IPsec

MONTAVISTA SOFTWARE, LLC.

Prem Karat

Aug-2016

Contents

ABSTRACT 3

IMPLEMENTATION OVERVIEW 4

Security Basics 4

IPsec Basics 5

IPsec Implementation 5

Essential Terminologies 6

IPsec Request Flow 8

Sample Configuration 9

IKE 9

strongSwan 10

Crypto Hardware Accelerators 11

APPENDICES 12

Abstract

This whitepaper provides an overview of IPsec implementation in MontaVista Carrier Grade Edition 7 (CGE 7). This paper is intented to users who have basic knowledge of IPsec and looking for what is and is not supported by CGE7.

Overview of the following is covered in this paper

- Security basics
- IPsec basics
- IPsec implementation - the kernel space, userspace and XFRM.
- IPsec protocols
- IPsec modes of operation
- Essential terminologies
- IPsec request flow
- IKEv1 & IKEv2
- strongSwan
- Crypto Hardware Accelerators

Implementation Overview

Security Basics

The most primary & commonly accepted aspects of security are **confidentiality, integrity, and authentication**. *Confidentiality* refers to protecting data from disclosure to unauthorized parties. *Integrity* refers to protecting data from being modified by unauthorized parties. *Authentication* refers to assurance to one party that the other authorized party is who that party claims to be.

Confidentiality is provided through *cryptography* (symmetric key & public key cryptography). Few implementation of symmetric and public key cryptographic techniques are

Symmetric Key Cryptography:

- DES
- 3DES
- AES

Public Key Cryptography:

- RSA
- Diffie-Hellman

In real world scenario, public key cryptography finds its use in authentication and during key exchange it encrypts the keys to ensure confidentiality.

Integrity is provided through various mechanisms like

- Message Digest (MD)
- Checksum
- Hash Algorithms (HA)
- CRC

Some examples of security protocols available across the OSI layers.

Communication Layers	Security Protocols
Application Layer	ssh, kerberos, PGP, WSS , S/MIME
Transport Layer	TLS, SSL
Network Layer	IPsec
Data Link Layer	L2TP, PPTP, IEEE 802.1X, WPA2
Physical Layer	Quantum Communications

Authentication is provided through various mechanisms and really depends on applications and context. But few examples are

- Digital Signatures
- HMAC
- LDAP
- EAP

IPsec Basics

IPsec is designed to protect the traffic at the IP level (IPv4 & IPv6) & all upper layer protocols by providing a multi-layer approach to secure IP packets with confidentiality, data integrity, authentication & protection against replays. Optional in IPv4, IPsec is mandatory for any implementation of IPv6.
IPsec protocols use standard symmetric key encryption algorithms for providing ***confidentiality***.

IPsec protocols use hash message authentication codes (HMAC) for ***data integrity & authentication***. To derive this HMAC the IPsec protocols use hash algorithms like MD5 and SHA to calculate a hash based on a secret key and the contents of the IP packet.

IPsec protocols use a sliding window to protect against ***denial of service*** attacks. Each packet gets assigned a sequence number and is only accepted if the packet's number is within the window or newer. Older packets are immediately discarded. This protects against replay attacks where the attacker records the original packets and replays them later.

IPsec Implementation

IPsec implementation has both userspace and kernelspace components. The kernel space part does the actual job of IPsec (encrypting, data integrity verification, authenticating etc..) . Userspace part helps in setting up IPsec between two network entities (what algorithms to use for key exchange, encryption, data integrity & authentication etc..). Setup is done either by manual configuration or by automation using a protocol called IKE. For bigger networks, setup is done using automation (IKE daemon tools).

Userspace Part:

Some userspace tools are

- ipsec-tools (includes setkey and racoon)
- strongSwan
- openSwan
- isakmpd from OpenBSD.

CGE7 supports both ipsec-tools and strongSwan.

setkey from ipsec-tools can be used for manual configuration/setup. setkey in combination with racoon can be used for automatic configuration/setup. In this case, setkey will be used for specifying security policies (SP) and racoon will be used in setting up security associations (SA). More on this is covered in later part.

strongSwan is the Internet Key Exchange(IKE) daemon tool. It implements both IKEv1 & IKEv2. Its a pure userland application that interoperates with a IPsec-enabled Linux kernel. It provides both Site-Site & Remote access VPN solutions.

Kernelspace Part:

Two IPsec kernel stacks are currently available:

- KLIPS
- NETKEY

The Linux kernel NETKEY code is a rewrite from scratch of the KAME IPsec code and is the default stack supported by CGE7. KLIPS is not supported in CGE7.

XFRM:

XFRM is the configuration and monitoring interface between the IPsec userspace part and IPsec kernel components.

IPsec is implemented using two protocols namely

- AH(Authentication Header) or
- ESP (Encapsulating Security Payload).

At the same time IPsec supports two modes of operation namely

- transport mode (or)
- tunnel mode.

Based on the requirement a user may choose one of the above protocol and the modes of operation while configuring IPsec.

Essential Terminologies

Authentication Header (AH) protocol:

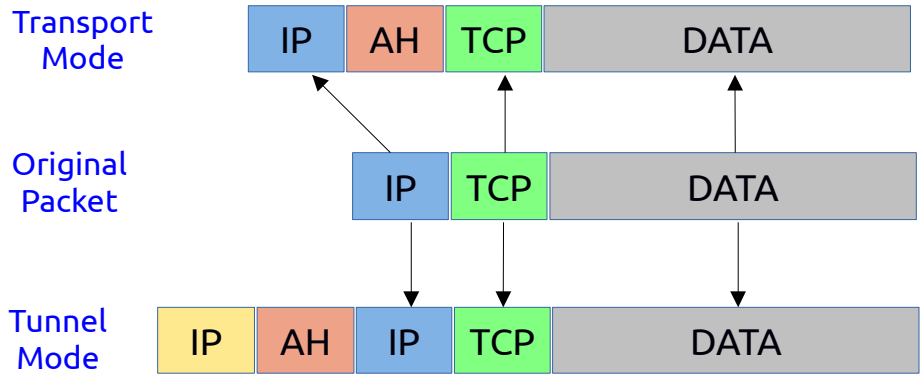
This is one of the two implementations of IPSec. AH provides source authentication & data integrity for IP datagrams. But it does not provide confidentiality/encryption.

ESP (Encapsulating Security Payload):

This is another widely used IPSec protocol. It provides source authentication, data integrity, and confidentiality/encryption.

Transport Mode:

In transport mode, only the payload/data of the IP packet is usually encrypted and/or authenticated. The routing is intact, since the IP header is neither modified nor encrypted.



Tunnel Mode:

In tunnel mode, the entire IP packet is encrypted and/or authenticated. It is then encapsulated into a new IP packet with a new IP header.

Security Policy (SP) & Security Policy Database (SPD):

Security policy is a rule which decides whether a given flow needs to go for IPSec processing or not. If no policy matches, the packet takes the default flow in the network stack. SPD is a kernel data structure *(struct xfrm_policy)*, that indicates "what" to do with arriving packet.

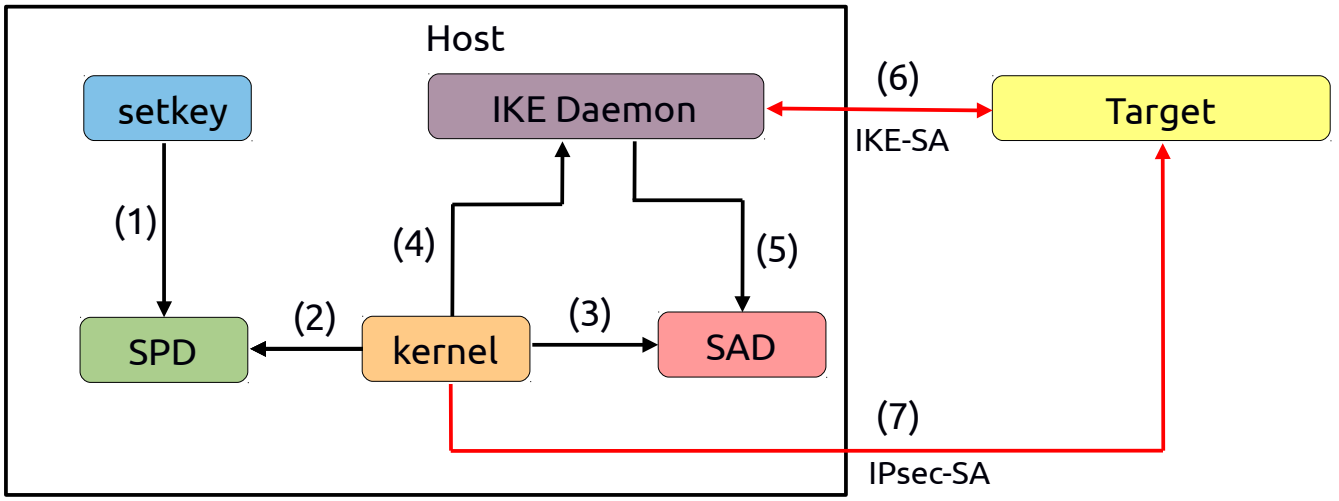
Security Association (SA) & Security Association Database (SAD):

A security association is the bundle of algorithms and parameters (such as keys) that is being used to encrypt and authenticate a particular flow in one direction. SAD is a kernel data structure, *(struct xfrm_state)* that indicates, "how" to apply IPsec to a packet.

Security Parameter Index (SPI):

SPI is an index to the security association database - SAD, along with the destination address in a packet header, which together uniquely identify a security association (SA) for that packet.

IPsec Request Flow



(1) The administrator sets a policy to SPD by using setkey (manual) or by using strongSwan ipsec.conf configuration file (automation).

(2) Kernel refers to SPD in order to make a decision of applying IPsec to a packet.

(3) If IPsec is required, then kernel gets the key for IPsec-SA from SAD.

(4) If it fails to get the key, then kernel sends a request to get the key to IKE daemon.

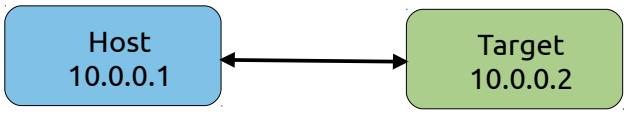
(5) IKE daemon establishes an IKE-SA and exchanges the key with target to establish IPsec-SA.

(6) IKE daemon puts the key into SAD.

(7) Kernel can send a packet to the target after applying IPsec.

Sample Configuration

Security protocol is ESP. Encapsulation mode is Tunnel.



A typical simple Security Policy (SP) looks like this:

```
#Security policy
        spdadd 10.0.0.1 10.0.0.2 any -P out ipsec esp/tunnel//require ;
```

This says 'traffic going from 10.0.0.1 to 10.0.0.2 should be encrypted using ESP protocol with encapsulation mode as tunnel. The SP can be added manually using setkey or if strongSwan is used, then can be specified using ipsec.conf configuration file.

A typical simple Security Association (SA) looks like this:

```
#Security Association
        add 10.0.0.1 10.0.0.2 esp 0x201 -m tunnel -E 3des-cbc "12345" -A
hmac-md5 "67890";
```

This says 'traffic going from 10.0.0.1 to 10.0.0.2 should be encrypted using 3des-cbc with key '12345' and authenticated using hmac-md5 using key '67890'. The SPI id is '0x201' . The SA can be added manually by using setkey or will be automatically populated by IKE daemon if either racoon or strongSwan is used.

Now if you ping 10.0.0.2 from 10.0.0.1, the *tcpdump* output should show something like this

```
        IP 10.0.0.1 > 10.0.0.2: ESP(spi=0x201,seq=0xa)
```

IKE

Internet Key Exchange (IKE) protocol is responsible in setting up security associations (SA) that allow two parties to send data securely. There are two versions, IKEv1 & IKEv2. IKEv2 is the second and latest version of the IKE protocol. As mentioned earlier, IKE is important in bigger networks for automating IPsec setup. RFC 4306 describes in great detail the advantages of IKEv2 over IKEv1, however it is important to note that the entire IKE exchange was overhauled.

IKEv1 operates in 2 phases, Phase 1 (main mode) requires 6 messages for IKE SA and Phase 2 (quick mode) requires 3 messages for IPsec SA. IKEv2, at best, requires only 4 messages to establish IKE SA and first IPsec SA (IKE_SA_INIT/IKE_AUTH). 2 messages for additional IPsec SA (CREATE_CHILD_SA). At worst, this can increase to many packets depending on complexity of authentication, number of EAP (Extensible Authentication Protocol) attributes as well as number of SA's.

IKEv2 supports EAP authentication. IKEv2 can use an AAA server to remotely authenticate mobile and PC users and assign private addresses to these users. IKEv1 does not support EAP authentication.

racoon (ipsec-tools) and strongSwan are the two IKE tools supported in CGE7. strongSwan supports both IKEv1 & IKEv2. racoon supports only IKEv1 and does not support IKEv2.

strongSwan

There are notable changes from strongSwan v5.0.0 and since CGE7 supports strongSwan v5.0.4 (as on writing this paper), MV users should note the following.

Pluto is no longer the deamon used for IKEv1. Charon is the keying deamon for IKEv1 & IKEv2.

AH protocol is “not supported” in Strongswan v5.0.4. The support was added in v5.1.1.

NAT-T (NAT traversal) is supported by default in both IKEv1 & IKEv2 without enabling it explicitly.

MOBIKE is supported by strongSwan IKEv2.

Dead Peer Detection (DPD) is supported by both IKEv1 and IKEv2 but the packet retransmission mechanism is implemented differently. For IKEv1 the *dpdtimeout* configuration parameter determines the timeout interval, whereas for IKEv2 the default retransmission timeouts apply. Refer to strongSwan documentation for more details.

IPComp (IP Payload Compression) is not supported on natted connections in strongSwan 5.0.4. The support is available from v5.1.0

ipsec SP policies can be listed using 'ip' command

```
ip -s xfrm policy
```

ipsec SA connections can be listed using 'ip' command

```
ip -s xfrm state
```

Crypto Hardware Accelerators

Cryptographic algorithms like DES, AES, SHA-256 are implmented in hardware using one of the following ways

- As a seperate processor integrated into the SoC
- As a coprocessor on the circuit board
- As a chip on an extension board, connected to the main board via BUS
- As extensions in CPU instruction set and thus integral part of CPU. e.g AES-NI

Kernel has access to the hardware accelartor through kernel driver. For example in IPsec, since the encryption, authentication etc.. happens in the kernel, IPsec can automatically make use of the hardware accelartor provided the Linux Scatterlist Crypto API kernel driver is implemented for the hardware accelerator.

The complexity lies in accessing the hardware crypto from userpace. There are many implmentations in accessing hardware crypto from userland. To mention a few,

- AF_ALG
- Cryptodev
- OCF

AF_ALG is default interface supported by linux kernel and is based on socket interface.

Cryptodev has out-of-tree support and is based on device interface /dev/crypto

OCF is a port of the OpenBSD Cryptographic Framework to Linux that also based on device interface /dev/crypto

strongSwan has AF_ALG plugin that can help in accelerating IKE packets by making use of hardware accelerators but it may not be important due to less volume of IKE packets to encrypt.

Refer https://wiki.strongswan.org/projects/strongswan/wiki/Pluginlist for full plugin list support in strongSwan.

Appendices

[1] http://lartc.org/howto/index.html

[2] http://www.ipsec-howto.org

[3] http://www.linuxjournal.com/article/9916?page=0,0

[4] http://www.kame.net/newsletter/20001119/

[5] https://www.strongswan.org/docs/LinuxTag2008-strongSwan.pdf

[6] http://www.unixwiz.net/techtips/iguide-ipsec.html

[7]http://security.hsr.ch/lectures/Information_Security_2/Vorlesungsunterlagen/04-VPN_Notes.pdf

[8] http://www.estoile.com/links/ipsec.htm

[9] https://tools.ietf.org/html/rfc4301

[10] https://tools.ietf.org/html/rfc4306

[11] https://tools.ietf.org/html/rfc4555

[12] http://cryptodev-linux.org/

[13] https://wiki.strongswan.org/projects/strongswan/wiki

[14]https://events.linuxfoundation.org/sites/events/files/slides/lcj-2014-crypto-user.pdf


MontaVista