

Omnichannel Transaction Collector v1.0.0

Manual de Instalación

Índice de Contenidos

[Índice de Contenidos 2](#_Toc523761326)

[**Introducción** 4](#_Toc523761327)

[**Antecedentes** 4](#_Toc523761328)

[**Objetivo** 4](#_Toc523761329)

[**Información de Despliegue** 5](#_Toc523761330)

[**Instalación de Software Base** 6](#_Toc523761331)

[1 Prerrequisitos 6](#_Toc523761332)

[1 Creación de usuario OTC 7](#_Toc523761333)

[1 Configuración de Server Proxy (en el caso de salir a Internet por Proxy) 7](#_Toc523761334)

[2 Creación de Certificado Auto firmado 7](#_Toc523761335)

[3 Instalación y configuración Docker 8](#_Toc523761336)

[4 Crear una separada partición para contenedores. 10](#_Toc523761337)

[5 Configuración de Proxy Server en Docker 11](#_Toc523761338)

[6 Instalación y configuración Docker Compose 11](#_Toc523761339)

[7 Instalación y configuración Docker Swarm 12](#_Toc523761340)

[1 Instalación y configuración GIT 12](#_Toc523761341)

[2 Auditar todas las actividades del daemon de Docker 12](#_Toc523761342)

[3 Restringir y encriptar trafico de red entre contenedores 13](#_Toc523761343)

[4 Habilitar soporte user namespace 13](#_Toc523761344)

[5 Configuración centralizada y remota de regitro de LOGs 13](#_Toc523761345)

[6 Deshabilitar operaciones en registro legado 14](#_Toc523761346)

[7 Deshabilitar Userland proxy 14](#_Toc523761347)

[8 Habilitar contenido valido para docker 14](#_Toc523761348)

[9 Proteger Docker Daemon 14](#_Toc523761349)

[10 Instalación y configuración solución OTC 16](#_Toc523761350)

# **Introducción**

Omnichannel Transaction Collector (OTC) permite centralizar la administración y ejecución de los servicios de recaudo del Banco, desde una sola interfaz o API por medio de acceso. Los servicios de recaudaciones son configurados por empresa y tipo de recaudación (ejemplo: recaudación de Agua), y habilitados por canal (ejemplo: Banca en línea); cada canal este asociado a un medio de acceso (ejemplo: soap/xml), sobre el cual se expone un API con las operaciones de consulta, pago y reverso; cada una de las operaciones representan un flujo de ejecución que puede ser parametrizado para invocar los diferentes servicios de recaudo expuesto por el Banco.

# **Antecedentes**

El Banco Bolivariano se encuentra en la implementación del proyecto Cash Management, para el cual ha adquirido una solución tecnología que le permite brindar los servicios de Cash Management a sus clientes por medio de la Banca en línea. Este producto permite consultar y ejecutar servicios de recaudo expuesto con por el Banco, sin embargo, requiere de un servicio (SOAP/XML con un contrato definido) que centralice todos los servicios de recaudos del Banco.

El Banco Bolivariano requiere que el servicio de recaudo centralizado pueda ser reutilizado por otros canales (omnicanal), adicional que sea administrable.

# **Objetivo**

El presente documento tiene como objetivo principal definir los pasos para la instalación de la solución OTC

# **Información de Despliegue**

|  |  |  |  |
| --- | --- | --- | --- |
| **Artefacto** | **Tipo artefacto** | **Tipo Despliegue** | **Servidor** |
| otc\_web.jar | JAR | Contenedor | CCMSHOSTSRV |
| otc-admin.jar | JAR | Contenedor | CCMSHOSTSRV |
| otc-admin.jar | JAR | Contenedor | CCMSHOSTSRV |
| otc-admin.jar | JAR | Contenedor | CCMSHOSTSRV |
| otc-proxy | NGINX | Contenedor | CCMSHOSTSRV |
| otc-cache | Redis | Contenedor | CCMSHOSTSRV |

# **Instalación de Software Base**

## **Prerrequisitos**

* El servidor debe tener instalador la versión del sistema operativo definida para el proyecto (CentOS 7).
* Se debe tener acceso a root (por lo menos para las tareas definidas como root).
* El servidor deberá tener los siguientes sistemas de archivos (filesystem):
  1. /otc
  2. /logs
  3. /cache
  4. /docker (recomendado por seguridad informática)
* El servidor deben tener acceso a internet en el momento de la instalación.
* Se recomienda tener instalado GIT en los servidores para obtener los instaladores de los repositorios centralizados de GIZLO, caso contrario en los pasos donde se descargue los instaladores se debe reemplazar por copiar los instaladores en el servidor correspondiente.
* El servidor debe exponer los puertos 8090 y 8091, sobre HTTPS.
* El servidor debe tener acceso al servidor IBM MQ, Oracle OSB, Oracle Database y CCADMHOSTSRV (172.17.1.167)
* La base de datos debe estar instalada y se debe haber ejecutado los scripts de creación y carga iniciar del proyecto OTC.
* Todos los registros (LOGS) de la solución se almacenarán en el sistema de archivos /logs.

## **Creación de usuario OTC**

Esta configuración es requerida por seguridad informática, la instalación no debe ser con el usuario root (excepto los puntos descritos en el documento).

1. Crear el usuario otc:

adduser otc

1. Modificar la clave del usuario:

passwd otc

1. Agregar usuario a grupo sudors:

gpasswd -a otc wheel

usermod -aG wheel otc

1. Crear grupo docker:

sudo groupadd docker

1. Agregar usuario a grupo docker:

sudo usermod -aG docker otc

1. Actualizar el archivo /etc/sudoers (incluir la segunda línea dentro del archivo, y guardar el archivo)

visudo -f /etc/sudoers

otc ALL=(ALL) ALL

1. Cambiar al usuario otc:

su otc

## **Configuración de Server Proxy (en el caso de salir a Internet por Proxy)**

Esta configuración es requerida para poder tener acceso a Internet. Todos los paquetes requeridos a nivel de sistema operativo son instalados a través de la herramienta YUM

1. Configuración a nivel de sistema operativo:

sudo export http\_proxy=http://<IP\_PROXY>:<PUERTO\_PROXY>

sudo export https\_proxy=http://<IP\_PROXY>:<PUERTO\_PROXY>

**NOTA**: Reemplazar <IP\_PROXY> por la IP del servidor Proxy, y <PUERTO\_PROXY> por el puerto.

## **Creación de Certificado Auto firmado**

Esta configuración es requerida en para activas puertos HTTPS.

1. Crear directorio para certificados

sudo mkdir /root/certs && cd /root/certs

1. Crear el certificado, ejecutando el siguiente comando (cambiar el nombre del certificado y clave, de acuerdo con el servidor correspondiente):

openssl req -new -newkey rsa:4096 -x509 -sha256 -days 1095 -nodes -out hostname.crt -keyout hostname.key

1. Configurar el certificado con los datos de la empresa. En el atributo **Common Name** debe llevar la IP o **nombre de dominio** del servidor.

## **Instalación y configuración Docker**

Para la instalación de Docker sobre CentOS o Redhat 7, tenemos dos opciones, una basados en su administrador (para nuestro caso Rancher), y la otra de forma independiente. A continuación, las dos opciones:

NOTA: Se recomienda la OPCION 2.

**OPCIÓN 1 (Sin dependencia RANCHER)**

1. Remover cualquier versión de Docker instalada en el servidor. Si no hay una versión antigua no es necesario este paso.

sudo yum remove docker \

docker-client \

docker-client-latest \

docker-common \

docker-latest \

docker-latest-logrotate \

docker-logrotate \

docker-selinux \

docker-engine-selinux \

docker-engine

1. Actualizar paquetes del sistema operativo.

sudo yum check-update

1. Instalar paquetes requeridos por Docker.

sudo yum install -y yum-utils \

device-mapper-persistent-data \

lvm2

1. Agregar repositorio yum Docker en el sistema operativo.

sudo yum-config-manager \

--add-repo \

https://download.docker.com/linux/centos/docker-ce.repo

1. Instalar Docker CE.

sudo yum install docker-ce-18.03.1.ce

1. Iniciar Docker.

sudo systemctl start docker

1. Verificar que Docker este corriendo en el servidor.

sudo systemctl status docker

* 1. Respuesta esperada de la verificación:

docker.service - Docker Application Container Engine

Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)

Active: active (running) since Sun 2016-05-01 06:53:52 CDT; 1 weeks 3 days ago

Docs: https://docs.docker.com

Main PID: 749 (docker)

1. Configurar inicio automatico cuando el servidor se reinicie.

sudo systemctl enable docker

**OPCIÓN 2 (Depende de RANCHER)**

1. Remover cualquier versión de Docker instalada en el servidor. Si no hay una versión antigua no es necesario este paso.

sudo yum remove docker \

docker-client \

docker-client-latest \

docker-common \

docker-latest \

docker-latest-logrotate \

docker-logrotate \

docker-selinux \

docker-engine-selinux \

docker-engine

1. Actualizar paquetes del sistema operativo.

sudo yum check-update

1. Instalar Docker CE basado en Rancher

curl https://releases.rancher.com/install-docker/18.03.sh | sh

1. Verificar que Docker este corriendo en el servidor.

sudo systemctl status docker

* 1. Respuesta esperada de la verificación:

docker.service - Docker Application Container Engine

Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)

Active: active (running) since Sun 2016-05-01 06:53:52 CDT; 1 weeks 3 days ago

Docs: https://docs.docker.com

Main PID: 749 (docker)

1. Configurar inicio automático cuando el servidor se reinicie.

sudo systemctl enable docker

## **Crear una separada partición para contenedores.**

Esta sección cubre recomendaciones de seguridad que debe seguir para preparar la máquina host que planea usar para ejecutar cargas de trabajo en contenedores

1. Identificar el Filesystem docker en el servidor para la instalación, en el caso de no existir este Filesystem, crear el directorio, y asignar como dueño a el usuario otc.

sudo chown otc:docker /docker

1. Modificar el archivo /usr/lib/systemd/system/docker.service, agregando la información del directorio /docker.

vim /usr/lib/systemd/system/docker.service

1. Modificando el parámetro ExecStart de la sección [Service] para añadir el parámetro -g /docker
2. Refrescar estado de Docker

sudo systemctl daemon-reload

1. Verifique que la configuración se haya cargado:

sudo systemctl show --property Environment docker

1. Reiniciar Docker:

sudo systemctl restart docker

## **Configuración de Proxy Server en Docker**

1. Crear un directorio de instalación de sistema para el servicio de acoplador:

mkdir /etc/systemd/system/docker.service.d

1. Crear un archivo llamado /**etc/systemd/system/docker.service.d/http-proxy.conf** que agregue la variable de entorno **HTTP\_PROXY**:

vim /etc/systemd/system/docker.service.d/http-proxy.conf

Agregar dentro del archivo:

[Service]  
Environment = "HTTP\_PROXY = http://<IP\_PROXY>:<PUERTO\_PROXY>/"

1. Actualizar el archvio ~/.docker/config.json

vim ~/.docker/config.json

**Agregar dentro del archivo:**

{

"proxies":

{

"default":

{

"httpProxy": "http:// <IP\_PROXY>:<PUERTO\_PROXY>"

}

}

}

1. Refrescar estado de Docker

sudo systemctl daemon-reload

1. Verifique que la configuración se haya cargado:

sudo systemctl show --property Environment docker

1. Reiniciar Docker:

sudo systemctl restart docker

## **Instalación y configuración Docker Compose**

Para la instalación de Docker Compose sobre CentOS 7, vamos a ejecutar los siguientes comandos:

1. Instalar Docker Compose

sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose

1. Aplicar permisos de ejecución

sudo chmod +x /usr/local/bin/docker-compose

1. Probar instalación

docker-compose --version

## **Instalación y configuración Docker Swarm**

Para la instalación de Docker Swam sobre CentOS 7, vamos a ejecutar los siguientes comandos:

1. Inicializar Docker Swam (esta configuración requiere desbloquear cada vez que se requiere levantar servicios, se aplica recomendación)

docker swarm init --autolock

## **Instalación y configuración GIT**

Esta instalación y configuración es requerida en **todos los servidores**.

1. Instalar GIT en servidor:

sudo yum install git

1. Configurar proxy:

git config --global http.proxy http:// <IP\_PROXY>:<PUERTO\_PROXY>

Reemplazar <IP\_PROXY> por la IP del servidor Proxy, y <PUERTO\_PROXY> por el puerto.

## **Auditar todas las actividades del daemon de Docker**

1. Modificar archivo etc/audit/rules.d/audit.rules

sudo vim /etc/audit/rules.d/audit.rules

1. Agregar reglas de auditoria (en el archivo)

-w /usr/bin/docker -k docker

-w /usr/lib/systemd/system/docker.service -k docker

-w /usr/lib/systemd/system/docker.socket -k docker

-w /etc/default/docker -k docker

-w /etc/docker/daemon.json -k docker

-w /usr/bin/docker-containerd -k docker

-w /usr/bin/docker-runc -k docker

1. Reiniciar servicio de auditoria

sudo service auditd restart

## **Restringir y encriptar trafico de red entre contenedores**

1. Eliminar red por defecto de docker swarm

docker network disconnect -f docker\_gwbridge gateway\_ingress-sbox

docker network rm docker\_gwbridge

1. Actualizar red por defecto de docker swarm (verificar valores de SUBNET y GATEWAY)

SUBNET=172.20.0.0/20

GATEWAY=172.20.0.1

docker network create --subnet=${SUBNET} --gateway ${GATEWAY} -o encrypted -o com.docker.network.bridge.enable\_icc=false -o com.docker.network.bridge.name=docker\_gwbridge docker\_gwbridge

## **Habilitar soporte user namespace**

No se lo recomienda debido a las limitaciones en ejecución de contenedores con docker compose y stack.

## **Configuración centralizada y remota de regitro de LOGs**

1. Modificar archivo /etc/docker/daemon.json

sudo vim /etc/docker/daemon.json

1. Agregar la siguiente configuración:

"log-driver": "syslog",

"log-opts": {

"syslog-address": "udp://<IP SERVIDOR>:<PUERTO SERVIDOR>"

}

**NOTA**: actualizar los valores de <IP SERVIDOR>:<PUERTO SERVIDOR>

1. Refrescar estado de Docker

sudo systemctl daemon-reload

1. Verifique que la configuración se haya cargado:

sudo systemctl show --property Environment docker

1. Reiniciar Docker:

sudo systemctl restart docker

## **Deshabilitar operaciones en registro legado**

1. Modificar archivo /etc/docker/daemon.json

sudo vim /etc/docker/daemon.json

"icc": false,

"disable-legacy-registry":true

1. Refrescar estado de Docker

sudo systemctl daemon-reload

1. Verifique que la configuración se haya cargado:

sudo systemctl show --property Environment docker

1. Reiniciar Docker:

sudo systemctl restart docker

## **Deshabilitar Userland proxy**

1. Modificar archivo /usr/lib/systemd/system/docker.service

vim /usr/lib/systemd/system/docker.service

--userland-proxy=false

1. Refrescar estado de Docker

sudo systemctl daemon-reload

1. Verifique que la configuración se haya cargado:

sudo systemctl show --property Environment docker

1. Reiniciar Docker:

sudo systemctl restart docker

## **Habilitar contenido valido para docker**

1. Agregar variable de ambiente

export DOCKER\_CONTENT\_TRUST=1

**Agregar en**

vim /etc/profile

## **Proteger Docker Daemon**

Las siguientes tareas ejecutar con el usuario **root**:

1. Crear el directorio /etc/docker/certs.d/, e ingresar en este

mkdir /etc/docker/certs.d/

cd /etc/docker/certs.d/

1. Generar CA privado y publico

HOST=1.1.1.1

openssl genrsa -aes256 -out ca-key.pem 4096

openssl req -new -x509 -days 365 -key ca-key.pem -sha256 -out ca.pem

openssl genrsa -out server-key.pem 4096

openssl req -subj "/CN=$HOST" -sha256 -new -key server-key.pem -out server.csr

echo subjectAltName = DNS:$HOST,IP:1.1.1.1,IP:127.0.0.1 >> extfile.cnf

echo extendedKeyUsage = serverAuth >> extfile.cnf

openssl x509 -req -days 365 -sha256 -in server.csr -CA ca.pem -CAkey ca-key.pem \

-CAcreateserial -out server-cert.pem -extfile extfile.cnf

**NOTA:** reemplazar la variable HOST por la IP de la maquina (o nombre de dominio), y la IP 1.1.1.1 por la IP de la maquina

1. Generar certificado autofirmado

openssl genrsa -out key.pem 4096

openssl req -subj '/CN=client' -new -key key.pem -out client.csr

echo extendedKeyUsage = clientAuth >> extfile.cnf

openssl x509 -req -days 365 -sha256 -in client.csr -CA ca.pem -CAkey ca-key.pem -CAcreateserial -out cert.pem -extfile extfile.cnf

rm -v client.csr server.csr

1. Remover permisos de escritura

chmod -v 0400 ca-key.pem key.pem server-key.pem

chmod -v 0444 ca.pem server-cert.pem cert.pem

1. Modificar archivo /usr/lib/systemd/system/docker.service

vim /usr/lib/systemd/system/docker.service

--tlsverify --tlscacert=/etc/docker/certs.d/ca.pem --tlscert=/etc/docker/certs.d/server-cert.pem --tlskey=/etc/docker/certs.d/server-key.pem -H=0.0.0.0:2376

1. Refrescar estado de Docker

sudo systemctl daemon-reload

1. Verifique que la configuración se haya cargado:

sudo systemctl show --property Environment docker

1. Reiniciar Docker:

sudo systemctl restart docker

Las siguientes tareas ejecutar con el usuario **otc**:

1. Crear directorio para cliente, y copiar archivos de aseguramiento:

mkdir -pv ~/.docker

cp -v \* ~/.docker

1. Agregar variables de ambiente para conexión segura

export DOCKER\_HOST=tcp://$HOST:2376

export DOCKER\_TLS\_VERIFY=1

**Agregar en**

vim /etc/profile

1. Probar conexión

docker ps

## **Instalación y configuración solución OTC**

Para la instalación de la solución OTC sobre Docker, vamos a ejecutar los siguientes comandos:

1. Ingresar al director otc

cd otc

1. Descargar del repositorio GIT los archivos Docker Compose. Caso contrario copiar de CD de instaladores.

git clone <https://andresgiler@bitbucket.org/gizlocorp/otc-install.git>

1. Crear directorio para XSLT, copiar archivos XSLT

mkdir /otc/xslt

1. Actualizar archivo yml con la configuración especifica del Banco.

vim /otc-core/otc-core\_1.yml

vim /otc-core/otc-core\_2.yml

vim /otc-admin/otc-admin.yml

**Propiedades que modificar en los archivos del archivo:**

otc-admin.yml

|  |  |
| --- | --- |
| otc:  idApplication | ID de Aplicación para framework de seguridad |
| otc:  userApplication | Usuario para framework de seguridad |
| otc:  fmUrl | URL del framework de seguridad |

otc-core.yml

|  |  |
| --- | --- |
| mq:  host: 172.16.11.16 | Servidor conexión MQ |
| mq:  port: 1416 | Puerto conexión MQ |
| mq:  queue-manager: CSPBRK\_QM | Gestor de Colas conexión MQ |
| mq:  channel: IVR\_CHL\_MF | Canal conexión MQ |
| mq:  queue:  request: TS\_OTC\_RECAUDACIONES\_REQ  replay: TS\_OTC\_RECAUDACIONES\_RESP | Canal conexión MQ |
| mq:  timeout: 50000 | Timeout conexión MQ |

1. Ejecutar Docker Compose garantizando su funcionalidad.

cd /otc/otc-install

docker-compose -f docker-otc-prod.yml build

docker-compose -f docker-otc-prod.yml up -d

Presionar Control+C (si no se presenta ningun error)

docker-compose down

docker stop $(docker ps -a -q)

docker rm $(docker ps -a -q)

1. Instalar OTC como servicio Docker.

docker stack deploy -c docker-otc-prod.yml otc