to design VPN systems that meet business needs.

Give designers the knowledge where to position certain features.

Give implementors the understanding of how various features work along with end-to-end configurations examples that can be used as reference.

Give support staff an understanding of the protocols, configuring them, and how features integrate. This will result in a deep understanding, enabling timely debugging and troubleshooting.

Provide Security Operation Center guidance on telemetry that can be gained when an IKE and IPsec SA are created. This provides a methodology to perform monitoring and troubleshooting.

Provide advice and guidance on how to migrate existing IKEv1 architecture to using IKEv2.

Allow accreditors to understanding technologies, resulting in assurance that the architecture presented will meet the intended security requirements.

Give project managers an understanding of the components required to perform migrations from IKEv1 to IKEv2.

Who Should Read This Book?

Anyone that is involved with the lifecycle of deploying an IPsec VPN. This includes architects, designers, security engineers, support engineers, accreditors, and members of a SOC/NOC.

This book tries to explain the protocols at an RFC level, so it will provide the reader with an understanding that is not just specific to Cisco, but is applicable to any standards-based implementation.

For any individual that is developing services that are consumed by an IPsec VPN architecture (RADIUS, PKI, and similar ones), this book allows the reader to understand the protocol flows and the interaction between IKEv2/IPsec and their services.

VPN technologies are an integral part of the many of the Cisco certification tracks. This book would serve as a valuable study aid providing an in-depth coverage of the IPsec VPN foundational topics in an easy to understand manner. Some of such certification tracks are

- Security-CCNA, CCNP and CCIE
- Routing and Switching-CCNA, CCNP and CCIE
- Design-CCDA, CCDP, CCDE
- Service provider-CCIE

Simply put—if you want to understand IPsec VPN building blocks and architectures, and deploying IPsec VPNs when using IKEv2 this book is for you.

How this book is organized

This book contains a structured approached to VPN technologies.

Chapter 1 Introduction to IPsec VPNs

This chapter describes the purpose of VPNs and the types of cryptography (symmetric and asymmetric). We cover cryptographic protocols used in the generation of IPsec VPNs.

We explain how confidentiality and integrity are achieved using Encapsulation Security Payload (ESP) and how integrity is achieved using Authentication Header (AH).

We introduce IKE and IPsec and the relationship that these have.

This chapter describes the components that make up IPsec, including the Security Parameter Index (SPI), Security Policy Database (SPD), Security Association Database (SADB), Peer Authorization Database (PAD), lifetime, and sequence numbers. We explain how these are interlinked and what relationship exists.

The two modes of IPsec, tunnel and transport, are described. We explain the benefits of each. The benefits of ESP version 3 are described.

Chapter 2 IKEv2 the Protocol

The Internet Key Exchange (IKE) protocol is described in detail. The format of the IKE header and the various packet exchanges (SA_INIT, IKE_AUTH, INFORMATIONAL, CREATE_CHILD_SA) are described. You will understand how the IKE SA is created and the components that are used to construct this, such as key material generation.

Features of IKEv2, such as anti-replay, the anti-DDoS cookie, configuration payload, and acknowledged responses, are described along with the protocols used by IKE; encryption, integrity, PRF, and Diffie-Hellman are listed.

This chapter details how IKEv2 operates when NAT is used on the transport network. The various keepalive mechanisms are covered, including IKE and NAT keepalives. This chapter covers a number of additional IKEv2 related RFCs.

Chapter 3 Comparison of IKEv1 and IKEv2

Within this chapter the history of IPsec and IKEv1 is covered, including all the RFCs (2401 to 2412) that were created to define the implementation of IKEv1-based IPsec VPNs. The key similarities and the key differences of IKEv2 compared to IKEv1 are covered, including exchange modes, authentication, use of identities, anti-DDoS, lifetimes, and many more topics.

Chapter 4 IOS IPsec implementation

The specific types of VPN implementation of Cisco IOS and IOS-XE are introduced. This chapter describes how to implement tunnel or transport mode. The two encapsulation types, GRE and VTI, are described, along with their benefits and limitations. The various implementation modes (dual stack, mixed mode, and auto) are covered. We also introduce VRF-aware IPsec.

Chapter 5 IKEv2 Configuration

This chapter contains an overview of the IKEv2 configuration features and how these interoperate. The various components of IKEv2 are covered: IKEv2 proposal, IKEv2 policy, IKEv2 profile, IKEv2 keyring, and the IKEv2 global configuration. We also cover other components that are critical to configuring IPsec VPNs, such as PKI and IPsec. The powerful pre-configured attributes are introduced, and their benefits are explained.

Chapter 6 Advanced IKEv2 features

This chapter covers IKEv2 advanced features, including some that are not part of the standard IKEv2 RFC. IKEv2 fragmentation and the transportation of Security Group Tags (SGT) are described, along with the methods to delete a session should a peer be revoked or the peer's certificate expire. The lifetime of the IKEv2 session is examined and the effect this has is described in detail.

Chapter 7 IKEv2 deployments

This chapter described a number of scenarios to give the reader an understanding of the various types of IKEv2 deployments. Both IPv4 and IPv6 are covered, with authentication using pre-shared keys, RSA certificates, ECDSA certificates, and HTTP URL Cert. The IKE anti-DDoS mechanism is illustrated in detail.

Chapter 8 Introduction to FlexVPN

After an overview of FlexVPN, the tunnel interface types (static, virtual-template, and virtual-access) and IOS AAA infrastructure are described in detail. The building blocks of FlexVPN—Name Mangler, IKEv2 authorization policy, with user, group. and implicit authorization—are described. The configuration exchange is illustrated, along with advertising prefixes using IKEv2 routing.

Chapter 9 FlexVPN Server

The chapter provides an overview of FlexVPN Server. EAP authentication is described in detail, along with AAA-based pre-shared keys. Deriving virtual-access interfaces from virtual-templates is illustrated, along with automatic detection of the tunnel mode and encapsulation type using mode auto. RADIUS Packet of Disconnect and Change of Authorization (CoA) are described. The IKEv2 auto-reconnect, AnyConnect-EAP, and dual-factor authentication features are described. The FlexVPN Server supported clients are covered.

Chapter 10 FlexVPN Client

This chapter begins with an overview of FlexVPN Client. EAP authentication is described in detail. Client-specific attributes are described: split-DNS, WINS, and Domain Name. The FlexVPN client profile is described. The following specific features of FlexVPN client are illustrated: Backup gateways, dial backup, backup groups, tunnel interface types, tunnel initiation types, and FlexVPN with NAT. This chapter describes design considerations and troubleshooting specific to the FlexVPN client.

Chapter 11 FlexVPN Load Balancer

This chapter presents an overview of the FlexVPN Load Balancer. It details the core components and RFC 5685 "IKEv2 Redirect and Hot Standby Routing Protocol." How the cluster operates, including cluster load, is detailed. FlexVPN client and server configurations are illustrated. Troubleshooting a specific FlexVPN load balancer configuration is described, and a number of example configurations are shown.

Chapter 12 FlexVPN Deployments

This chapter contains a number of example scenarios which illustrate the following FlexVPN deployments: AAA-based pre-shared key, user and group authorization, FlexVPN routing with dual-stack and tunnel mode auto, NAT with server-assigned IP addresses, WAN resilient using dynamic tunnel source, hub resiliency using backup peers, and FlexVPN backup tunnel using track-based activation.

Chapter 13 Monitoring IPsec VPNs

This chapter describes common methods for monitoring IPsec VPNs using AAA, SNMP, and syslog. A monitoring methodology is described that covers IP connectivity, VPN tunnel establishment, authentication, authorization, data encapsulation, data encryption, and overlay routing.

Chapter 14 Troubleshooting IPsec VPNs

This chapter describes the tools of troubleshooting: Event Trace Monitoring, IKEv2, IPsec, KMI and conditional debugging. Troubleshooting steps are described for IP connectivity, VPN tunnel establishment, authentication, authorization, data encapsulation, data encryption, and overlay routing.

Chapter 15 IPsec overhead and Fragmentation

This chapter describes computing IPsec overhead for ESP and AH and the effect that IPsec and fragmentation have for both IPv4 and IPv6. The following topics are illustrated: Path MTU Discovery (PMTUD), TCP MSS clamping, fragmentation, and PMTUD (specifically on tunnel interfaces). The impact of fragmentation is described.

Chapter 16 Migration Strategies

The chapter illustrates the considerations when migrating from IKEv1 to IKEv2. It covers hardware, VPN technologies, routing protocols, restrictions for IKEv1 and IKEv2, capacity planning, global commands, FlexVPN features, PKI authentication, high availability, and asymmetric routing. Migration strategies for hard and soft migrations are covered. It also discusses considerations for specific topologies: site-to-site, hub-and-spoke, and remote access. There is also an example migration scenario.

This page intentionally left blank

Chapter 7

IKEv2 Deployments

This chapter introduces a number of designs where IKEv2 is used. Each design will use a simple deployment of two routers with the focus on the configuration of IKEv2. Although each scenario uses only two routers, the configuration can scale as required if needed.

The configuration is intended to be as simple as possible, and the emphasis is focused on the IKEv2 configuration.

Pre-shared-key Authentication with Smart Defaults

This configuration is the simplest to set up. By using smart defaults, a VPN is created between two peers using minimal configuration: only the IKEv2 profile and corresponding IKEv2 keyring are required.

Figure 7-1 illustrates the topology. The transport network is using IPv6, and the overlay network is using IPv4.

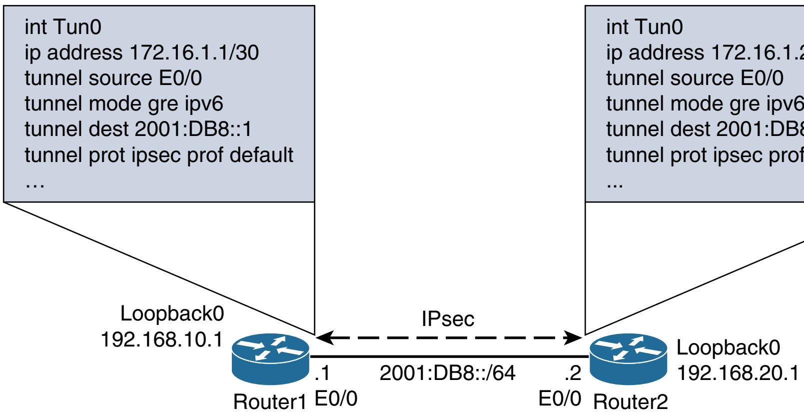


Figure 7-1 *PSK Authentication with Smart Defaults Topology*

The following example illustrates the relevant configuration used on Router1. This is a very minimal configuration which leaves little room for error.

Note that the shared secrets used in the example below are for illustrative purposes and, if used in a production environment, should contain sufficient entropy.

The example might seem complex as this scenario uses IPv4 and IPv6; however, the main focus of interest is to illustrate the IKEv2 configuration and the simplicity of using smart defaults.

An IKEv2 keyring is created with a peer entry which matches the peer's IPv6 address. Asymmetric pre-shared-keys are used with each device having a unique local and remote key.

```
crypto ikev2 keyring local_keyring
 peer 2001:DB8::2
  address 2001:DB8::2/128
  pre-shared-key local bartlett
  pre-shared-key remote inamdar
```

The IKEv2 profile is the mandatory component and matches the remote IPv6 address configured on Router2. The local IKEv2 identity is set to the IPv6 address configured on E0/0. The authentication is set to pre-shared-key with the locally configured keyring defined previously.

```
crypto ikev2 profile default
 match identity remote address 2001:DB8::2/128
 identity local address 2001:DB8::1
 authentication remote pre-share
 authentication local pre-share
 keyring local local_keyring
```

The local loopback interface is configured, which will allow testing over the IPsec Security Association.

```
interface Loopback0
 ip address 192.168.10.1 255.255.255.0
```

The tunnel interface is created as tunnel mode GRE IPv6. This is required as the transport network is IPv6 and the overlay is IPv4. The default IPsec profile is used to protect this interface; this uses the default IKEv2 profile which was configured earlier.

```
interface Tunnel0
 ip address 172.16.1.1 255.255.255.252
 tunnel source Ethernet0/0
 tunnel mode gre ipv6
 tunnel destination 2001:DB8::2
 tunnel protection ipsec profile default
```

The physical interface used as the tunnel source uses IPv6.

```
interface Ethernet0/0
 no ip address
 ipv6 address 2001:DB8::1/64
```

Enhanced interior gateway routing protocol (EIGRP) is used to establish a peer relationship over the tunnel interface and distribute the loopback prefix.

```
router eigrp 1
 network 172.16.1.0 0.0.0.3
 network 192.168.10.0
```

The following example illustrates the relevant configuration on Router2.

```
interface Loopback0
 ip address 192.168.20.1 255.255.255.0

interface Tunnel0
 ip address 172.16.1.2 255.255.255.252
 tunnel source Ethernet0/0
 tunnel mode gre ipv6
 tunnel destination 2001:DB8::1
 tunnel protection ipsec profile default

interface Ethernet0/0
 no ip address
 ipv6 address 2001:DB8::2/64

router eigrp 1
 network 172.16.1.0 0.0.0.3
 network 192.168.20.0
```

The following example illustrates the EIGRP neighbor relationship built over the tunnel interface. The prefix for IP address assigned to the loopback interface on Router2 is reachable via the protected tunnel.

```
Router1#show ip route 192.168.20.0
Routing entry for 192.168.20.0/24
  Known via "eigrp 1", distance 90, metric 27008000, type internal
  Redistributing via eigrp 1
  Last update from 172.16.1.2 on Tunnel0, 00:12:04 ago
  Routing Descriptor Blocks:
  * 172.16.1.2, from 172.16.1.2, 00:12:04 ago, via Tunnel0
      Route metric is 27008000, traffic share count is 1
      Total delay is 55000 microseconds, minimum bandwidth is 100 Kbit
      Reliability 255/255, minimum MTU 1418 bytes
      Loading 1/255, Hops 1
```

The following example illustrates the IKEv2 SA that is created. The IKEv2 SA is protected by the PRF and integrity algorithms using SHA512, encryption using AES-CBC-256, and Diffie-Hellman group 5, which are the most preferred algorithms within the IKEv2 default proposal. The authentication is performed using pre-shared-key.

```
Router1#show crypto ikev2 sa detailed
 IPv4 Crypto IKEv2  SA

 IPv6 Crypto IKEv2  SA

Tunnel-id    fvrf/ivrf              Status
1          none/none             READY
Local  2001:DB8::1/500
Remote  2001:DB8::2/500
      Encr: AES-CBC, keysize: 256, PRF: SHA512, Hash: SHA512, DH Grp:5, Auth sign:
PSK, Auth verify: PSK
      Life/Active Time: 86400/10523 sec
      CE id: 1002, Session-id: 2
      Status Description: Negotiation done
      Local spi: 261B9BD2F208A02A       Remote spi: 0B28D2A21FC6304D
      Local id: 2001:DB8::1
      Remote id: 2001:DB8::2
      Local req msg id:  4              Remote req msg id:  4
      Local next msg id: 4              Remote next msg id: 4
      Local req queued:  4              Remote req queued:  4
      Local window:      5              Remote window:      5
…
```

The following example illustrates traffic being sent over the IPsec Security Association. The tunnel source and destination being the IPv6 addresses configured on the physical E0/0 interfaces.

Traffic is sent via the tunnel interface, from the locally configured loopback interface to the loopback on Router2.

```
Router1#ping 192.168.20.1 source 192.168.10.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.20.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.10.1

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/5 ms
```

The IPsec Security Association is verified where the default IPsec transform set is used, which is created using Encapsulation Security Payload with AES-CBC-256 for encryption and SHA1-HMAC for integrity. Transport mode is used.

```
Router1#show crypto ipsec sa

interface: Tunnel0
    Crypto-map tag: Tunnel0-head-0, local addr 2001:DB8::1

   protected vrf: (none)
   local  ident (addr/mask/prot/port): (2001:DB8::1/128/47/0)
   remote ident (addr/mask/prot/port): (2001:DB8::2/128/47/0)
   current_peer 2001:DB8::2 port 500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 523, #pkts encrypt: 523, #pkts digest: 523
    #pkts decaps: 523, #pkts decrypt: 523, #pkts verify: 523
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 0, #recv errors 0

     local crypto endpt.: 2001:DB8::1,
     remote crypto endpt.: 2001:DB8::2
     plaintext mtu 1462, path mtu 1500, ipv6 mtu 1500, ipv6 mtu idb Ethernet0/0
     current outbound spi: 0x5FC3C94A(1606666570)
     PFS (Y/N): N, DH group: none

     inbound esp sas:
      spi: 0xB8435B94(3091422100)
        transform: esp-aes esp-sha-hmac ,
        in use settings ={Transport, }
        conn id: 10, flow_id: SW:10, sibling_flags 80000001, crypto-map:
Tunnel0-head-0
        sa timing: remaining key lifetime (k/sec): (4315844/2543)
        IV size: 16 bytes
        replay detection support: Y
        Status: ACTIVE(ACTIVE)

     inbound ah sas:

     inbound pcp sas:

     outbound esp sas:
      spi: 0x5FC3C94A(1606666570)
        transform: esp-aes esp-sha-hmac ,
        in use settings ={Transport, }
        conn id: 9, flow_id: SW:9, sibling_flags 80000001, crypto-map:
Tunnel0-head-0
```

```
sa timing: remaining key lifetime (k/sec): (4315844/2543)
IV size: 16 bytes
replay detection support: Y
Status: ACTIVE(ACTIVE)
```

Elliptic Curve Digital Signature Algorithm Authentication

The scenario looks to use digital signatures to authenticate both peers. Rather than the more common RSA certificates, Elliptic Curve (EC) certificates are used that provide the ability to authenticate both parties, using the Elliptic Curve Digital Signature Algorithm (ECDSA).

The configuration in this example is intended to be simple, with the main focus on the IKEv2 configuration.

Figure 7-2 illustrates the physical IP addressing and the setup of the tunnel interface.

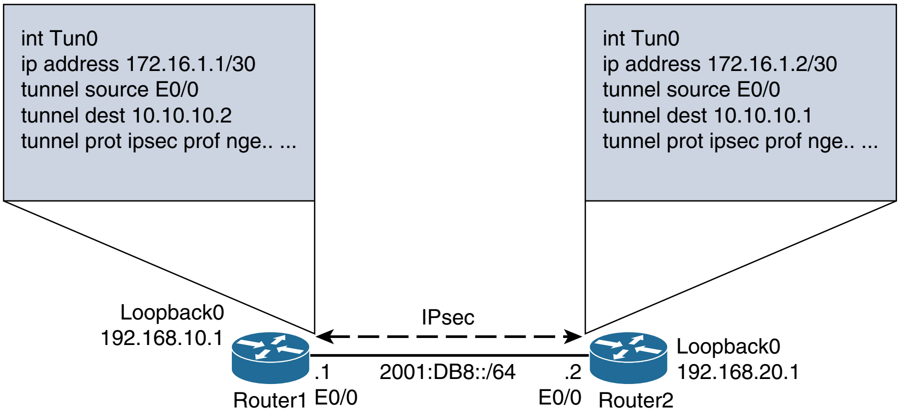


Figure 7-2 *Topology with Configuration*

In addition to ECDSA for authentication, Cisco Next Generation Encryption (NGE) algorithms secure the IKEv2 and IPsec session, as shown in Table 7-1.

Table 7-1 *Security Algorithms Used*

Method	Algorithm
IKEv2 encryption	AES-GCM-256
IKEv2 PRF	SHA512
Diffie-Hellman	Group 21
Authentication	Elliptic Curve Digital Signature Algorithm
IPsec encryption	AES-GCM-256
IPsec PFS	Group 21

Rather than using the default IKEv2 proposal, the default IKEv2 proposal is disabled, and a new IKEv2 proposal created containing the IKEv2 algorithms defined in Table 7-1.

Static routes are used to send traffic down the freshly created tunnel interface.

The following example illustrates the configuration that is used on Router1.

Note Although not shown, the trustpoint uses a locally configured elliptic curve keypair.

The trustpoint is configured using manual enrollment, with the local and CA certificate.

```
crypto pki trustpoint ecdsa_tp
 enrollment terminal

crypto pki certificate chain ecdsa_tp
 certificate 6156E3D5000000000009
  308203BF 30820365 A0030201 02020A61 56E3D500 00000000 09300A06 082A8648
…
  68656c6c 6f736861 627333E4FDDC 642DA416 F57D4962 C5DF6545 FEC931FA F84BAF40
  A9829E
      quit
 certificate ca 780887F0CDD97E9E49DB893FA5D74238
  30820206 308201AB A0030201 02021078 0887F0CD D97E9E49 DB893FA5 D7423830
  0A06082A 8648CE3D 04030230 4F311330 11060A09 92268993 F22C6401 19160363
…
  816AA443 9191FBAC 731C
      quit
```

A certificate map is created that will match certificates containing a subject name of *cisco.com*. This is used within the IKEv2 profile to anchor the certificates presented by the peers. As this is a site-to-site VPN with only two peers, the certificate map could have been more granular to include the peer DN.

```
crypto pki certificate map certmap 10
 subject-name co cisco.com
```

The default IKEv2 proposal is disabled, and a new IKEv2 proposal is created that contains the relevant cryptographic algorithms.

Note Because this is a combined mode cipher, no integrity algorithm is required.

```
no crypto ikev2 proposal default
crypto ikev2 proposal nge
 encryption aes-gcm-256
 prf sha512
 group 21
```

An IKEv2 policy is created, which encompasses the IKEv2 proposal created above. Because the default IKEv2 proposal is disabled, this then ensures that only the IKEv2 proposal named *nge* will be used and minimizes the chance of mis-configuration.

```
crypto ikev2 policy default
 match fvrf any
 proposal nge
```

An IKEv2 profile is created, which uses the certificate map created earlier. The identity is set to DN, which will use the DN from the certificate. The authentication method is set to ECDSA and the PKI trustpoint used which was configured earlier. This profile will only match peer certificates, which contain the string *cisco.com* within the subject name. Dead-peer detection is enabled to ensure that the IKEv2 SA and corresponding IPsec Security Associations are torn down in a timely manner if IKE connectivity is lost.

```
crypto ikev2 profile nge
 match certificate certmap
 identity local dn
 authentication remote ecdsa-sig
 authentication local ecdsa-sig
 pki trustpoint ecdsa_tp
 dpd 10 2 on-demand
```

An IPsec transform set is created, which uses AES-GCM-256. Because this is a combined mode cipher, no integrity algorithm is required.

```
crypto ipsec transform-set nge-transform esp-gcm 256
 mode transport
```

The default IPsec profile is disabled, which ensures that it is not used due to mis-configuration. A new IPsec profile is created which uses the IKEv2 profile and IPsec transform-set created earlier. Additionally, perfect forward secrecy is enabled to ensure that a fresh Diffie-Hellman exchange is performed on rekey.

```
no crypto ipsec profile default

crypto ipsec profile nge-profile
 set transform-set nge-transform
 set pfs group21
 set ikev2-profile nge
```

A loopback interface is used that will allow traffic to be sourced from and destined to as it transverses the VPN.

```
interface Loopback0
 ip address 192.168.10.1 255.255.255.0
```

The tunnel interface is created with the relevant source interface configured and with the destination address of Router2. This is protected by the IPsec profile created above.

```
interface Tunnel0
 ip address 172.16.1.1 255.255.255.252
 tunnel source Ethernet0/0
 tunnel destination 10.10.10.2
 tunnel protection ipsec profile nge-profile
```

The E0/0 interface is used as the tunnel source.

```
interface Ethernet0/0
 ip address 10.10.10.1 255.255.255.0
```

A static route is configured to send all traffic for the 192.168.20.0/24 network, which is the subnet protected by the peer, via the peer tunnel IP address.

```
ip route 192.168.20.0 255.255.255.0 172.16.1.2
```

Router2 has a nearly similar configuration; the following example illustrates the unique configuration. The tunnel interface has a unique IP address, and the destination is configured as E0/0 on Router1.

Note the unique IP address and the tunnel destination of Router1.

```
interface Tunnel0
 ip address 172.16.1.2 255.255.255.252
 tunnel source Ethernet0/0
 tunnel destination 10.10.10.1
 tunnel protection ipsec profile nge-profile

interface Ethernet0/0
 ip address 10.10.10.2 255.255.255.0
```

The following example illustrates verification that the IKEv2 SA established. The algorithms used to secure the IKE session as described in Table 7-1 can be seen.

```
Router1#show crypto ikev2 sa detailed
 IPv4 Crypto IKEv2  SA

Tunnel-id Local                 Remote                fvrf/ivrf            Status
1         10.10.10.1/500        10.10.10.2/500        none/none            READY
      Encr: AES-GCM, keysize: 256, PRF: SHA512, Hash: None, DH Grp:21, Auth sign:
ECDSA, Auth verify: ECDSA
      Life/Active Time: 86400/6 sec
      CE id: 1030, Session-id: 13
```

```
      Status Description: Negotiation done
      Local spi: 313404E23B3A5707       Remote spi: 13FE5BCC09FFAAAB
      Local id: hostname=Router1.cisco.com
      Remote id: hostname=Router2.cisco.com
      Local req msg id:  2              Remote req msg id:  0
      Local next msg id: 2              Remote next msg id: 0
      Local req queued:  2              Remote req queued:  0
      Local window:      5              Remote window:      5
      DPD configured for 10 seconds, retry 2
      Fragmentation not configured.
      Extended Authentication not configured.
      NAT-T is not detected
      Cisco Trust Security SGT is disabled
      Initiator of SA : Yes

 IPv6 Crypto IKEv2  SA
```

The creation of the IPsec Security Association can be seen in the following example. The tunnel interface is configured with the default GRE mode, the traffic selectors can be seen indicating this by the use of IP protocol 47.

```
Router1#show crypto sockets

Number of Crypto Socket connections 1

   Tu0 Peers (local/remote): 10.10.10.1/10.10.10.2
       Local Ident  (addr/mask/port/prot): (10.10.10.1/255.255.255.255/0/47)
       Remote Ident (addr/mask/port/prot): (10.10.10.2/255.255.255.255/0/47)
       IPSec Profile: "nge-profile"
       Socket State: Open
       Client: "TUNNEL SEC" (Client State: Active)
Crypto Sockets in Listen state:
Client: "TUNNEL SEC" Profile: "nge-profile" Map-name: "Tunnel0-head-0"
```

The following example illustrates the route to 192.168.20.0/24, which be seen via the tunnel interface. All traffic intended for this network will be sent via the tunnel and encrypted by the corresponding IPsec Security Association.

```
Router1#show ip route 192.168.20.0 255.255.255.0
Routing entry for 192.168.20.0/24
  Known via "static", distance 1, metric 0
  Routing Descriptor Blocks:
  * 172.16.1.2
      Route metric is 0, traffic share count is 1
Router1#show ip route 172.16.1.2
Routing entry for 172.16.1.0/30
```

```
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via Tunnel0
      Route metric is 0, traffic share count is 1
```

Traffic is sent from Router1 to Router2 via the tunnel interface. Note that this traffic has been protected by the IPsec Security Association, as indicated by the increasing *encaps* and *decaps* counters.

```
Router1#ping 192.168.20.1 source 192.168.10.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.20.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.10.1

Success rate is 100 percent (5/5), round-trip min/avg/max = 3/4/6 ms

Router1#show crypto ipsec sa

interface: Tunnel0
    Crypto-map tag: Tunnel0-head-0, local addr 10.10.10.1

   protected vrf: (none)
   local  ident (addr/mask/prot/port): (10.10.10.1/255.255.255.255/47/0)
   remote ident (addr/mask/prot/port): (10.10.10.2/255.255.255.255/47/0)
   current_peer 10.10.10.2 port 500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 10, #pkts encrypt: 10, #pkts digest: 10
    #pkts decaps: 10, #pkts decrypt: 10, #pkts verify: 10
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 0, #recv errors 0

     local crypto endpt.: 10.10.10.1, remote crypto endpt.: 10.10.10.2
     plaintext mtu 1466, path mtu 1500, ip mtu 1500, ip mtu idb Ethernet0/0
     current outbound spi: 0x3FD4A2AF(1070899887)
     PFS (Y/N): Y, DH group: group21

     inbound esp sas:
      spi: 0x349334C6(882062534)
        transform: esp-gcm 256 ,
        in use settings ={Transport, }
        conn id: 6, flow_id: SW:6, sibling_flags 80000000, crypto-map:
Tunnel0-head-0
        sa timing: remaining key lifetime (k/sec): (4207250/3566)
        IV size: 8 bytes
```

```
        replay detection support: Y
        Status: ACTIVE(ACTIVE)

     inbound ah sas:

     inbound pcp sas:

     outbound esp sas:
      spi: 0x3FD4A2AF(1070899887)
        transform: esp-gcm 256 ,
        in use settings ={Transport, }
        conn id: 5, flow_id: SW:5, sibling_flags 80000000, crypto-map:
Tunnel0-head-0
        sa timing: remaining key lifetime (k/sec): (4207250/3566)
        IV size: 8 bytes
        replay detection support: Y
        Status: ACTIVE(ACTIVE)

     outbound ah sas:

     outbound pcp sas:
```

RSA Authentication Using HTTP URL Lookup

In this scenario, we will use RSA certificates to authenticate both peers. However, for Router2, we will not send the certificate within the IKE AUTH exchange, but will send a HTTP URL from Router2 to Router1 to inform it where to obtain the certificate. Router1 will then retrieve the certificate from the HTTP URL and verify that the presented AUTH payload was signed by the private key relating to the public key contained within the certificate.

Router1 has been set up as a certificate authority; from this CA, a certificate is obtained for both Router1 and Router2. These certificates are used to authenticate the IKEv2 SA.

Figure 7-3 illustrates the operation of the HTTP URL lookup feature. Router2 will sign the AUTH payload with its private key. Router1 will retrieve the certificate from the HTTP server and validate the AUTH payload by using the public key obtained from the retrieved certificate.

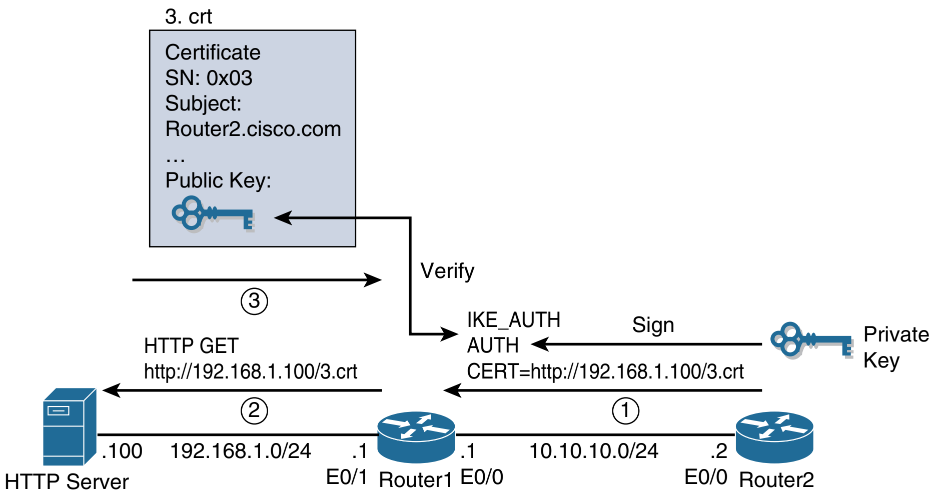


Figure 7-3 *HTTP URL Lookup Feature*

> **Note** The certificate generated by the IOS CA is in Privacy Enhanced Mail (PEM) format. Although the IKEv2 RFC states that the HASH and URL feature returns a URL with the SHA1 hash of the requested certificate, Cisco IOS allows for any URL to be used. As per the IKEv2 RFC, Cisco IOS requires the obtained certificate to be in distinguished encoding rules (DER) encoding. The following example illustrates the OpenSSL commands to manually convert a certificate from PEM to DER encoding, with the PEM encoded certificate in file 3.crt.

```
openssl x509 -outform der -in 3.crt -out 3.der
```

Figure 7-4 illustrates the topology used in the tunnel interface configuration.

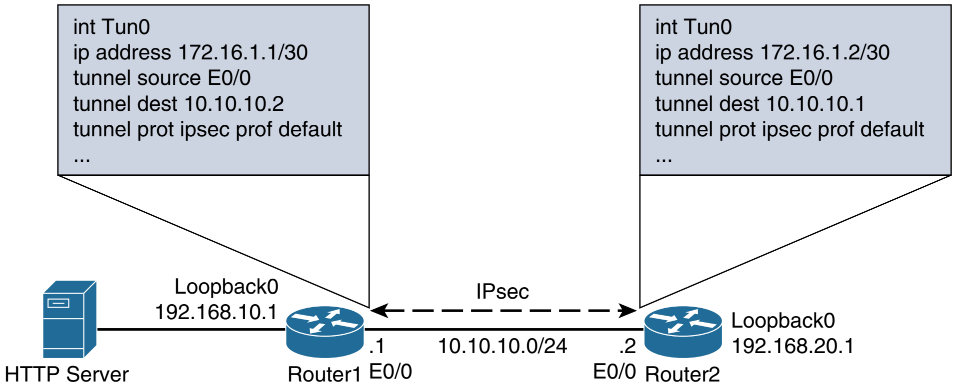


Figure 7-4 *Topology with Configuration*

The configuration is similar to the ECDSA example earlier, but RSA certificates are used, which results in a different authentication method. However, the base concepts are the same with regards to the PKI.

The subject information access (SIA) is an attribute within a certificate that defines some type of offered services. An example of where to access a server can be included in the SIA with a uniform resource identifier (URI). The SIA is amended to contain the URL that the peer will use for the HTTP URL lookup. This is achieved by the use of the certificate map that matches the locally used certificate and is attached to the trustpoint. This removes the inclusion of the certificate within the IKE exchange and uses the value defined in the SIA as the location for the peer to obtain the certificate.

The following example illustrates the configuration used on Router2.

The PKI trustpoint is defined; it has been authenticated, and the local device enrolled. The critical component to ensure that this client does not send its certificate but instead sends the HTTP URL is the *match certificate* command. This command will match the defined certificate map and override the SIA to contain the configured URL. This is then sent in replacement of the certificate in the IKE_AUTH exchange.

```
crypto pki trustpoint CA
 enrollment url http://10.10.10.1:80
 revocation-check crl
 match certificate local override sia 1 http://192.168.1.100/3.der
```

A certificate map is created that will match certificates containing a subject name of *router1.cisco.com*. This is used within the IKEv2 profile to anchor the peer's presented certificate.

```
crypto pki certificate map certmap 10
 subject-name eq router1.cisco.com
```

The following certificate map is used by the match statement within the trustpoint configuration to match the local certificate. This is achieved by matching the local subject name (which is not case sensitive) of *router2*.

```
crypto pki certificate map local 10
 subject-name co router2
```

The mandatory IKEv2 profile is configured which uses the certificate map created earlier. This will match any certificates which contain a subject name of *cisco.com*. The authentication method is set to RSA signatures, and the trustpoint configured earlier is used.

```
crypto ikev2 profile default
 match certificate certmap
 identity local dn
 authentication remote rsa-sig
 authentication local rsa-sig
 pki trustpoint CA
```

The tunnel interface is created with the relevant source interface configured and the destination address of Router1. This is protected by the default IPsec profile which uses the default IKEv2 profile which was created earlier.

```
interface Tunnel0
 ip address 172.16.1.2 255.255.255.0
 tunnel source Ethernet0/0
 tunnel destination 10.10.10.1
 tunnel protection ipsec profile default
```

The following physical interface is used as the tunnel source.

```
interface Ethernet0/0
 ip address 10.10.10.2 255.255.255.0
```

Note Should a certificate hierarchy exist where there is a requirement to send a certificate chain with multiple URLs in multiple CERT payloads starting from "ID cert url," "subca1," "subca2," until "root CA"; then each additional certificate can be included as a separate line within the trustpoint configuration as illustrated below.

```
crypto pki trustpoint CA
 match certificate local override sia 1 http://192.168.1.100/3.der
 match certificate local override sia 2 http://192.168.1.100/subca1.der
 match certificate local override sia 3 http://192.168.1.100/subca2.der
 match certificate local override sia 4 http://192.168.1.100/root.der
```

The following example illustrates the configuration used on Router1.

The certificate authority function is enabled. Note that the automatic granting of certificates is used here for ease of configuration and should not occur in a production environment where un-authenticated access to the CA can occur.

```
crypto pki server local
 database level complete
 no database archive
 grant auto
```

The relating PKI trustpoint for the IOS CA is:

```
crypto pki trustpoint local
 revocation-check crl
 rsakeypair local
```

A trustpoint is used to enroll into the local CA.

```
crypto pki trustpoint CA
 enrollment url http://10.10.10.1:80
 revocation-check crl
```

A certificate map is created that will match certificates containing a subject name of *router2.cisco.com*. This is used within the IKEv2 profile to anchor the peer's presented certificate.

```
crypto pki certificate map certmap 10
 subject-name eq router2.cisco.com
```

The mandatory IKEv2 profile is configured that uses the certificate map created earlier. This will match any certificates, which contain a subject name of *cisco.com*. The authentication method is set to RSA signatures, and the trustpoint configured earlier is used.

```
crypto ikev2 profile default
 match certificate certmap
 identity local dn
 authentication remote rsa-sig
 authentication local rsa-sig
 pki trustpoint CA
```

The tunnel interface is created with the relevant source interface configured, and the destination address of Router1. This is protected by the default IPsec profile that uses the default IKEv2 profile, which was created earlier.

```
interface Tunnel1
 ip address 172.16.1.1 255.255.255.0
 tunnel source Ethernet0/0
 tunnel destination 10.10.10.2
 tunnel protection ipsec profile default
```

The physical interface used as the tunnel source.

```
interface Ethernet0/0
 ip address 10.10.10.1 255.255.255.0
```

The physical interface used to reach the HTTP server containing the certificates.

```
interface Ethernet0/1
 ip address 192.168.1.1 255.255.255.0
```

Note When using the HTTP URL lookup feature, the router that retrieves the HTTP URL should be protected from malicious intent by restricting HTTP access to only the server storing the certificates. As the certificate obtained via the HTTL URL method is processed prior to authentication, an intruder could redirect the gateway to a large file containing garbage, or a URI that will slowly introduce a file, a little at a time, causing a DoS on the gateway. Mitigation can be achieved using controls, such as access-control-lists, control-plane policing, or control-plane protection.

The following example illustrates IKEv2 debugs taken from Router1. It can be seen that Router2 sends the IKE_AUTH exchange with the CERT payload containing the HASH and URL format. Also note the NOTIFY payload which indicates the HTTP URL method is supported.

```
IKEv2:(SESSION ID = 4,SA ID = 1):Received Packet [From 10.10.10.2:500/To
10.10.10.1:500/VRF i0:f0]
Initiator SPI : 52D538043A8E330C - Responder SPI : 5CE063D07E8745EA Message id: 1
IKEv2 IKE_AUTH Exchange REQUEST
IKEv2-PAK:(SESSION ID = 4,SA ID = 1):Next payload: ENCR, version: 2.0 Exchange
type: IKE_AUTH, flags: INITIATOR Message id: 1, length: 816
Payload contents:
 VID  Next payload: IDi, reserved: 0x0, length: 20
 IDi  Next payload: CERT, reserved: 0x0, length: 44
    Id type: DER ASN1 DN, Reserved: 0x0 0x0
 CERT  Next payload: CERTREQ, reserved: 0x0, length: 52
    Cert encoding Hash and URL of PKIX
 CERTREQ  Next payload: NOTIFY, reserved: 0x0, length: 25
    Cert encoding Hash and URL of PKIX
 NOTIFY(HTTP_CERT_LOOKUP_SUPPORTED)  Next payload: AUTH, reserved: 0x0, length: 8
    Security protocol id: Unknown - 0, spi size: 0, type:
HTTP_CERT_LOOKUP_SUPPORTED
 AUTH  Next payload: CFG, reserved: 0x0, length: 136
    Auth method RSA, reserved: 0x0, reserved 0x0
 CFG  Next payload: SA, reserved: 0x0, length: 304
    cfg type: CFG_REQUEST, reserved: 0x0, reserved: 0x0
```

A short time later, Router1 opens a TCP socket with 192.168.1.100, when the certificate is obtained.

```
TCP: sending SYN, seq 2191097267, ack 0
TCP0: Connection to 192.168.1.100:80, advertising MSS 1460
TCP0: state was CLOSED -> SYNSENT [42603 -> 192.168.1.100(80)]
TCP0: state was SYNSENT -> ESTAB [42603 -> 192.168.1.100(80)]
TCP: tcb 32417230 connection to 192.168.1.100:80, peer MSS 1460, MSS is 1460
```

The following example illustrates verification on Router1 that the certificate was obtained by way of HTTP.

```
Router1#show crypto ikev2 stats ext-service
--------------------------------------------------------------
AAA OPERATION                                   PASSED     FAILED
--------------------------------------------------------------
RECEIVING PSKEY                                     0          0
AUTHENTICATION USING EAP                            0          0
```

```
START ACCOUNTING                                 0          0
STOP ACCOUNTING                                  0          0
AUTHORIZATION                                    0          0
--------------------------------------------------------------
IPSEC OPERATION                             PASSED     FAILED
--------------------------------------------------------------
IPSEC POLICY VERIFICATION                        3          0
SA CREATION                                      3          0
SA DELETION                                      3          0
---------------------------------------------------------------
CRYPTO ENGINE OPERATION                     PASSED     FAILED
---------------------------------------------------------------
DH PUBKEY GENERATED                             34          0
DH SHARED SECKEY GENERATED                      29          0
SIGNATURE SIGN                                  28          0
SIGNATURE VERIFY                                 3          0
--------------------------------------------------------------
PKI OPERATION                               PASSED     FAILED
--------------------------------------------------------------
VERIFY CERTIFICATE                               3          0
FETCHING CERTIFICATE USING HTTP                  1          0
FETCHING PEER CERTIFICATE USING HTTP             1          0
GET ISSUERS                                     31          0
GET CERTIFICATES FROM ISSUERS                   28          0
GET DN FROM CERT                                 3          0
--------------------------------------------------------------
GKM OPERATION                               PASSED     FAILED
--------------------------------------------------------------
GET_POLICY                                       0          0
SET_POLICY                                       0          0
```

The certificate that is obtained via HTTP is cached locally. By default, 200 certificates will be cached. As the certificate is cached, if the IKE session drops and is re-established, the certificate will not be required to be obtained via HTTP as it is already cached. This saves numerous HTTP requests to occur if the peer is required to re-authenticate. The following example illustrates viewing the contents of the certificate cache.

```
Router1#show crypto ikev2 certificate-cache
No of entries in ikev2 certificate-cache = 1

Certificate entry:
Certificate
  Status: Available
  Certificate Serial Number (hex): 03
  Certificate Usage: General Purpose
```

```
  Issuer:
    cn=CA.cisco.com
  Subject:
    Name: Router2.cisco.com
    hostname=Router2.cisco.com
  Validity Date:
    start date: 10:44:26 UTC Feb 8 2016
    end   date: 10:44:26 UTC Feb 7 2017
  Associated Trustpoints:
```

The following example illustrates the IKEv2 SA being verified. The cryptographic algorithms used have been negotiated via the use of smart defaults. The authentication method of RSA can be seen. There is no differentiation that the certificate was received via the HTTP URL method; the authentication is performed in the same manner as RSA authentication when certificates are sent in the IKE_AUTH exchange.

```
Router1#show crypto ikev2 sa detailed
 IPv4 Crypto IKEv2  SA

Tunnel-id Local                 Remote                fvrf/ivrf            Status
1         10.10.10.1/500        10.10.10.2/500        none/none            READY
      Encr: AES-CBC, keysize: 256, PRF: SHA512, Hash: SHA512, DH Grp:5, Auth sign:
RSA, Auth verify: RSA
      Life/Active Time: 86400/509 sec
      CE id: 1034, Session-id: 7
      Status Description: Negotiation done
      Local spi: 5CE063D07E8745EA       Remote spi: 52D538043A8E330C
      Local id: hostname=Router1.cisco.com
      Remote id: hostname=Router2.cisco.com
      Local req msg id:  0              Remote req msg id:  2
      Local next msg id: 0              Remote next msg id: 2
      Local req queued:  0              Remote req queued:  2
      Local window:      5              Remote window:      5
      DPD configured for 0 seconds, retry 0
      Fragmentation not  configured.
      Extended Authentication not configured.
      NAT-T is not detected
      Cisco Trust Security SGT is disabled
      Initiator of SA : No
```

IKEv2 Cookie Challenge and Call Admission Control

The following scenario highlights the use of the cookie challenge and the maximum in negotiation SA features, and the benefits that each brings.

IKEv2 call admission control (CAC) limits the maximum number of IKEv2 SAs that can be established. CAC limits the number of simultaneous negotiations with the default being 40 in-negotiation SAs, although this value is configurable using the **crypto ikev2 limit max-in-negotation-sa** command.

To illustrate the CAC in action, the architecture in Figure 7-5 was developed. This setup consists of an IOS device acting as a VPN headend. Imagine a device created to send many IKE_SA_INIT requests to the headend from random spoofed source IP addresses. The IOS headend is configured with a default gateway, which is where all replies to any received IKE_SA_INIT messages will be sent and then discarded. The IKEv2 generator is pre-configured with an IKEv2 proposal that will be accepted by the IKEv2 headend and sends approximately 12 spoofed packets every second.

The IKEv2 generator sends an IKE_SA_INIT request with a spoofed source IP address of 192.168.1.1 to 10.10.10.1. The IKEv2 headend receives the IKE_SA_INIT, checks that the transforms are valid, allocates state and returns its IKE_SA_INIT response. This response will be received by the router and then forwarded to the 192.168.1.1 destination where it will be discarded.

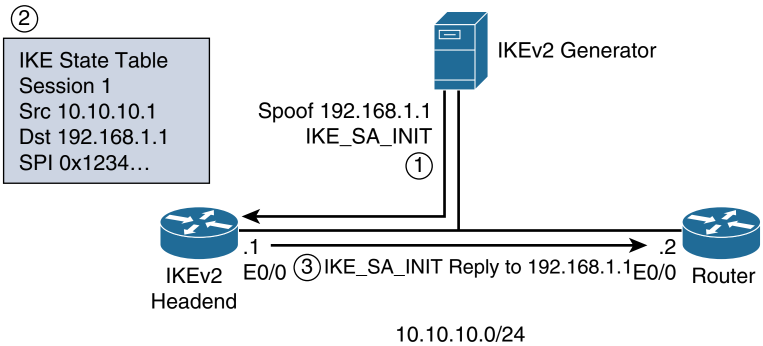


Figure 7-5 *CAC Architecture*

The hardware used for the IKEv2 headend was purposely chosen as a low-powered device. This was to illustrate the load when generating a large number Diffie-Hellman calculations and the software crypto engine was used. The following example illustrates the CPU history when a constant stream of spoofed IKEv2 SA_INIT requests is sent from the IKEv2 generator. The sudden initial spike in CPU (40 to 60 seconds) is due to the device processing the first forty spoofed IKE_SA_INIT requests, these are processed and replies sent. The CPU then drops to zero percent for approximately fifteen seconds and once again rises back to near full CPU at ninety percent. The drop in CPU processing was due to the CAC feature becoming active. Once forty IKE SAs are in negotiation, no more IKE_SA_INIT requests will be processed. Although the IKEv2 generator is sending a constant stream of these, the IKEv2 headend will only process forty at any given time (although this value is configurable). Some of the initial forty requests time out, and the state for these are removed before any new requests are processed and state allocated.

```
Router#show processes cpu history
```

```
100
 90          *****
 80          *****                         *****
 70          *****                         *****
 60          *****                         *****
 50          **********                    **********
 40          **********                    **********
 30          **********                    **********
 20          **********                    **********
 10          **********               ***************
   0....5....1....1....2....2....3....3....4....4....5....5....6
             0    5    0    5    0    5    0    5    0    5    0
             CPU% per second (last 60 seconds)
```

When an IKEv2 device acting as a responder receives a number of half-open IKE_SA_INIT requests, the cookie challenge mechanism can be deployed. This will enable the responder to include the cookie notification payload in the response to the initiator. The responder does not allocate any state to the session. If the initiator was legitimate, the response containing the cookie will reach the initiator who will then re-attempt the IKE_SA_INIT exchange, including the cookie notification payload, which is then verified by the responder. The responder will then allocate state to the IKE session.

If a device is under a Denial-of-Service (DoS) attack where spoofed IKE_SA_INIT are sent with the purpose of overloading the CPU, the device can be configured to activate the cookie-challenge mechanism. In this situation, the responder will reply with the cookie notification payload. Because this reply is sent to an IP address that was spoofed by an attacker, this reply will be discarded, or dropped by the receiver.

To illustrate this behavior, the IKEv2 headend was amended to allow 1000 in negotiation SAs. The following example shows the command used to achieve this.

```
Router(config)#crypto ikev2 limit max-in-negotation-sa 1000
```

The CPU of the IKEv2 headend was then constantly at 100 percent. This was due to the amount of constant spoofed IKE_SA_INIT requests from the IKEv2 generator that overwhelmed the IKEv2 state machine.

To rectify this issue, the cookie-challenge is enabled by default. This was enabled, using the value of 0, so all received IKE_SA_INIT requests will be returned with the cookie notification payload.

```
Router(config)#crypto ikev2 cookie-challenge 0
```

The value configured can be between 0 and 1000, which denotes the maximum number of in-negotiation IKE SAs before the cookie challenge is engaged.

No state is allocated to any IKE sessions as all IKE_SA_INIT replies are resent. The following example illustrates the impact that enabling the cookie challenge mechanism has. Once cookie challenge is enabled, the CPU drops from 100 to 0 percent. This is due to the fact that no state is allocated to any of the received IKE_SA_INIT requests.

```
Router#show processes cpu history

100      *******************************************************
 90      *******************************************************
 80      *******************************************************
 70      *******************************************************
 60      *******************************************************
 50      *******************************************************
 40      *******************************************************
 30      *******************************************************
 20      *******************************************************
 10      *******************************************************
   0....5....1....1....2....2....3....3....4....4....5....5....6
             0    5    0    5    0    5    0    5    0    5    0
             CPU% per second (last 60 seconds)
```

The cookie challenge is a useful feature when an IKEv2 headend is under a DoS attack whereby source IP addresses are spoofed. It can be enabled by default. However, this will incur an additional two-packet exchange to any IKE negotiation which might not be optimal in some situations. Using a value for the maximum in negotiation SAs that is a little higher than what is observed in a known good state will allow this mechanism to engage should a DoS condition occur.

Summary

The examples used in this chapter illustrate a variety of IKEv2 configurations. Numerous authentication methods were used to illustrate the broad range of options available and the benefits that they bring. Smart defaults were used to show the simplicity of the configuration when these are employed. PKI is mandatory when using RSA or EC digital signatures which isn't needed when using pre-shared-key authentication. However, this is not as scalable.

The use of the HTTP URL cert feature was described, where the certificate is not sent in the exchange but instead is retrieved by the IKEv2 peer. This allows for a substantially reduced packet size of the IKE_AUTH exchange.

The use of the maximum in-negotiation SAs and the cookie challenge mechanism was observed to illustrate how IKE can be susceptible to DoS attacks. The use of the cookie notification payload can reduce the impact of a DoS attack; however, in non-DoS conditions, it does add an additional round trip to any IKEv2 exchange.

Index

Symbols

A

B

C

D

E

F

G

H

I

K

N

Q

R

S

T

U

V

W

X-Y-Z

IKEv2 IPsec Virtual Private Networks

- Understand IKEv2 improvements: anti-DDoS cookies, configuration payloads, acknowledged responses, and more
- Implement modern secure VPNs with Cisco IOS and IOS-XE
- Plan and deploy IKEv2 in diverse real-world environments
- Configure IKEv2 proposals, policies, profiles, keyrings, and authorization
- Use advanced IKEv2 features, including SGT transportation and IKEv2 fragmentation
- Understand FlexVPN, its tunnel interface types, and IOS AAA infrastructure
- Implement FlexVPN Server with EAP authentication, pre-shared keys, and digital signatures
- Deploy, configure, and customize FlexVPN clients
- Configure, manage, and troubleshoot the FlexVPN Load Balancer
- Improve FlexVPN resiliency with dynamic tunnel source, backup peers, and backup tunnels
- Monitor IPsec VPNs with AAA, SNMP, and Syslog
- Troubleshoot connectivity, tunnel creation, authentication, authorization, data encapsulation, data encryption, and overlay routing
- Calculate IPsec overhead and fragmentation
- Plan your IKEv2 migration: hardware, VPN technologies, routing, restrictions, capacity, PKI, authentication, availability, and more

*This book is part of the **Networking Technology Series** from Cisco Press®, which offers networking professionals valuable information for constructing efficient networks, understanding new technologies, and building successful careers.*

Category: Networking Technology

Covers: Security

Create and manage highly-secure IPsec VPNs with IKEv2 and Cisco FlexVPN

The IKEv2 protocol significantly improves VPN security, and Cisco's FlexVPN offers a unified paradigm and command line interface for taking full advantage of it. Simple and modular, FlexVPN relies extensively on tunnel interfaces while maximizing compatibility with legacy VPNs. Now, two Cisco network security experts offer a complete, easy-to-understand, and practical introduction to IKEv2, modern IPsec VPNs, and FlexVPN.

The authors explain each key concept, and then guide you through all facets of FlexVPN planning, deployment, migration, configuration, administration, troubleshooting, and optimization. You'll discover how IKEv2 improves on IKEv1, master key IKEv2 features, and learn how to apply them with Cisco FlexVPN.

IKEv2 IPsec Virtual Private Networks offers practical design examples for many common scenarios, addressing IPv4 and IPv6, servers, clients, NAT, pre-shared keys, resiliency, overhead, and more. If you're a network engineer, architect, security specialist, or VPN administrator, you'll find all the knowledge you need to protect your organization with IKEv2 and FlexVPN.

Graham Bartlett, CCIE No. 26709, has designed large-scale VPNs throughout the U.K., and has collaborated with customers to deploy IKEv2 and Next Generation Encryption worldwide. Graham has discovered many IKEv2 zero-day vulnerabilities, contributed to IETF RFCs, and published intellectual property as prior art. A CiscoLive speaker, he has developed exam content for both CCIE and CCNP certifications. He holds CCP (Senior) IA Architect, CCP (Practitioner) Security & Information Risk Advisor, CCNP, CISSP, and Cisco Security Ninja credentials, and a BSc (Hons) in Computer Systems and Networks.

Amjad Inamdar, CISSP No. 460898, Senior Technical Leader with Cisco IOS Security Engineering in India, specializes in designing, developing, and deploying Cisco IOS secure connectivity solutions, including FlexVPN, DMVPN, GETVPN, and EzVPN. He is now working on Cisco's next generation SD-WAN solution. A contributor to IETF draft standards and regular presenter on security topics, Amjad holds a Cisco patent and has prior art publications. His credentials include CISSP, CCSK, CCNP Security, CCDP, CCNP R/S, CCNA (SP, Data Center, Wireless, and Voice) and Cisco Security Ninja; and a B.E. in Electronics and Communication Engineering.

ISBN-13: 978-1-58714-460-8
ISBN-10: 1-58714-460-3

56499

9 781587 144608

ciscopress.com

$64.99 USA / $80.99 CAN