DJIGZO Email Encryption Gateway

Integration Guide



THE DATA PROTECTION COMPANY Part Number: 007-012080-001 (Rev A, 06/2012)

© 2012 SafeNet, Inc. All rights reserved

Preface

All intellectual property is protected by copyright. No part of this document may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, chemical, photocopy, recording or otherwise without the prior written permission of SafeNet.

SafeNet makes no representations or warranties with respect to the contents of this document and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Furthermore, SafeNet reserves the right to revise this publication and to make changes from time to time in the content hereof without the obligation upon SafeNet to notify any person or organization of any such revisions or changes.

SafeNet invites constructive comments on the contents of this document. These comments, together with your personal and/or company details, should be sent to the address below.

4690 Millennium Drive Belcamp, Maryland 21017, USA

Limitations

This document does not include the steps to set up the third-party software. The steps given in this document must be modified accordingly. Refer to Luna SA / Luna PCI documentation for general Luna setup procedures.

Disclaimers

The foregoing integration was performed and tested only with the specific versions of equipment and software and only in the configuration indicated. If your setup matches exactly, you should expect no trouble, and Customer Support can assist with any missteps. If your setup differs, then the foregoing is merely a template and you will need to adjust the instructions to fit your situation. Customer Support will attempt to assist, but cannot guarantee success in setups that we have not tested.

Technical Support

If you encounter a problem while installing, registering or operating this product, please make sure that you have read the documentation. If you cannot resolve the issue, please contact your supplier or SafeNet support.

SafeNet support operates 24 hours a day, 7 days a week. Your level of access to this service is governed by the support plan arrangements made between SafeNet and your organization. Please consult this support plan for further information about your entitlements, including the hours when telephone support is available to you.

Technical Support Contact Information:

Phone: 800-545-6608, 410-931-7520 Email: safenet-inc.com

Table Of Contents

Chapter 1 Introduction	5
Scope	5
Prerequisites	5
Chapter 2 Integration with Luna SA/Luna PCI	7
Red Hat Enterprise Linux 6.2 /5.5 (32-bit)	7
Integrating DJIGZO Email Encryption Gateway 2.4 with Luna SA/Luna PCI	7
Requirements	7
Installation of DJIGZO HSM module	7
PKCS#11 Configuration	7
Using Sun PKCS#11 Provider	7
Change PIN	8
Create new CA	9
Create End-User Certificate	10
Sign & Encrypt E-mail	11
Decrypt E-mail	12

Luna SA/Luna PCI and DJIGZO Email Encryption Gateway Integration Guide

Chapter 1 Introduction

This document covers the necessary information to install, configure and integrate DJIGZO Email Encryption Gateway with SafeNet Luna SA & Luna PCI Hardware Security Module (HSM).

DJIGZO Email Encryption Gateway is a standard based centrally managed email server (MTA) that encrypts and decrypts your incoming and outgoing email at the gateway level. DJIGZO Email Encryption Gateway is compatible with any existing email infrastructure like Microsoft Exchange and Lotus Notes and has support for S/MIME and PDF encryption.

Scope

3rd Party Application Details

DJIGZO Email Encryption Gateway v2.4

Supported Platforms

The following platforms are supported for Luna SA v5.1

• Red Hat Enterprise Linux Server release 6.2 x86

The following platforms are supported for Luna PCI v5.0

Red Hat Enterprise Linux Server release 5.5 x86

HSMs and Firmware Version

K6 HSM f/w 6.2.1

Library and Driver Support

PKCS#11 v2.01 dynamic library

Distributions

- Luna SA Client s/w v5.1 (32-bit)
- Luna PCI Client s/w v5.0 (32-bit)

Prerequisites

Luna SA/Luna PCI Setup

Please refer to the Luna SA / Luna PCI documentation for installation steps and details regarding to configure and setup the box on system. Before you get started ensure the following:

- Luna SA appliance a secure admin password
- Luna SA a hostname, suitable for your network
- Luna SA network parameters are set to work with your network
- Initialized the HSM on the Luna SA appliance.
- Created and exchanged certificates between the Luna SA and your "Client" system.
- Created a partition on the HSM, remember the partition password that will be later used by Email Encryption Gateway. Register the Client with the partition. And run the "vtl verify" command on the client system to display a partition from Luna SA. The general form of command is "/usr/lunasa/bin/vtl verify" for Linux.
- Enabled Partition "Activation" and "Auto Activation" (Partition policy settings 22 and 23 (applies to Luna SA with Trusted Path Authentication [which is FIPS 140-2 level 3] only).

DJIGZO Email Encryption Gateway Setup:

DJIGZO Email Encryption Gateway V2.4 must be installed on the target machine to carry on with the integration process.

For a detailed installation procedure, please refer Dijgzo installation guide.

Luna SA/Luna PCI and DJIGZO Email Encryption Gateway Integration Guide

Chapter 2 Integration with Luna SA/Luna PCI

Red Hat Enterprise Linux 6.2 /5.5 (32-bit)

Integrating DJIGZO Email Encryption Gateway 2.4 with Luna SA/Luna PCI

This chapter explains how to configure the DJIGZO Email Encryption Gateway for Luna SA & Luna PCI support. It is assumed that a DJIGZO gateway is already installed in the directory /usr/local/djigzo and that the gateway is fully functional.

Requirements

- A functional DJIGZO gateway (version _ 2.4.0)
- Luna SA/ Luna PCI
- DJIGZO HSM module

Installation of DJIGZO HSM module

The DJIGZO HSM module will be installed in the directory /opt/djigzo-hsm.

Note: it is assumed that the user is logged-in with a non-root account and that sudo is used for commands that require root access.

\$ sudo mkdir /opt/djigzo-hsm

Untar the DJIGZO HSM module to /opt/djigzo-hsm:

\$ sudo tar xzf djigzo-hsm_*.tar.gz --directory /opt/djigzo-hsm

PKCS#11 Configuration

This section explains how to configure a PKCS#11 provider for DJIGZO. Java uses PKCS#11 to communicate with the HSM. A Java PKCS#11 provider is therefore required.

A soft-link to the PKCS#11 configuration files should be added to the DJIGZO configuration directory:

\$ cd /usr/local/djigzo/conf/

\$ sudo In -s /opt/djigzo-hsm/conf/hsm/

The default PKCS#11 provider for Java is provided by Sun (now Oracle) and comes installed with the Java runtime.

Using Sun PKCS#11 Provider

This section explains how to configure the HSM module to be used with the Sun PKCS#11 provider.

DJIGZO should be configured to store the private keys on the HSM:

\$ cd /usr/local/djigzo/conf/spring/spring.d

\$ sudo In -s /opt/djigzo-hsm/conf/spring/hsm.xml

To load the Sun PKCS#11 jar file at startup, a soft-link to the Sun PKCS#11 library and a soft-link to the HSM module should be added to the lib.d directory:

\$ cd /usr/local/djigzo/lib/lib.d

\$ sudo In -s /opt/djigzo-hsm/lib/djigzo-hsm.jar

\$ sudo In -s /usr/lib/jvm/jre-openjdk/lib/ext/sunpkcs11.jar

Modify "pkcs11-config.properties.safenet" file located at /opt/djigzo-hsm/conf/hsm to include below text:

name=SafenetLuna

```
# the path to the Safenet PKCS11 module
```

library=/usr/lunasa/lib/libCryptoki2.so # For Luna PCI it will be =/usr/lunapci/lib/libCryptoki2.so

description=Safenet PKCS11 provider

slot=1

attributes(generate,*,*) = {

CKA_TOKEN = true

}

```
attributes(generate,CKO_PUBLIC_KEY,*) = {
```

CKA_ENCRYPT = true

CKA_VERIFY = true

CKA_WRAP = true

}

```
attributes(generate,CKO_PRIVATE_KEY,*) = {
```

CKA_EXTRACTABLE = false

CKA_DECRYPT = true

CKA_SIGN = true

CKA_UNWRAP = true

```
}
```

A soft-link to the SafeNet specific properties file should be created:

\$ cd /opt/djigzo-hsm/conf/hsm

\$ sudo In -s pkcs11-config.properties.safenet pkcs11-config.properties

Change PIN

The PIN for authenticating to the SafeNet LunaSA/Luna PCI is stored in the file /opt/djigzo-hsm/conf/spring/hsm.xml (the default PIN is set to "123") and should be set to the PARTITION PASSWORD for Luna SA/Luna PCI.

<!-- for logging into the PKCS11 device -->

<bean id="pkcs11PasswordCallbackHandler"

class="mitm.common.security.password.StaticPasswordCallbackHandler">

<!-- the PKCS11 password *** CHANGE THIS *** -->

<constructor-arg value="123"/>

</bean>

DJIGZO should be restarted for the changes to take effect:

\$ sudo /etc/init.d/djigzo restart

DJIGZO should now be ready to use the Luna SA/ Luna PCI. Check the log file (/var/log/djigzo.log) for any problems.

The administrator can login to the administration page by opening the following URL in a browser:

https://<IP Address>:8443/djigzo

The login page should appear. After logging in with the correct credentials, the user's page will be opened.

Gateway login Please enter your user	🔗 djig <u>z</u> o	
Name Your user name	admin	
Password required	••••	
	Login	

Login credentials: Use the following default credentials:

username: admin

password: admin

Create new CA

Before starting to create end-user certificates, a root and intermediate certificate should be created. The Create new CA page can be used to create a new CA (i.e., create a new intermediate and root certificate).

Certificates	Roots	CRLS	CA	SMS	Settings	Queues	Logs	Ad
Create	new CA							
Root certifi	cate							
	Validity in days	1825						
	Key length in bits	2048	~					
	Emai							
Con	nmon name required							
nore Intermediat	e certificat	9						
	Validity in days	1825						
	Key length in bits	2048	~					
	Emai							
Con	nmon name required							
General								
Make	e default CA							
Signatu for certific	re algorithm ate signature	Sha25	6 With I	Rsa 🗸				
Create	Close							

Check with the Luna SA / Luna PCI tools to make sure that the private keys are actually stored on the HSM

Create End-User Certificate

After CA is created click on CA tab and create end-user certificate for test@example.com. Enter validity, key length, Signature algorithm, email as test@example.com and common name. Then click on button "Request". It will create an end-user certificate.

DJIGZO	
Users Domains	Certificates Roots CRLS CA DLP SMS Settings Queues Logs Admin About
ogout dd user	Create new end-user certificate
CA settings Request handlers Create new CA Select default CA	create CRL send certificates bulk request pending requests General
	validity in days
	in bits
	Signature algorithm Sha1 With Rsa 🔽
	Certificate subject
	Email test@example.com
	Common name persona non-validated
	email delivery
	Send by email send key file to user
	Password a sword b swo
	SMS password a SMS
	Store password store the pfx password in the user preferences
	Advanced
	show advanced settings

You can view end-user certificate created in previous step from "Certificate" tab:

🙆 DJIGZO		🚹 🔹 🔝 👘 🖃 🌐 💌 Page + Safety + To
Users Domains	Certificates Roots CRLS CA DLP SMS Settings Queues Logs Admin About	
Logout Add user Import certificates	Intermediate and user certificates	
Import keys	S Filter	
Trust List	delete selected download certificates download keys invert selection 25 🕑	
	Email Subject Expired Not Before Not After Key Usage E	xtended Key Usage Issuer
	🗌 test@example.com 🖗 EMAILADDRESS=test@example.com, CN=Test No Jun 14, 2012 Jun 14, 2017 keyEncipherment, digitalSignature e	mailProtection, clientAuth EMAILADDRESS=dgijzo@loc
		Valid 🗌 Invalid 📃 Revo
	DJIGZO	

Sign & Encrypt E-mail

For test purpose here we are using E-mail client tool that comes with djigzo.

1. Create a "raw" email file with name test.eml containing the following:

From: test@example.com Subject: test

SOME TEXT

Note: the "From" should be on the first line

2. Goto the directory where djigzo is installed i.e. /user/local/djigzo

Run following command from commad prompt:

[user@localhost djigzo]# java -cp djigzo.jar mitm.common.tools.SendMail -r test@example.com -h 127.0.0.1 -p 25 -in /<PATH TO TEST EMAIL FILE>/test.eml

the test.eml should now be delivered to test@example.com

There should now be an encrypted email in the MTA queue.

🙆 DJIGZO													â • 🖻
Users I	Domains	Certificates	Roots	CRLS	CA	olp sms	Settings	Queues	Logs	Admin	About		
Logout Add user		Mail transfer agent Queue											
0		MTA MPA outgoing MPA error MPA spool MPA respool DLP quarantine											
		Filter											
		delete selected hold selected release selected requeue selected flush inver								election	25 💌		
		Queue	ID status	s size Arri	val Time	Sender	Rec	ipients	Failure				
		X 🗌 C384764	46E6 Active	743 Tue	Jun 19 03:	33:06 test@exa	ample.com tes	@example.co	m				
1								DJIG	zo				

NOTE: It's best to use example.com domain since that is allowed for tests (see rfc2606) and it will never result in email being sent.

Now open the MTA queue page, view the email contents by clicking on the Queue ID link of the email. This will show the "raw" Postfix mail contents as shown in screen shot below:

Domanis	
t ser	MTA email message with queue ID: EE849646E6
	for <test@example.com>; Tue, 19 Jun 2012 03:44:08 -0400 (EDT)</test@example.com>
	Received: from localhost ([127.0.0.1])
	by iccallest.localdomain with Ship iD 646
	Tue, 19 Jun 2012 03:44:06 -0400 (EDT)
	Received: from localhost (localhost [127.0.0.1])
	by host.example.com (Postfix) with ESMTP id 136C6646E6
	for <test@example.com>; Tue, 19 Jun 2012 03:44:06 -0400 (EDT)</test@example.com>
	Date: Tue, 19 Jun 2012 03:44:05 -0400 (EDI)
	To: udisclosed-recipients::
	Message-ID: <20120619074406.136C6646E6@host.example.com>
	Subject: test
	MIME-Version: 1.0
	Content-Type: application/pkcs7-mime; name="smime.p7m"; smime-type=enveloped-data
	Content-l'ansier-indoding: paseo: Content-l'ansieitor: strachmert: filename="smime n]w!
	Content-Description: S/MIME Encrypted Message
	MIRGCSqGSIbSDQEHA6CAMIACAQAxggfqMIBZqIBADBOMDoxEDAOBgNVBAMMB010VCBDQSAxJJAk
	BgkqhkiG9w0BCQEMF21zaGFybWFAc2FmZW51dC1pbmMuY29tAhABOAOyfefOsKNWwnpS8Te/MA0G
	CSqGSIb3DgEBAQUABIIBAGzjpa00/jQuIIGNsFvCf6QQQXjtioGUBeai+h/Vqe8x5C/hatlDtyO1
	pitoruvatuoversoominutaatosrimatay naadimatay kuitobatotai yalayoo kuutoka kuutoka kuutoka kuutoka kuutoka kuuto 11 YVanki si Minufimadi ai siitu YHSynyyyy anixaa kuutoka kuutoka kuutoka kuutoka kuutoka kuutoka kuutoka kuuto
	abintznyD4X0gdf2zGFbgF11+AW3MexR+g21RUZIXowbFaFY1n9SaC1YKO8hw7dWWb3Zr1BGnWN1
	W0/YjCHUG5GoB5Fat/4q4Y03MDn+gmClh2wn1iuThCQkQKHmfHiQlLQ/dzkwgAYJKoZlhvcNAQcB
	MBQGCCqGSIb3DQMHBAgpSDVBUHA5PKCABIID6CuajXyac3rDaj05XM8inN+L1aoA6qkasdlibV0j
	6s/ce3Tdg79KmnRqlHI4iAfBreNr8A5KyF8a0iw9q7CI1anaxtYRScLdi+u41EYy+66P4w6gEAFv
	0DD3mcWuIMDGXx71E3Ah90B54L1K0/sQ00965X1DFeaUhcr4RQ12qq25+rqM51YA+K0Fjr0jEF1c 0AD2///A200/mc1E3_A/M-n=00/A2/WuIMC/A200/DFAUhcr4RQ12qq25+rqM51YA+K0Fjr0jEF1c
	DateMbdRJarPabadZJisoCFedeSUXXBoxDFDivtChicCOoi4mbdrZD0LJiUTJUJJTN
	FBUSSUA/v+uDFIwFtGu390BK6HopvcEvVc9YppfJ7Y1Z5kJscdctzHLoMHcTOLRkC1U6watOuFPD
	1pwE0eSKNEof1xf60Ur4Kd0M6bSg1+0apLuaJpvu1z+StKJIvnfWIE0bJKs/3Bb7kg26Rf8vYZ1T

Decrypt E-mail

- 1. Open the MTA queue page
- 2. View the email contents by clicking on the Queue ID link of the email. This will show the "raw" Postfix mail contents
- 3. Copy the exact contents between

*** MESSAGE CONTENTS deferred/??/????? ***

and

*** HEADER EXTRACTED deferred/??/????? ***

to the Encrypted.eml file (or to some other .eml file)

- 4. Add a domain example.com to the Domains page of djigzo
- 5. Set the Locality of the domain to "Internal" (if the domain is internal, email will be decrypted for users of that domain, if external email will be encrypted)
- 6. Now send the encrypted message from step 3 to the gateway using below command:

[user@localhost djigzo]# java -cp djigzo.jar mitm.common.tools.SendMail -r test@example.com -h 127.0.0.1 -p 25 -in /<PATH TO TEST EMAIL FILE>/Encrypted.emI

Because the example.com domain is now an "internal" domain, the gateway should try to decrypt it. After decryption there should be a digitally signed email in the MTA queue.

View the email contents by clicking on the Queue ID link of the email which is sent in above step, you can see the decrypted text as shown highlighted text in screenshot below:

🙆 DJIGZO		🏠 🔹 🔂 👘 📼 🖶 🕈 Page 🕶
	for <test@example.com>;</test@example.com>	
	Fri, 15 Jun 2012 07:30:22 -0400 (EDT)	
	Received: from localhost (localhost [127.0.0.1])	
	by host.example.com (Postfix) with ESMTP id 5BBE3646EC	
	for <test@example.com>; Fri, 15 Jun 2012 07:30:22 -0400 (EDT)</test@example.com>	
	Content-Type: multipart/signed; protocol="application/pkcs7-signature"; micalg=sha-1;	
	boundary="= Part_12_12522438.1339759823610"	
	Date: Tue, 19 Jun 2012 05:30:42 -0400 (EDT)	
	From: test@example.com	
	To: undisclosed-recipients:;	
	Message-ID: <20120615113022.5BBE3646EC@host.example.com>	
	Subject: test	
	MIME-Version: 1.0	
	X-Djigzo-Info-Encryption-Algorithm-0: 3DES, Key size: 168	
	X-Djigzo-Info-Encryption-Recipient-0-0: EMAILADDRESS=dgijzo@localhost.com,	
	CN=Int CA/137EFA8C9A542BF698A89DF3250447E//1.2.840.113549.1.1.1	
	X-Djigzo-Info-Signer-ID-0-1: EMAILADDRESS=dgijzo@localhost.com, CN=Int	
	CA/137EFA8C9A542BF698A89DF3250447E/	
	X-Djigzo-Info-Signer-Verified-0-1: True	
	X-Djigzo-Info-Signer-Trusted-0-1: True	
	=_Part_12_12522438.1339759823610	
	lesting bjigzo Encryption Decryption	
	Part 12 12522438.1339/59823610	
	Content-Type: application/pkcs/-signature; name=smime.p/s; smime-type=signed-data	
	Content-Fransfer-Encoding: Daseos	
	Content-Disposition: attachment; filename-"smime.p/s"	
	Content-Description: Symme Cryptographic Signature	
	MTLCCSACSTN3DOFHLaCLMTLCLOFXC7L.TRAIL*DaMCGAULMTLCCSACSTN3DOFHLOLLATLwaaPPMTTC	
	t 61D34FC3b1RN++ovaVCv2mKid8vUFR+M10CCScGSTb3D0FBR0UIMDYxD2MRcNVR1MMRkludCBD	
	OTE: MCEGCSGGSTb3D0EJA0wUZGdpanpv0GxvV2FaaG9zdC5tb20wHbcNMTTwNEDMTAvMD01WbcN	
	MT GWN HEOMTAW DOI W HAWNOO WCWYD YDYODDARUZXNOMR AWHOYJK AZ LINYGNAO KRDBBOZXNO OGY 4 YW1 W	
	bGUuY29tMIBTiANBakankiG9w0BAQEFAAOCAQ8AMIBCaKCAQEAwQIOc+vvzvPCWmBrZLxO78Gh	
	9GLsATOnVtwpZa3PYW6n0f41REsS+HAOShHTL8he2vymUSu4gy4MXLgEBf82+mMSrzfMY52mTMaY	
	Close	

These above steps test whether the private keys stored on the HSM are accessible for signing and decryption. Verify Djigzo log, all the above steps should not result in some kind of PKCS11 exception in the logs.