USGv6 IPsec Implementation Guide

Version 1.1

Informational Document

http://www.iol.unh.edu

Modification Record

- May 7, 2012
  - Version 1.1 – Minor edits and additions
- July 16, 2010
  - Version 1.0 – First release

Acknowledgments

The University of New Hampshire would like to acknowledge the efforts of the following individuals in the development of this document.

Timothy Carlin	University of New Hampshire
Thomas Peterson	University of New Hampshire
Timothy Winters	University of New Hampshire

Table of Contents

Introduction

This document is intended to help those who are implementing IPsec in order to meet the USGv6 testing requirements. There are several brief goals of this document:

- Discuss necessary User Interfaces required for testing
- Outline testing methods and test topologies
- Highlight common failures

While the primary motive for this document is adhering to the USGv6 Testing Requirements, the features outlined should also meet many of the requirements of other testing programs, including IPv6Ready.

Definitions

- IPsec/IPsecv3 – Any one of a number of protocols, RFCs, and Technologies, or all of them together, or both. Also, a specific architecture defined by RFC 4301. When referenced by this document, IPsec refers only to the IPsecv3 test selection table requirements. (See next section)
- IKEv2 – Internet Key Exchange Protocol, version 2. This protocol has seen several edits, most recently via RFC 5996, which, while not incrementing the version, obsoleted the previous document, RFC 4306, and RFC 4718.

User Interface Requirements

IKEv2

There are a number of issues to consider for IKEv2 Testing. These issues are broken down in detail below.

Device Type

An IKEv2 implementation can act as either an Initiator, or a Responder during a negotiation. The majority of devices, both End-Nodes and Gateways, may operate in either position depending on the situational needs. There are certain End-Node devices, known as Passive Nodes, which may never initiate any traffic, and instead, only transmit traffic in response. A classic example would be a simple printer. For this category of devices, the associated Initiator test cases are omitted (Labeled as IKEv2.EN.I.*). Security Gateways, by definition, may act in either position, and both sections must be tested.

For End-Nodes, which are not passive, there must be an interface through which to cause the device to initiate an IKEv2 negotiation. There are at least two ways of accomplishing this. The first method is to have a function which by itself begins the negotiation. Often times this is the most straightforward method. A second, more common, method simply enables IKEv2 in a ``waiting'' state. Following this, egress traffic matching the configured selectors causes the device to initiate IKEv2. The testing is accomplished using ICMPv6, therefore the most common application for generating matching traffic is the common Ping (Echo Request/Reply). In the absence of a ping-like program, another application should be provided. Often times an application using TCP may be used. In this case, the port need not be open on the destination, as the returning TCP RST should also be protected.

Authentication Method

Two methods of authentication are required: Pre-Shared Keys (PSK) and RSA Certificate Signature Authentication.

The User Interface must allow for both ASCII formatted, as well as Hexadecimal PSKs.

Certificates pose a challenge for configuration, as there are many approaches to take during implementation. For simplicity, an offline Test Certificate Authority is created, and the certificates used for testing are signed using this Test CA.

Therefore, a method to generate a Certificate Signing Request is useful and a method to upload a Certificate is required.

Another useful feature is to allow the user to upload both a certificate, and the corresponding private key. This allows all certificate generation to be done offline, without device interaction, reducing testing time.

In addition, there is no requirement that both IKEv2 Initiator and Responder use the same method in authenticating the peer. Therefore, the capability to specify the ``Local'' as well as the ``Remote'' authentication method is required. Lastly, support for several ID Types is required, namely: 1) ID_IPv6_Address, 2) FQDN, and 3) RFC822 (email address). A device must be able to configure the reception, and transmission of these types for maximum interoperability.

Peer Address

The identity of the Peer, as well as the algorithms and traffic selectors to be used with this peer should be configurable.

Transport/Tunnel Mode

Both Transport and Tunnel Modes are required configuration options. In particular, devices should be able to Interoperate with both End-Nodes as well as Security Gateways.

Traffic Selectors

The Traffic Selectors define which classes of traffic are to be protected by the Security Association being negotiated. Said another way, Security Associations will be negotiated according to the Traffic Policies specified.

Both the Local, as well as Remote Traffic selector should be configurable. Another way of viewing this is as the Outgoing and Incoming traffic, respectively. The Remote Traffic Selector is the traffic to be protected by the peer device on egress, and the Local Traffic Selector is the traffic to be protected by the device under test on egress.

Specifying Traffic Selectors as “Incoming” and “Outgoing”, is a more intuitive way of understanding the directionality, however some implementations may choose to identify them as “Local” and “Remote”. This is largely a matter of circumstance.

There are a number of options per Traffic Selector, all of which must be configurable. They are:

- Protocol ID – For example, ICMP, TCP, UDP, etc.
- Port Range – Two 16 Bit fields. Note that this may be implemented independently from Protocol ID. In other words, there is no requirement to check that the value here ``makes sense'' for the protocol ID.
    - When the Protocol Selected is ICMPv6, it may be useful to allow the user to specify Type and Code, rather than Port, and to do the appropriate changes to formatting in code.
- Address Range – Two IPv6 Addresses. Note that they may be the same address, or a range. e.g. 3000::/64 translates to 3000:: through 3000::FFFF:FFFF:FFFF:FFFF.

IKE SA

There are 4 types of algorithms that must be configurable for the IKE SA. They are Encryption, Integrity Check, Pseudo-Random Function, and Diffie-Hellman Group. They need not be configured as independent proposals. Some tests do require that only a bare minimum of algorithms be enabled; therefore an ``all algorithms all the time'' approach is invalid.

Often times the Integrity and PRF algorithms are configured via the same option. While this is not a requirement, it may be more straightforward, and in fact, more secure. Either method is acceptable.

IPsec (Child) SA

The Encryption and Integrity Check algorithm must be configurable for the IPsec SA (historically known as the Phase-2, or QuickMode SA). In addition, there may be a configuration option to enable or disable Extended Sequence Numbers.

Lifetimes

While not required, the ability to configure both IKE SA, and Child SA lifetimes is a useful feature for testing. The user should be able to specify either a length of time and an amount of data, with either value being infinite, though not both at the same time.

Deletion

While not required, the ability to remove all existing SAs or IKEv2 state is extremely useful in reducing testing time.

Forceful Deletion

IKEv2 uses an Informational Exchange to gracefully attempt deletion of IKE SA and IPsec SA state. While this is required to conform to the standard, a method to forcefully delete all state for testing is desirable. This usually manifests itself as a command, or UI button, which causes the IKEv2 process to "forget" all state information.

Logging

While not required, access to a detailed log or debugging facility is extremely useful in reducing testing time. Minor differences between implementations often require slight modifications to the test tool; access to a debug facility greatly aids the process of identifying issues, which may or may not be protocol failures.

IPsecv3

The focus of the IPsecv3 testing is the architecture defined by RFC 4301. IPsecv3 testing may be accomplished using either Manually configured Keys, or through IKEv2 Automated keying. The ability to support both Manual keys and Automated keys is required.

Simultaneous Instances

An IPsec device must be able to protect communications with at lease 2 peers simultaneously. The two peer configurations should be independent, allowing for different algorithms, keys, and negotiation models (Automated/Manual).

ICMPv6 Selectors

IPsecv3 devices must be able to select traffic based on ICMPv6 Type and Code. There should be a facility to configure different Incoming and Outgoing SAs based on ICMPv6 type. For example:

SA1-Incoming for Echo Request with 3DES/SHA1
SA2-Incoming for Echo Reply with AES/SHA1
SA3-Outgoing for Echo Request with 3DES/SHA256
SA4-Outgoing for Echo Reply with AES/AESXCBC

The use of different algorithms in the above example is for demonstration purposes only. It is used to illustrate the fact that outgoing traffic may use different SAs, depending solely on the protocol and type, NOT on destination! For instance, an

Echo Request to Destination1 may use 3DES/SHA1, but a device should allow the user to specify that an Echo Reply to Destination1 uses an entirely different encryption scheme!

Manual Keys

A device must have a method for configuring manual keys. In addition, the device should allow the keys to be entered in either hexadecimal or ASCII mode! Also, the most straightforward method to specify keys and SPI values is to label them according to direction, incoming or outgoing. Local and Remote are often ambiguous to many users.

Bypass/Discard

Based on the configured selectors, a device must be able bypass, discard, or protect traffic. In addition, the default policy for unmatched traffic should be configurable to allow for it to be either bypassed or discarded.

Support for this feature may manifest itself inside a simple firewall, rather than as a separate IPsec Policy. This is an acceptable implementation strategy.

Tunnel Mode

Tunnel mode is a requirement of IPsecv3 devices. The device should be interoperable with both End-Nodes and Security Gateways in Tunnel Mode, which is to say it should support tunneling of a remote network, as well as remote host.

In other words, the Inner and Outer IP headers of tunneled packet will be different in the case of End-Node vs. Gateway, and identical in the case of End-Node vs. End-Node. Note that many times, an implementer is tempted to "force" the user to use different addresses for Inner and Outer IP, despite the topology being End-Node vs. End-Node, where no such addressing scheme is required. This often leads to confusion to the user, and complexity in the configuration. The best way to support this is to abstract any use of sub-interfaces from the users, and to merely ask that they configure "Tunnel Endpoints" and "Data Endpoints", which can be identical!

In addition, it should be able to support selection for two nodes behind a single gateway. This means allowing the user to specify distinct algorithms for different sets of traffic behind a single SGW.

ESP

ESP is a subset of IKEv2 and IPsec testing. If all required algorithms are supported, there are no other functions tested by this section, not already tested in other areas.

Fragmentation

Fragmentation in IPsec and IKEv2 is made complicated through the addition of the ESP header, and possibly an extra IP header in tunnel mode.

The best way to handle Path MTU is to treat IPsec traffic no differently than any other IPv6 traffic. A common error is to discount the added ESP error for Path MTU calculations, resulting in an overestimate. Another common error is to place the Fragmentation Header in the wrong location in the frame (it should occur Prior to ESP!).

Tunnel mode raises additional complexity. Errors here are usually caused by not treating ICMPv6 Packet Too Big messages in the correct context, that is, does it apply to the Data Channel (the traffic being protected) or the Control Channel (traffic between endpoints). Once the context is understood, the correct fragmentation scheme is apparent.

Algorithms

The following algorithms must be supported:

Encryption:
- 3DES
- AES (128-Bit)

Authentication:
- HMAC-SHA1
- AESXCBC-MAC (SHOULD+)

Diffie-Hellman:
- Group 2
- Group 24

Summary

This section concisely summarizes the requirements put forth in the above sections. This is by no means exhaustive, and is rather to point out commonly overlooked requirements. For an extensive list of requirements, the respective RFC and NIST USGv6 Profile should be consulted.

IKEv2

- DH-24
- PSK in ASCII and Hexadecimal format
- RSA Certificate Authentication
- Tunnel Mode with Remote Network
- ICMPv6/UDP/TCP/etc. Protocol and Type Selectors
- Simultaneous independent instances

IPsec

- Transport Mode
- Tunnel Mode vs. End-Node
- Tunnel Mode vs. Gateway (Remote Network)
- Independent incoming and outgoing selectors and SA DB
- ICMPv6/UDP/TCP/etc. Protocol and Type Selectors
- Bypass and Discard functions for Security Policy Entry
- Simultaneous independent instances

Algorithms

- 3DES-CBC
- AES-CBC (128 Bit)
- HMAC-SHA1
- AES-XCBCMAC (Should +)
- DH-2
- DH-24
- NULL Encryption

For More Information

Contact:

Timothy Carlin (tjcarlin@iol.unh.edu)
Thomas Peterson (thomasp@iol.unh.edu)
Timothy Winters (twinters@iol.unh.edu)

References:

NIST USGv6 Program
IKEv2 [RFC 5996]
IPsec [RFC 4301]
ESP [RFC 4303]

Test Selection Tables

The test selection tables are documents defined by NIST that call out specifically the tests and test suites, which must be executed in order to meet USGv6 Requirements.

IKEv2 Tables

IPsecv3 Tables

ESP Tables