

URAC



Manual de implementación del servicio Murachí y Portal web en entornos de producción



01010

FIRMA ELECTRÓNICA

2019 CENDITEL

PUBLICADO POR EL CENTRO NACIONAL DE DESARROLLO E INVESTIGACIÓN EN TECNOLOGÍAS LIBRES

Este Manual se distribuye bajo la Licencia de Contenidos Versión 1.0, elaborada por la Fundación Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (CENDITEL), ente adscrito al Ministerio del Poder Popular para Ciencia y Tecnología (MPPCYT).

Todas los documentos, imágenes, audios, vídeos y código fuente contenidos en este manual están bajo la Licencia de Contenidos Versión 1.0, desarrollada por la Fundación CENDITEL. Cada vez que copie y distribuya este contenido debe acompañarlo de una copia de la licencia. Para más información sobre los términos y condiciones de la licencia visite la siguiente dirección electrónica: <u>http://derechoinformatico.cenditel.gob.ve/licencia-decontenido/</u>

Autor: Ing. Laura Colina

Índice

1. Presentación	4
2. Despliegue del servicio web Murachí	6
2.1 Requerimientos	6
2.2 Instalación de Apache Tomcat 7/8	6
2.3 Implementación del servicio Murachí	9
2.3.1 Cargar el código fuente	9
2.3.2 Crear y configurar un certificado SSL	12
3. Despliegue del portal web de Murachí	19
3.1 Requerimientos	19
3.2 Implementación del portal web con Apache	19
3.2 Implementación del portal web con Apache Tomcat	22

1. Presentación

Murachí es un servicio web para la firma y verificación de documentos firmados electrónicamente que ofrece las herramientas necesarias para incorporar, de manera sencilla, la funcionalidad de firma electrónica en sistemas como correo electrónico o páginas web. La firma electrónica es una acción jurídica que traslada el concepto de firma manuscrita al medio digital. Para ello se utiliza un código de verificación que permite al receptor comprobar la veracidad del documento recibido. Estos documentos van desde archivos de texto hasta paquetes multimedia de audio y vídeo.

El servicio web Murachí, permite incorporar la funcionalidad de firma y verificación de documentos en otros sistema de manera rápida y segura. Esto gracias a la implementación de la tecnología REST (*Representational State Transfer*), técnica de arquitectura de software basada en estándares que ofrece independencia de software y de lenguaje de programación. Murachí fue desarrollado en Software Libre por la Fundación Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (CENDITEL) bajo la licencia AGPL.

Murachí comprende dos subsistemas: Una interfaz gráfica o portal web y un subsistema API REST Murachí.

La Interfaz de Programación de Aplicaciones (API) son subrutinas, funciones o procedimientos que ofrece una biblioteca utilizados por otro software como una capa de abstracción (wikipedia). Es un conjunto de funciones o procedimientos utilizados por los programas informáticos para acceder a los servicios del sistema operativo, biblioteca de software u otros sistemas.

¿Cómo se usa?

El servicio web Murachí permite a los desarrolladores incorporar el sistema de firma electrónica en sus plataformas. En el enlace <u>https://murachi.cenditel.gob.ve/</u> se ofrece la

posibilidad de probar el servicio a través de una interfaz sencilla. Allí se describen los pasos para ilustrar de manera general el proceso.

Requerimientos previos

Antes de utilizar el sistema, el usuario debe poseer un certificado digital emitido por un Proveedor de Servicios de Certificación (PSC) acreditado ante la Superintendencia de Servicios de Certificación Electrónica (SUSCERTE) o una tarjeta para firma electrónica, esta ultima es de manera opcional, puesto que el servicio Murachí y su Portal web están diseñados para:

- Firmar electrónicamente documentos con formato PDF y/o BDOC mediante un token de seguridad (tarjeta o usb) los cuales almacenan las contraseñas y certificados digitales.
- Firmar electrónicamente documentos con formato PDF mediante certificados digitales con formato X.509 (PKCS12 extensión .p12).

Cabe destacar que Murachí y su Portal web presentan adicionalmente la funcionalidad de verificación de firma electrónica donde el único requerimiento previo es poseer el documento a comprobar.

Lugar de la publicación: https://seguridad.cenditel.gob.ve/proyectos/murachi/

2. Despliegue del servicio web Murachí

El servicio web Murachí es un sistema API REST que incorpora funcionalidades como: firmar y verificar documentos electrónicamente mediante certificados digitales emitidos por Proveedor de Servicios de Certificación.

Nota:

Usaremos \$ para describir los comandos que se ejecutarán con usuario regular. Usaremos # para describir los comandos que se ejecutarán con usuario administrador.

2.1 Requerimientos

- Sistema operativo: Debian 8 y/o 9
- Servidor web: Apache Tomcat 7 y/o 8
- Conjunto de desarrollo de OpenJDK (JDK):
 - Openjdk-7-jdk (7 y/o 8)
 - Openjdk-7-jre (7 y/o 8)

2.2 Instalación de Apache Tomcat 7/8

1. Instalar Tomcat 7/8 desde el repositorio de Debian 8/9

apt-get install tomcat7 tomcat7-admin tomcat7-examples tomcat7-docs

La estructura de directorio para Tomcat instalado desde el repositorio de Debian es la siguiente:

- En /var/lib/tomcat7 están los fuentes del servicio Apache Tomcat7:
 - En la carpeta *conf:* se encuentran los ficheros XML y los correspondientes DTD para la configuración de Tomcat, el cual esta enlazado a */etc/tomcat7*
 - En la carpeta *logs:* se encuentran logs de Catalina y de las aplicaciones, el cual esta enlazado a */var/log/tomcat7*

- En la carpeta *webapps:* se encuentran alojadas las aplicaciones web.
- En la carpeta *work:* se encuentra el almacenamiento temporal de ficheros y directorios.
- El servicio de Tomcat7 se ejecuta mediante systemd y/o init.d, sus configuración y guiones de arranque se encuentran en /etc/init.d/tomcat7 y /etc/default/tomcat
- 2. Para detener o iniciar el servicio de Tomcat se utiliza systemd o init.d:

systemctl stop tomcat7 # systemctl start tomcat7 # /etc/init.d/tomcat7 stop

/etc/init.d/tomcat7 start

3. Finalmente se levanta el navegador de su preferencia y coloca la ruta <u>http://localhost:8080</u> (ver figura 1)

ti works !

If you're seeing this page via a web browser; it means you've setup Tomcat successfully. Congratulations!
This is the default Tomcat home page. It can be found on the local filesystem at: /var/tib/tomcat/neebapps/N007/index.html
Tomcat7 veterans might be pleased to learn that this system instance of Tomcat is installed with CATALINA, HAVE in /usr/share/tomcat7 and CATALINA_BASE in /var/Lib/tomcat7, following the rules from /usr/share/doc/tomcat7-common/RUMING.txt.gz.
You might consider installing the following packages, if you haven't already done so:
tomcat7-docs: This package installs a web application that allows to browse the Tomcat 7 documentation locally. Once installed, you can access it by clicking here.
tomcat7-examples: This package installs a web application that allows to access the Tomcat 7 Servlet and JSP examples. Once installed, you can access it by clicking here.
tomcat7-admin: This package installs two web applications that can help managing this Tomcat instance. Once installed, you can access the manager webapp and the host-manager webapp.
NOTE: For security reasons, using the manager webapp is restricted to users with role "manager-gui". The host-manager webapp is restricted to users with role "admin-gui". Users are defined in /etc/towat7/towat-users.wnl.

Figura 1. Interfaz de Apache Tomcat7

En la interfaz podemos observar tres secciones:

- tomcat7-docs: documentación oficial de Apache Tomcat
- *tomcat7-examples:* ejemplos de aplicaciones del servicio Apache Tomcat
- tomcat7-admin: sistema de administración del Apache Tomcat
 - Manager WebApp: aplicación de administrador
 - Host-Manager WebApp: gerente de host
- 4. Editar los roles del servicio tomcat, los roles que se pueden encontrar son:
 - manager-gui: acceso a la interfaz HTML
 - manager-status: acceso a la pagina Estado del servidor

- manager-script
- manager-jmx

Para nuestro caso solo vamos a utilizar o definir los roles manager-gui y manager-status

Agregar dentro del archivo *tomcat-user.xml* los permisos de usuarios para el Tomcat
 # cd /etc/tomcat7/
 # vim tomcat-user.xml

Donde vamos agregar la siguiente sección:

<role rolename="manager-gui"/> <role rolename="admin-gui"/> <user username="tomcat" password="s3cret" roles="manager-gui"/> <user username="admin" password="s3cret" roles="admin-gui"/> </tomcat-users>

6. Guardar los cambios y reiniciar el servicio tomcat.

systemctl restart tomcat7

 Finalmente se prueba los roles creados en la pagina <u>http://localhost:8080</u> haciendo clic en el botón "Manager WebApp?". Se introduce el usuario y contraseña asignada en el archivo *tomcat-user.xml* (ver figura 2 y 3)

Iccalhost:8080		🔊 🔀 Q. Busc	ar 🔹 🛧 💮 🗏
It works !			
If you're seeing this page via a web browser, it means you've setup To	mcat successfully	. Congratulations!	
This is the default Tomcat home page. It can be found on the local file	esystem at: /var/lit	/tomcat7/webapps/ROOT/index.html	
Tomcat7 veterans might be pleased to learn that this system instance	of Tomcat is insta	ulled with CATALINA_HOME in /usr/share/toncat7 and CATALINA_BASE in /var/lib/toncat7, follo	wing the rules from /usr/share/doc/tomcat7-common/RUNNING.txt.gz.
You might consider installing the following packages, if you haven't a	lready done so:		
tomcat7-docs: This package installs a web application that allows to	browse the Tome	cat 7 documentation locally. Once installed, you can access it by clicking <u>here</u> .	
tomcat7-examples: This package installs a web application that allo	ows to access the '	Tomcat 7 Servlet and JSP examples. Once installed, you can access it by clickin	g <u>here</u> .
tomcat7-admin: This package installs two web applications that ca		Se requiere autenticación	host-manager webapp.
NOTE: For security reasons, using the manager webapp is restricted	Þ	http://localhost:8080 está pidiendo usuario y contraseña. El sitio dice: "Tomcat Manager Application	-gui". Users are defined in /etc/tomcat7/tomcat-users.xml.
	Nombre de usuario:	tomcat	
	Contraseña:		
		Cancelar <u>Aceptar</u>	

Figura 2. Interfaz de autenticación para acceder a Manager WebApp

Server Status

Figura 3. Interfaz de Manager WebApp

Hasta este instante se tiene corriendo el servidor Apache Tomcat 7/8 (que es un contenedor de aplicaciones web) y se configuro los usuarios para acceder a las diferentes secciones. Ahora procederemos a mostrar los pasos para correr una aplicación web en el contenedor Apache tomcat 7/8 para la misma debemos de disponer un archivo de aplicación web (.war)

Nota: WAR es un archivo JAR utilizado para distribuir una colección de JavaServer? Pages, servlets, clases Java, archivos XML, bibliotecas de tags y páginas web estáticas (HTML y archivos relacionados) que juntos constituyen una aplicación web. Referencia web: <u>https://es.wikipedia.org/wiki/WAR_(archivo)</u>

2.3 Implementación del servicio Murachí

2.3.1 Cargar el código fuente

1. Para este manual vamos a correr el servidor Murachí que se dispone del archivo .war en el siguiente enlace: <u>https://tibisay.cenditel.gob.ve/murachi/downloads</u>

- 2. Para cargar un archivo .war en Tomcat se puede realizar de dos maneras:
 - 2.1. Desde un terminal:
 - 2.1.1. Subir el archivo Murachi.war al servidor.
 - 2.1.2. Mover el archivo al directorio /var/lib/tomcat7/webapps/

mv Murachi.war /var/lib/tomcat7/webapps/

- 2.2. Usando la interfaz gráfica del servidor Tomcat:
 - 2.2.1. Ir a la sección Manager WebApp?.
 - 2.2.2. Bajar hasta la sección Archivo WAR a desplegar o WAR file to deploy (ver figura 4).

Jung in costs a web application has caused a memory leak on ston, reload or underlow						
спеск со зее н а мез арупкаком паз саизец а тепло у как он зхор, тепло и оницерну						

Figura 4. Sección Archivo WAR a desplegar

2.2.3. Hacer clic en el botón *Seleccionar archivo* luego de seleccionar el archivo Murachi.war hacer clic en el botón *Desplegar* (ver figura 5).

Control localhost:8080/man		ıl						😻 🥐 🔍 Buscar			(III) 🗢 🐨 🔍 III
							Start S	Stop Reload	Undeploy		
/docs	None sj	Cancelar			S	ubir archivo			Q	Abrir	
	None s	Recientes	CENDITEL	Seguridad-Murachi	Instalacion de Murac	:hi y portal 🕞					
/examples		☆ Carpeta personal	Nombre						Tamaño	Modificado 🔺	
			VersionApiMurachi.war						35,3 kB	25 feb	
theat manager	Nana		VersionApiMurachi.html						882 bytes	19 feb	
/nost-manager	None sj	Documentos	📄 Murachi.war						26,3 MB	19 feb	
		Imágenes	pruebaservicioweb.war						996,3 kB	19 feb	
/manager	None s		apache-tomcat-7.0.92.tar.gz						9,1 MB	19 feb	
		də Müsica									
Doplay		Vídeos									
Deploy directory or WAR fil	e locate	Servicios									
		Seguridad-Mura									
		Recursos Huma									
		CORPOANDES									
		Comite Tecnico									
WAR file to deploy	_	🛅 Comandos y pr									
		E CENDITEL									
	_	Datos servidores									
Diagnostics	_	🖿 Mis imagenes									
Check to see if a web appl	ication										
Find leaks									Todos I	os archivos 👻	
Server Information											
Tome	at Versi	on	IVM Version		tendor	OS Name	OS Version	05	Architecture	Hostname	IP Address
Apache Tomo	at/7.0.56	5 (Debian)	1.7.0_181-b01	Oracle Co	orporation	Linux	4.9.0-6-amd64		amd64	-	-
					Convright © 1999-201	4. Anache Software Found	ation				

Figura 5. Selección del archivo Murachi.war a desplegar

2.2.4. Una ves cargado y desplegado en archivo Murachi.war se puede observar en la sección *Aplicaciones* que se visualizan las aplicaciones que están contenida en el servidor Tomcat que ya aparece la aplicación Murachi (ver figura 6).

() localhost:8080/man	ager/html/upload?org.apache.ca	talina.filters.CSRF_NONCE=3C37D1A0E3A2A43F52948CD4C0737572			👽 🥂 🔍 Buscar	• *= 🕹 🄝 =
Software F	Apache Foundation					
		Tomcat We	b Applica	tion Manag	ger	
Message:	ĸ					
Manager List Applications		HTML Manager Help			Manager Help	Server Status
Applications	(Manalan	Planta Marca	Parentine .	and the second se	and the second se	
1	None specified		true	<u>0</u>	Start Stop Reload Undeploy Expire sessions with idle æ 30 minutes	
/Murachi	None specified	Murachi	true	Q	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes	
/docs	None specified	Tomcat Documentation	true	Q	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes	
/examples	None specified	Servlet and JSP Examples	true	٥	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes	
<u>/host-manager</u>	None specified	Tomcat Host Manager Application	true	٥	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes	
<u>Imanager</u>	None specified	Tomcat Manager Application	true	2	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes	
Deploy Deploy directory or WAR fil	e located on server					

Figura 6. Sección Aplicaciones - Lista de aplicaciones en Tomcat

3. Crear un directorio con el nombre de *murachiWorkingDirectory* en el servidor Tomcat donde se almacenara los archivos que se van a firmar:

cd /var/lib/tomcat7/ # mkdir murachiWorkingDirectory # chown -R tomcat7:tomcat7 murachiWorkingDirectory/

Nota: Para tomcat 8 se crea el directorio /var/lib/tomcat7/ y el enlace simbólico a .../tomcat8/murachiWorkingDirectory debido a que MurachiRest requiere un directorio llamado /var/lib/tomcat7/ # mkdir /var/lib/tomcat7/ # cd /var/lib/tomcat7/ # cd /var/lib/tomcat7/

In -s ../tomcat8/murachiWorkingDirectory/

2.3.2 Crear y configurar un certificado SSL

Para la implementación del servicio Murachí es necesario que se ejecute mediante el protocolo HTTPS (SSL), por tal motivo se debe crear y configurar el certificado SSL para el servicio, en caso de tener una capa Proxy, el mismo hospedará el certificado y las configuraciones para abrir el puerto 443. En el caso de no poseer capa Proxy, se debe crear y configurar el certificado SSL en el servidor donde se esta implementando el servicio Murachí, para este este caso a continuación se describe un ejemplo de la creación de un certificado auto-firmado y la configuración necesaria para que se ejecute con el Apache Tomcat.

 Crear el par de clave (publica y privada) del certificado auto-firmado, donde se indica la creación de una solicitud (crt) de certificado x509, usando algoritmo sha256, la creación de claves (key) con tamaño 2048 y la salida en los archivos tomcat.key y tomcar.crt:

openssl req -x509 -sha256 -newkey rsa:2048 -keyout tomcat.key -out tomcat.crt -days 1024 -nodes

Generating a RSA private key

.....+++++

.....+++++

writing new private key to 'tomcat.key'

You are about to be asked to enter information that will be incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

12

Country Name (2 letter code) [AU]:VE State or Province Name (full name) [Some-State]:Merida Locality Name (eg, city) []:Merida Organization Name (eg, company) [Internet Widgits Pty Ltd]:Fundacion Cenditel Organizational Unit Name (eg, section) []:Seguridad Common Name (e.g. server FQDN or YOUR name) []:Murachi Email Address []:<u>seguridad@cenditel.gob.ve</u>

2. Exportar ambos archivos que se generaron en la sección anterior (tomcat.crt y tomcat.key) al almacén de claves keystore:

openssl pkcs12 -export -out keystore.pkcs12 -in tomcat.crt -inkey tomcat.key Enter Export Password: Verifying - Enter Export Password:

3. Instalar el archivo de certificado en el almacén de claves:

keytool -importkeystore -srckeystore keystore.pkcs12 -srcstoretype PKCS12 -destkeystore keystore.jks -deststoretype JKS

4. Para la configuración del Conector SSL (Tomcat necesitará un conector SSL configurado para poder aceptar las conexiones seguras) abrir el archivo *server.xml* que se encuentra en /var/lib/tomcat7/conf.

vim server.xml

5. Buscar el conector que desea proteger con el nuevo almacén de claves y, si es necesario, elimine el comentario (generalmente, se trata de un conector con el puerto 443 o 8443 como se muestra en el ejemplo siguiente). Especifique el nombre de archivo y la contraseña correctos del almacén de claves en la configuración del conector. Cuando finalice, el conector debe tener una apariencia similar al siguiente ejemplo:

<!-- Define a SSL HTTP/1.1 Connector on port 8443

This connector uses the BIO implementation that requires the JSSE style configuration. When using the APR/native implementation, the OpenSSL style configuration is required as described in the APR/native documentation -->

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile="/var/lib/tomcat7/keystore.jks" keystorePass="123456"
```

6. Guardar los cambios en el archivo server.xml y se procede a reinicie Tomcat. Para verificar el funcionamiento del certificado ssl se procede a levantar un navegador de su preferencia y vuelva a levantar el servidor pero agregando el protocolo https (<u>https://localhost:8443/</u>). En este manual el certificado es auto-firmado por tal motivo se debe ejecutar los siguientes pasos:

6.1. Aceptar el certificado auto-firmado:



Figura 7. Aceptar el certificado auto-firmado

6.2. Añadir agregar excepción de seguridad:

Ø	La conex	Agregar excepción de seguridad	×	
	Atria Informar errores localhost uses an inve The certificate is not The certificate is not Error code: SEC_ERR Agregar excepción	Guardar germanentemente esta excepción Confirmar excepción de seguridad Cancelar Confirmar excepción de seguridad Cancelar Confirmar escenarase:	Avanzadas	

Figura 8. Confirmar excepción de seguridad

6.3. Finalmente se puede observar la interfaz de configuración de Tomcat:



Figura 9. Interfaz de Tomcat con SSL

- 7. Para cambiar los puertos de los protocolos HTTP y HTTPS que el servicio Tomcat trae por defecto, se debe ejecutar los siguientes pasos:
 - 7.1. Activar la opción de AUTHBIND en el archivo /etc/default/tomcat7

vim /etc/default/tomcat7

AUTHBIND=yes

7.2. Editar los puertos en el archivo server.xml

vim /etc/tomcat7/server.xml

```
<!-- A "Connector" represents an endpoint by which requests are received
  and responses are returned. Documentation at :
  Java HTTP Connector: /docs/config/http.html
  Java AJP Connector: /docs/config/ajp.html
  APR (HTTP/AJP) Connector: /docs/apr.html
  Define a non-SSL/TLS HTTP/1.1 Connector on port 8080
-->
<Connector port="80" protocol="HTTP/1.1"
      connectionTimeout="20000"
      redirectPort="443" />
<!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443</p>
   This connector uses the NIO implementation. The default
   SSLImplementation will depend on the presence of the APR/native
  library and the useOpenSSL attribute of the
  AprLifecycleListener.
  Either JSSE or OpenSSL style configuration may be used regardless of
```

the SSLImplementation selected. JSSE style configuration is used below.

-->

```
<Connector port="443" protocol="org.apache.coyote.http11.Http11NioProtocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile="/var/lib/tomcat7/keystore.jks" keystorePass="123456" />
```

7.3. Reiniciar el servicio:

systemctl restart tomcat7

7.4. Acceder a https://localhost:443/ o https://localhost/

7.5. Para cambiar el nombre de dominio del servicio, se debe editar el archivo *server.xml*, agregando la linea de Alias:

vim /etc/tomcat7/server.xml

<Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true"> <Alias>murachi.prueba.cenditel.gob.ve</Alias>

7.6. Reiniciar el servicio y acceder a https://murachi.prueba.cenditel.gob.ve

systemctl restart tomcat7



Figura 10. Interfaz de Tomcat con SSL y su dominio

Nota: En caso de tener un servidor proxy que proporcionará el servicio https con los certificados SSL, se debe obviar todos los pasos anteriores y solo agregar las siguientes lineas en server.xml:

vim server.xml

```
<!-- Mark HTTP as HTTPS forward from SSL termination at nginx proxy -->
<Valve className="org.apache.catalina.valves.RemoteIpValve"
remoteIpHeader="x-forwarded-for"
remoteIpProxiesHeader="x-forwarded-by"
protocolHeader="x-forwarded-proto"
internalProxies="X\.X\.X\.X"/>
```

Donde X.X.X.X es la dirección IP de escucha del servicio Proxy.

Posteriormente agregar las siguientes lineas en web.xml (@@ -391,18 +391,6 @@) # vim web.xml

<!-- agregado para redirigir trafico HTTP a HTTPS --> <security-constraint> <web-resource-collection> <web-resource-name>Protected Context</web-resource-name> <url-pattern>/*</url-pattern> </web-resource-collection> <!-- auth-constraint goes here if you requre authentication --> <user-data-constraint> <transport-guarantee>CONFIDENTIAL</transport-guarantee> </user-data-constraint> </security-constraint>

3. Despliegue del portal web de Murachí

El portal web que se ofrece en este manual consume de los servicios Murachí y PortableSigner, este ultimo es un programa de firma (con certificados X.509) para archivos PDF. El portal brinda una interfaz gráfica sencilla y rápida al usuario final donde podrá:

- 1. Firmar electrónicamente archivos PDF y BDOC con dispositivo criptográfico.
- 2. Firmar electrónicamente archivos PDF con certificados digitales .p12.
- 3. Verificar firmas electrónicas.

Nota:

Usaremos \$ para describir los comandos que se ejecutarán con usuario regular. Usaremos # para describir los comandos que se ejecutarán con usuario administrador.

3.1 Requerimientos

- Sistema operativo: Debian 8 y/o 9
- Servidor web: Apache, Nginx o Apache Tomcat
- Conjunto de desarrollo de OpenJDK (JDK):
 - Openjdk-7-jdk (7 y/o 8)
 - Openjdk-7-jre (7 y/o 8)

En el presente manual se abordará dos maneras de implementar el portal web:

- 1. Utilizando el servidor web Apache
- 2. Utilizando el servidor web Apache Tomcat

3.2 Implementación del portal web con Apache

1. Instalar Apache:

apt-get install apache2 libapache2-mod-php7.0

2. Descargar el código fuente del portal web:

cd /var/www

git clone -b portal <u>https://tibisay.cenditel.gob.ve/murachi/scm/git/portal_2019.git</u>

3. Crear el directorio donde se guardarán los archivos gestionados para la firma electrónica con certificados .p12:

mkdir tmp

chown -R www-data:www-data tmp

4. Copiar la plantilla de sitios web de Apache:

cp 000-default.conf murachi.cenditel.gob.ve.conf

5. Editar el sitio web *murachi.cenditel.gob.ve.conf*, quedando de la siguiente manera:

<VirtualHost *:80>

The ServerName directive sets the request scheme, hostname and port that
the server uses to identify itself. This is used when creating
redirection URLs. In the context of virtual hosts, the ServerName
specifies what hostname must appear in the request's Host: header to
match this virtual host. For the default virtual host (this file) this
value is not decisive as it is used as a last resort host regardless.
However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin webmaster@localhost DocumentRoot /var/www/portal_2019

<Directory /> Options FollowSymLinks AllowOverride None </Directory> <Directory /var/www/portal_2019> Options Indexes FollowSymLinks MultiViews AllowOverride None Order allow,deny allow from all </Directory>

Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
error, crit, alert, emerg.
It is also possible to configure the loglevel for particular
modules, e.g.
#LogLevel info ssl:warn

ErrorLog \${APACHE_LOG_DIR}/error.log CustomLog \${APACHE_LOG_DIR}/access.log combined

For most configuration files from conf-available/, which are
enabled or disabled at a global level, it is possible to
include a line for only one particular virtual host. For example the
following line enables the CGI configuration for this host only
after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf

</VirtualHost>

vim: syntax=apache ts=4 sw=4 sts=4 sr noet

6. Desactivar el sitio web por defecto:# a2dissite 000-default.conf

7. Activar el sitio web murachi.cenditel.gob.ve.conf:# a2ensite murachi.cenditel.gob.ve.conf

8. Reiniciar el servicio de Apache:

9. Finalmente se puede observar el portal web:

€ → ୯ û	🛈 🚔 https://murachi.cenditel.gob.ve/php/sign.php		🛡 🏠 🔍 Buscar	± II\ ⊡ ≡
	Gobierno Bolivariano Ministerio del Por de Venezuela para Ciencia y T r	der Popular Centro Nacional de Desarrollo e ecnología Investigación en Tecnologías Lit	rres (Cenditel)	
		I de Firma Electrónica y Ver	HII ificación	
	Firmar.con.token Firmar.con.p12 Verificar	inicio Acerca de instructivo Equip	o Preguntas frecuentes	
	Firmar Elect	rónicamente PKCS12		
	Seleccionar PDF	C.		I
	Seleccionar certificado PKCS12			
	Contraseña del certificado			I
	Esta contraseña no será guardada en el servidor.			i
		Firmar		
				_

Figura 11. Portal Web del Servicio Murachí

Nota: Se recomienda que el portal web para el servicio Murachí se publique mediante el protocolo HTTPS, por tal motivo si se tiene un Proxy se debe configurar el puerto 443 y la redirección del puerto 80 al 443.

3.2 Implementación del portal web con Apache Tomcat

1. Descargar el código fuente del portal web:

cd /var/www

git clone -b portal https://tibisay.cenditel.gob.ve/murachi/scm/git/portal_2019.git

- 2. Crear un proyecto nuevo de Java con IDE Eclipse para la carpeta portal_2019
- Ejecutar el proceso de cargar y desplegar en el servidor Tomcat, para la misma debemos realizar los pasos de la sección anterior: 2.3 Implementación del servicio Murachí (.war) > 2.3.1 Cargar archivo .war

 Ejecutar el portal haciendo clic en el enlace o escribiendo <u>https://murachi.cenditel.gob.ve/portal_2019</u> desplegara el portal que realiza el consumo del servicio de Murachí

Para el consumo del servicio de Murachí desde su portal web se requiere:

- Para firmar electrónicamente con dispositivos criptográficos (eToken): instalar un complemento de firma en el navegador Chrome quien será el encargado de gestionar los dispositivos criptográficos, en tal sentido debe realizar la instalación del mismo siguiendo los paso de instalación descrito en el siguiente enlace <u>https://murachi.cenditel.gob.ve/configuracionInstalacion/contenido_1_instalar_el_com</u> <u>plemento de firma esteidfirefoxplugin.html</u>
- 2. Para firmar electrónicamente con certificados digitales (.p12): obtener el certificado digital .p12.