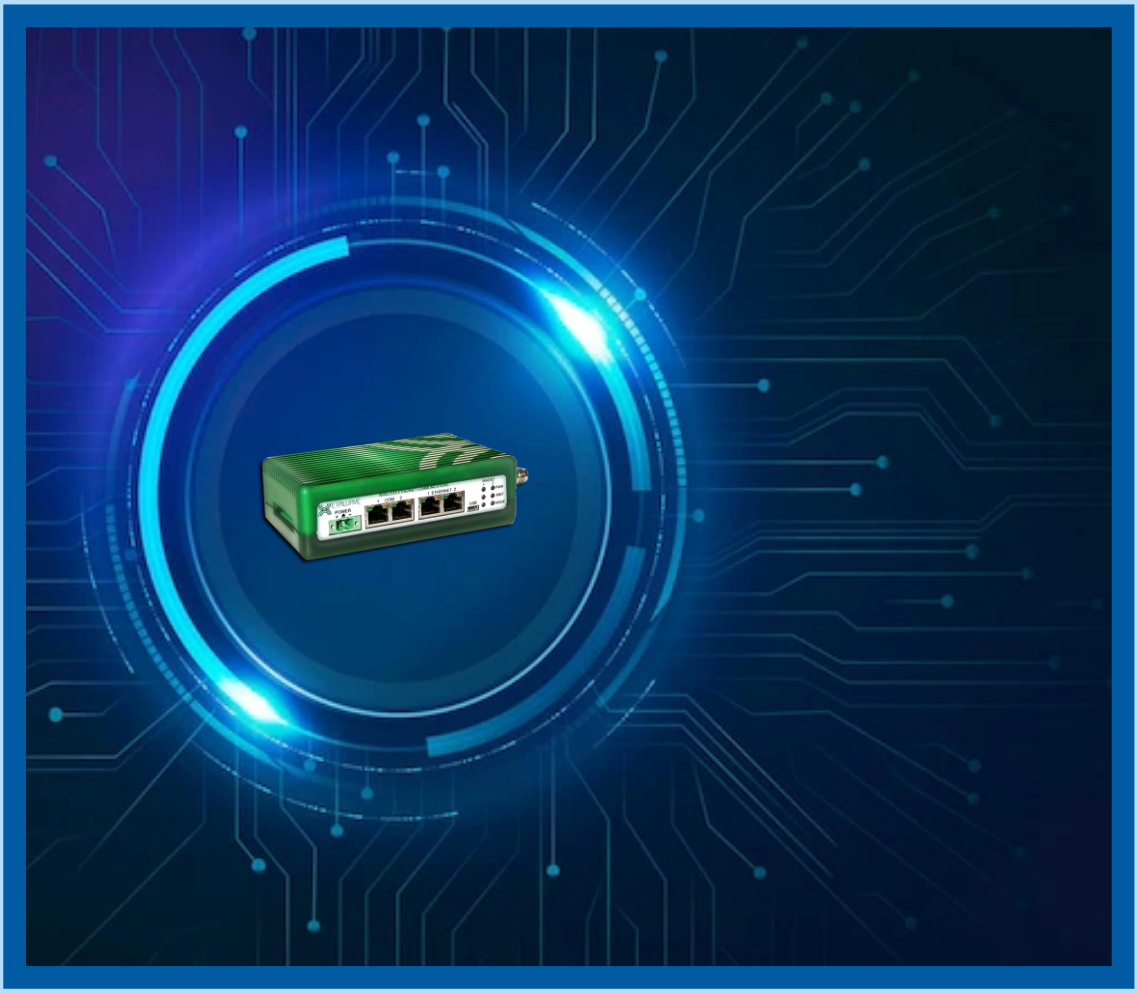
OAVV
OBSERVATORIO ARGENTINO DE VIGILANCIA VOLCÁNICA

SegemAR
Servicio Geológico Minero Argentino

PROCEDIMIENTO DE COMPILACIÓN E INSTALACIÓN DE SOFTWARE EARTHWORM EN RADIO DE TRANSMISIÓN DE DATOS XetaEdge9

INFORME TÉCNICO



Servicio Geológico Minero Argentino (SEGEMAR)

Buenos Aires 2022

Servicio Geológico Minero Argentino
Secretaría de Minería

Ministerio de Economía
Argentina

SegemAR
Servicio Geológico Minero Argentino

OAVV
OBSERVATORIO ARGENTINO
DE VIGILANCIA VOLCÁNICA

PROCEDIMIENTO
DE
COMPILACIÓN E INSTALACIÓN
DE SOFTWARE EARTHWORM
EN
RADIO DE TRANSMISIÓN DE DATOS
XetaEdge9

Noviembre 2022

PROCEDIMIENTO
DE
COMPILACIÓN E INSTALACIÓN
DE SOFTWARE EARTHWORM
EN
RADIO DE TRANSMISIÓN DE DATOS
XetaEdge9

Autores

Preatoni Victor [1]

Sebastian Garcia [1]

[1] *Observatorio Argentino de Vigilancia Volcánica (OAVV) – Servicio Geológico Minero Argentino (SEGEMAR).*

Buenos Aires, noviembre 2022

PROCEDIMIENTO DE COMPILACIÓN E INSTALACIÓN DE SOFTWARE EARTHWORM EN RADIO DE TRANSMISIÓN DE DATOS XetaEdge9

Contenido

PROCEDIMIENTO DE COMPILACIÓN E INSTALACIÓN DE SOFTWARE EARTHWORM EN RADIO DE TRANSMISIÓN DE DATOS XetaEdge9

1. INTRODUCCIÓN

1.1. MONITOREO VOLCÁNICO

Monitorear o vigilar un volcán implica estar atento a las señales que produce, analizarlas para conocer su comportamiento y tratar de pronosticar la ocurrencia de una erupción con la mayor certeza posible.

El monitoreo se realiza mediante la instalación de equipamiento específico sobre el volcán, y/o también utilizando algunos sensores alejados o remotos. Esto permite la observación continua y permanente, a través de diversos métodos visuales e instrumentales, de los distintos parámetros que caracterizan la dinámica interna del volcán. Estos parámetros se analizan a lo largo del tiempo con la finalidad de detectar oportunamente cambios en la actividad volcánica, y de ser posible, anticipar alguna condición anómala precursora de un proceso eruptivo.

Entre los tipos de monitoreo y vigilancia comúnmente utilizados en un volcán se encuentran la vigilancia visual, el monitoreo sísmico, el monitoreo geodésico, el monitoreo geoquímico y el monitoreo térmico.

Uno de los aspectos más importantes en el seguimiento y evaluación inmediata de los fenómenos volcánicos es contar con un sistema de comunicaciones que permita la centralización y acceso inmediato a todos los datos disponibles. Para ello, es necesario contar con un sistema de que posibilite la extracción de la información obtenida de los equipamientos instalados en los distintos volcanes, el almacenamiento y procesado de dicha información y finalmente la entrega de la información procesada a las autoridades correspondientes y a la población.

1.2. EARTHWORM

Earthworm (EW) es un sistema de adquisición, almacenamiento y procesamiento de señales sismológicas, el cual también permite la automatización de ciertos procesos como la detección y localización de eventos sísmicos. Su diseño es totalmente modular, y se compone de distintas piezas de software que interactúan entre si intercambiando mensajes a través de regiones de memoria compartida, llamadas anillos (RINGs).

Producto de su funcionamiento modular, esto hace que Earthworm sea un software altamente confiable y robusto, dado que cada proceso funciona de forma totalmente independiente del resto. Por ejemplo, el fallo de un proceso de adquisición de datos de un digitalizador no afectará al resto de los procesos (filtrado, procesado, alarmas, localización, etc).

A su vez, es un software fácilmente escalable, debido a que pueden agregarse o eliminarse módulos sin detener la ejecución del software, permitiendo esto no perder la operatividad del programa. Complementariamente, el software posibilita la implementación de nuevos desarrollos de código abierto, tales como NonLinLoc u otros algoritmos de localización y detección de ondas sísmicas.

1.3. XETAEDGE

XetaWave es una radio de transmisión de datos de tipo industrial. La misma tiene su origen de la empresa FreeWave, desarrolladora de radios industriales, con capacidad de recepción y transmisión de datos de diversos dispositivos (sensores, medidores, válvulas, etc) a través de largas distancias (más de 60 millas / 96 kilómetros) en entornos con visibilidad directa y bajo condiciones de instalación extremos.

Las radios industriales XetaWave están concebidas bajo un diseño moderno, utilizando la arquitectura SDR (Software Defined Radio). Es decir, en vez de utilizar un hardware específico para realizar la modulación de radiofrecuencia, esta se realiza por software. Esto permite una mayor flexibilidad en la radio, dado que es posible corregir fallos o actualizar los esquemas de modulación a medida que la tecnología avanza.

Dentro de la gama de productos que comercializa la empresa XetaWave, uno de estos productos denominado “XetaEdge”, posee además de una radio, una computadora industrial (ó Edge computer), cuya función es la de actuar como una “computadora de borde”, es decir, la primera línea de procesamiento a donde arribarán los datos.

XetaEdge9 está basada en un microprocesador ARM con sistema operativo Debian, el cual permite ejecutar aplicaciones Linux tradicionales.

2. OBJETIVO

El objetivo del trabajo es desarrollar una metodología para la implementación del software Earthworm, para su funcionamiento dentro de las radios XetaEdge9, con el objeto de centralizar la captura de datos de los equipos sismológicos de un volcán en su estación nodo, permitiendo esto robustecer las redes de monitoreo volcánico instrumental del Observatorio Argentino de Vigilancia volcánica (OAVV) del SEGEMAR.

De esta manera, la radio XetaEdge permitirá el funcionamiento de un Earthworm local, configurado para adquirir datos de las estaciones sismológicas de la red del volcán, y luego a través de un vínculo SeedLink, enviar todos los datos juntos a un Earthworm principal, localizado en los servidores del observatorio, a cientos de kilómetros de distancia.

A través de esta propuesta, se optimizaría el ancho de banda, dado que se establecerá una sola conexión TCP/IP entre el Earthworm del observatorio y el Earthworm corriendo en la XetaEdge en el nodo del volcán. Esto a su vez, simplifica la configuración del Earthworm del observatorio, dado que no es necesario crear un Module ID por cada estación, sino que se crea uno por cada volcán únicamente.

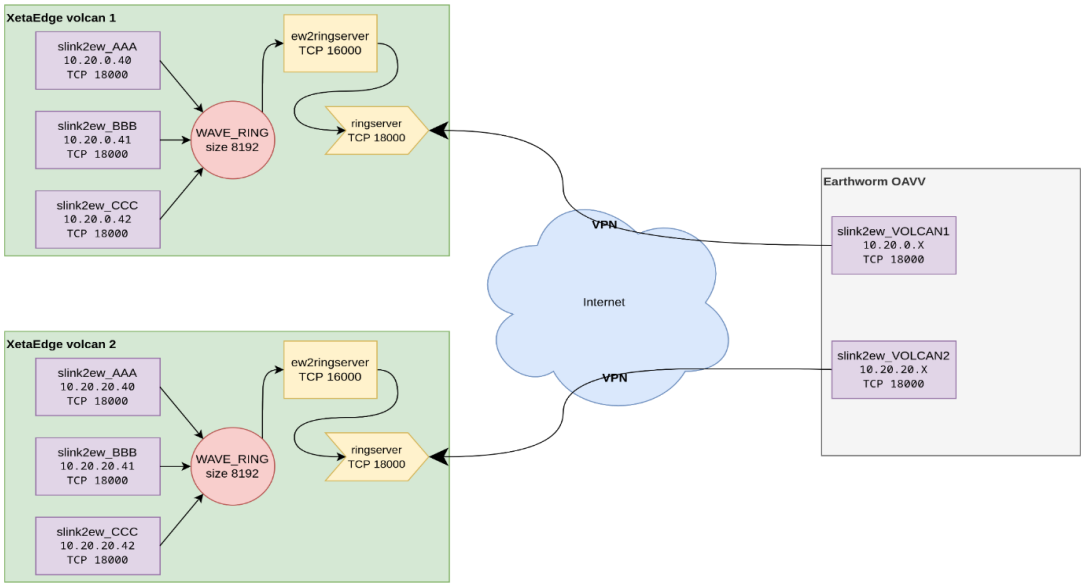


Figura 1: Topología típica de Earthworm remoto conectado a un Earthworm central.

3. CONEXIÓN POR SSH A LA XetaEdge

Para instalar el software Earthworm en la radio XetaEdge, es necesario conectarse a través de protocolo SSH. Previamente se deberá haber montado la maqueta de prueba con todos los elementos necesarios para alimentar la radio XetaEdge y poder efectuar una conexión IP, a saber:

- Notebook o PC de escritorio con placa Ethernet, configurada con una dirección IP fija dentro del rango *192.268.0.x/24.*
- Patchcord Cat5e o superior.
- Fuente de alimentación 12Vcc
- Radio XetaEdge9

Por defecto, las radios XetaEdge fabricadas luego de Marzo-2020 traen los siguientes parámetros de acceso:

IP: 192.168.0.3/24

Usuario: *debian*

Password: *temppwd*

Una vez alimentada la radio, para conectarse es posible utilizar algun software de terminal como el PuTTY o MobaXterm, o si se está trabajando en una PC bajo Linux, ejecutar directamente:

```
ssh debian@192.168.0.3
```

4. GENERACIÓN DE USUARIO PARA LA EJECUCIÓN DE EARTHWORM

El servicio del Earthworm debe ser ejecutado con permisos de usuario que le permitan funcionar correctamente en forma autónoma, pero que no permitan un acceso total al sistema. Para eso, crearemos un usuario cuya única función es ejecutar el servicio. Es decir, no podremos ingresar al sistema con ese usuario.

Ejecutar:

```
sudo useradd -s /usr/sbin/nologin -b /run -r earthworm
```

Agregar al usuario de Linux al grupo earthworm y viceversa:

```
sudo usermod -a -G earthworm $USER
```

```
sudo usermod -a -G $USER earthworm
```

Desloguearse de Linux con el comando "*exit*" y volver a loguearse:

```
ssh debian@192.168.0.3
```

5. DIRECTORIOS

Una vez dentro de la línea de comandos de la radio XetaEdge, crear una carpeta para la descarga e instalación de Earthworm en el directorio /opt:

```
sudo mkdir /opt/earthworm/
```

Asignar permisos para el usuario *earthworm*:

```
sudo chown earthworm:earthworm /opt/earthworm/
```

```
sudo chmod g+w /opt/earthworm/
```

Indicar al Bash que cambie a la nueva carpeta cada vez que nos logueamos al servidor:

```
echo "cd /opt/earthworm/" >> /home/$USER/.bashrc
```

Ir a la carpeta recién creada:

```
cd /opt/earthworm
```

Crear subcarpetas para la ejecución del programa:

```
mkdir -p run_working/log run_working/data/ring run_working/params
sudo chown -R earthworm:earthworm /opt/earthworm/
sudo chmod -R g+w /opt/earthworm/
```

6. INSTALACIÓN

6.1. DESCARGA

Finalmente, nuestro sistema está listo para proceder con la instalación. Descargar el código fuente de Earthworm con el siguiente comando:

```
git clone https://gitlab.com/seismic-software/earthworm.git
earthworm_git
```

6.2. COMPILACIÓN

Instalar los programas necesarios para compilar:

```
sudo apt install build-essential
```

Ir a la carpeta donde se descargó el código fuente:

```
cd earthworm_git/src/
```

Dado que el procesador y la memoria de la XetaEdge es muy limitada, editar el archivo *Makefile* para que solo se compilen los módulos estrictamente necesarios.

```
nano Makefile
```

Comentar las líneas que se ven a continuación, anteponiendo un **#,** y enviarlas al final de la sección SUBFOLDERS:

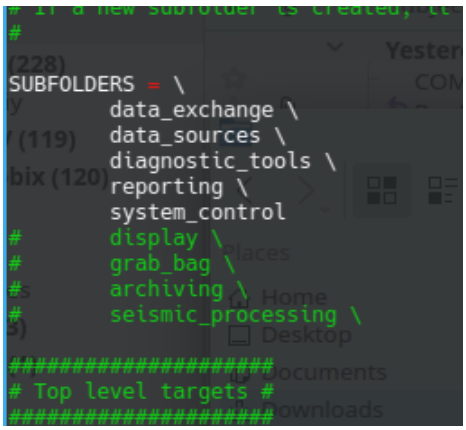


Figura 2: Sección SUBFOLDERS del archivo *Makefile.*

Aplicar las variables de entorno para poder compilar:

```
export EW_INSTALL_VERSION=earthworm_git
export EW_INSTALL_INSTALLATION=INST_xxxxxxxx
source ../environment/ew_linux.bash
make unix
```

Observar que la compilación termine correctamente.

Figura 3: Ejemplo de una compilación exitosa (No se observan mensajes de error).

6.3. *SYMLINK*

Dado que con el tiempo podrán surgir nuevas versiones de Earthworm, es conveniente crear un *symlink* a la versión actual, y luego referirse a ese *symlink* en todo momento. Si en algún momento se descarga una nueva versión, solo se deberá actualizar el *symlink* para que apunte a la nueva versión.

```
cd /opt/earthworm
ln -s earthworm_git earthworm_current
```

7. CONFIGURACIÓN INICIAL

Copiar los siguientes archivos:

```
cp earthworm_current/environment/earthworm.d run_working/params/
cp earthworm_current/environment/earthworm_global.d
run_working/params/
cp earthworm_current/environment/earthworm_commonvars.d
run_working/params/
cp earthworm_current/params/statmgr.* run_working/params/
cp earthworm_current/params/startstop_unix.d run_working/params/
cp earthworm_current/params/startstop.desc run_working/params/
```

Editar el archivo earthworm_commonvars.d:

```
nano run_working/params/earthworm_commonvars.d
```

```
#
# SetEnvVariable MYIPADDRESS 192.168.0.105
# SetEnvVariable EWLOGFILE 1
# SetEnvVariable STATIONFILE "${EW_PARAMS}/italy.hinv"
# SetEnvVariable MAINDIRWAVESERVER "/opt/waveserver_dir"
SetEnvVariable EW_LOG_MODE      2
SetEnvVariable EW_INST_ID       INST_OAVV
```

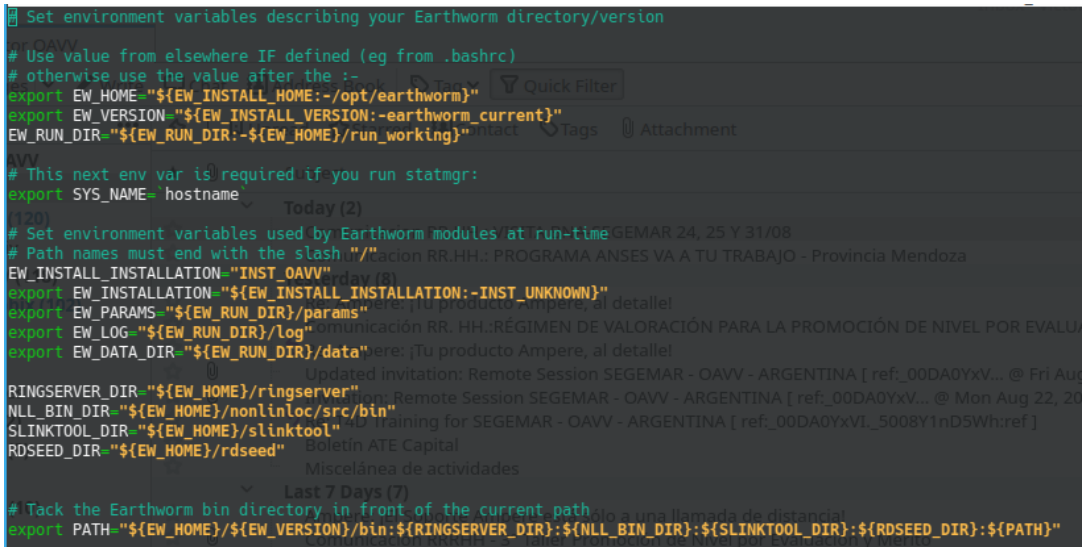
Figura 4: Ejemplo de archivo earthworm_commonvars.d

Agregar estas líneas al final:

```
SetEnvVariable EW_LOG_MODE      2
SetEnvVariable EW_INST_ID       INST_xxxxxxxxx
```

Crear el archivo ew_linux.bash:

```
nano run_working/params/ew_linux.bash
```



```
# Set environment variables describing your Earthworm directory/version
# Use value from elsewhere IF defined (eg from .bashrc)
# otherwise use the value after the :-
export EW_HOME="${EW_INSTALL_HOME:-/opt/earthworm}"
export EW_VERSION="${EW_INSTALL_VERSION:-earthworm_current}"
EW_RUN_DIR="${EW_RUN_DIR:-${EW_HOME}/run_working}"

# This next env var is required if you run statmgr:
export SYS_NAME=`hostname`

# Set environment variables used by Earthworm modules at run-time
# Path names must end with the slash "/"
EW_INSTALL_INSTALLATION="INST_OAVV"
export EW_INSTALLATION="${EW_INSTALL_INSTALLATION:-INST_UNKNOWN}"
export EW_PARAMS="${EW_RUN_DIR}/params"
export EW_LOG="${EW_RUN_DIR}/log"
export EW_DATA_DIR="${EW_RUN_DIR}/data"

RINGSERVER_DIR="${EW_HOME}/ringserver"
NLL_BIN_DIR="${EW_HOME}/nonlinloc/src/bin"
SLINKTOOL_DIR="${EW_HOME}/slinktool"
RDSEED_DIR="${EW_HOME}/rdseed"

# Tack the Earthworm bin directory in front of the current path
export PATH="${EW_HOME}/${EW_VERSION}/bin:${RINGSERVER_DIR}:${NLL_BIN_DIR}:${SLINKTOOL_DIR}:${RDSEED_DIR}:${PATH}"
```

Figura 5: Ejemplo de archivo ew_linux.bash

```
# Set environment variables describing your Earthworm directory/version
# Use value from elsewhere IF defined (eg from .bashrc)
# otherwise use the value after the :-
export EW_HOME="${EW_INSTALL_HOME:-/opt/earthworm}"
export EW_VERSION="${EW_INSTALL_VERSION:-earthworm_current}"
EW_RUN_DIR="${EW_RUN_DIR:-${EW_HOME}/run_working}"
# This next env var is required if you run statmgr:
export SYS_NAME=`hostname`
# Set environment variables used by Earthworm modules at run-time
# Path names must end with the slash "/"
EW_INSTALL_INSTALLATION="INST_xxxxxxxxx"
export EW_INSTALLATION="${EW_INSTALL_INSTALLATION:-INST_UNKNOWN}"
```

```
export EW_PARAMS="${EW_RUN_DIR}/params"
export EW_LOG="${EW_RUN_DIR}/log"
export EW_DATA_DIR="${EW_RUN_DIR}/data"
RINGSERVER_DIR="${EW_HOME}/ringserver"
# Tack the Earthworm bin directory in front of the current path
export PATH="${EW_HOME}/${EW_VERSION}/bin:${RINGSERVER_DIR}:${PATH}"
```

Vincular el archivo creado para que se cargue automáticamente al iniciar sesión en Linux:

```
sudo ln -s /opt/earthworm/run_working/params/ew_linux.bash
/etc/profile.d/ew_linux.sh
```

Editar el archivo earthworm.d y agregar los **Module** que sean necesarios de acuerdo a la cantidad de estaciones que tenga el volcán. Agregar también el módulo *MOD_EW2RINGSERVER*.

```
nano run_working/params/earthworm.d
```

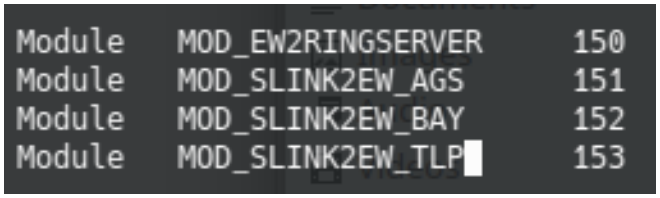


Figura 6: Ejemplo de archivo earthworm.d

Editar el archivo statmgr.d:

```
nano run_working/params/statmgr.d
```

Modificar la línea LogFile por la siguiente:

```
LogFile    ${EW_LOG_MODE}
```

Luego, al final del archivo, deben estar las siguientes líneas:

```
##################################
Descriptor  statmgr.desc
Descriptor  startstop.desc
##################################
```

Editar el archivo startstop_unix.d:

```
nano run_working/params/startstop_unix.d
```

Modificar la línea LogFile por la siguiente:

```
LogFile             ${EW_LOG_MODE}
```

Finalmente, en la parte inferior del archivo, definir los procesos que se van a iniciar cuando arranque Earthworm. Por el momento solo se inician procesos básicos. Luego a medida que se agreguen módulos, se irán agregando más líneas al final del archivo.

```
#
Process           "copystatus WAVE_RING STATUS_RING"
Class/Priority     OTHER 0
#
Process           "statmgr statmgr.d"
Class/Priority     OTHER 0
#
##################################################
```

8. SCRIPT DE INICIO AUTOMÁTICO

A través de *systemd*, Linux provee un sistema para el inicio y control de los procesos que deben correr en segundo plano. Crearemos el siguiente archivo:

```
sudo nano /etc/systemd/system/earthworm.service
```

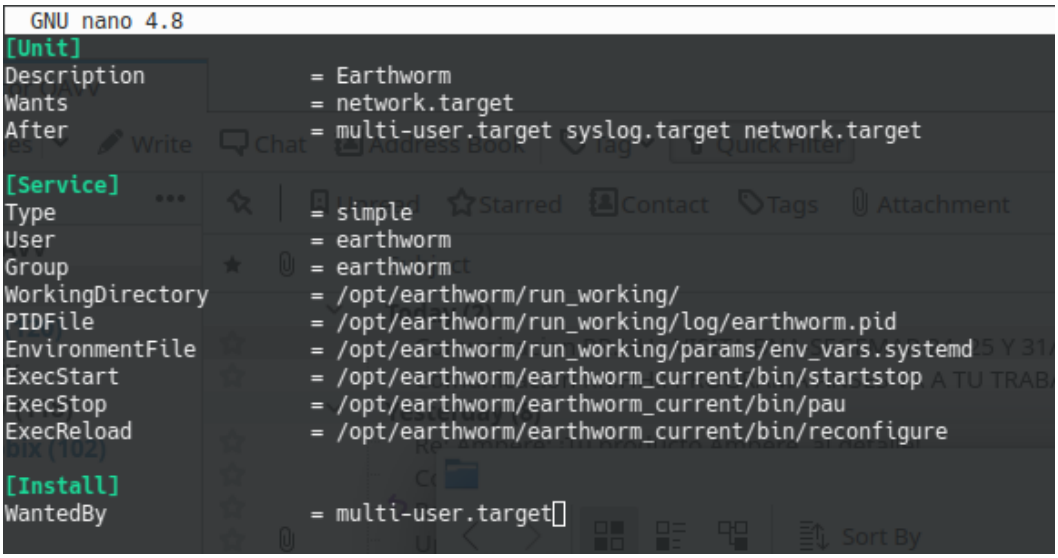


Figura 7: Ejemplo de archivo earthworm.service

El contenido del mismo es:

```
[Unit]
Description              = Earthworm
Wants                    = network.target
After                   = multi-user.target syslog.target network.target

[Service]
Type                     = simple
User                     = earthworm
```

```
Group                    = earthworm
WorkingDirectory         = /opt/earthworm/run_working/
PIDFile                  = /opt/earthworm/run_working/log/earthworm.pid
EnvironmentFile                                                    =
/opt/earthworm/run_working/params/env_vars.systemd
ExecStart              = /opt/earthworm/earthworm_current/bin/startstop
ExecStop                 = /opt/earthworm/earthworm_current/bin/pau
ExecReload                                                         =
/opt/earthworm/earthworm_current/bin/reconfigure
[Install]
WantedBy                 = multi-user.target
```

Guardar los cambios y ejecutar:

```
sudo systemctl enable earthworm
```

Crear también el siguiente archivo:

nano /opt/earthworm/run_working/params/env_vars.systemd

Contenido del archivo:

```
EW_HOME=/opt/earthworm
EW_RUN_DIR=/opt/earthworm/run_working
EW_PARAMS=/opt/earthworm/run_working/params
EW_DATA_DIR=/opt/earthworm/run_working/data
EW_LOG=/opt/earthworm/run_working/log
EW_VERSION=earthworm_current
EW_INSTALLATION=INST_xxxxxxxxx
SYS_NAME=earthworm
PATH=$PATH:/opt/earthworm/earthworm_current/bin:/opt/earthworm/ringse
rver
```

Reiniciar la Xeta9 con el comando:

```
sudo reboot
```

Finalmente ejecutar los siguientes comandos para verificar que la instalación básica de Earthworm quedó funcionando:

```
sudo systemctl status earthworm
```

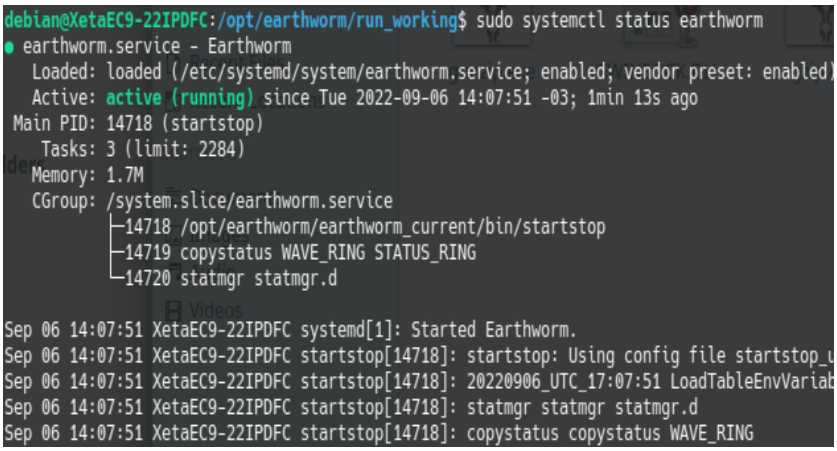


Figura 8: systemctl status de earthworm ejecutandose

```
status
```

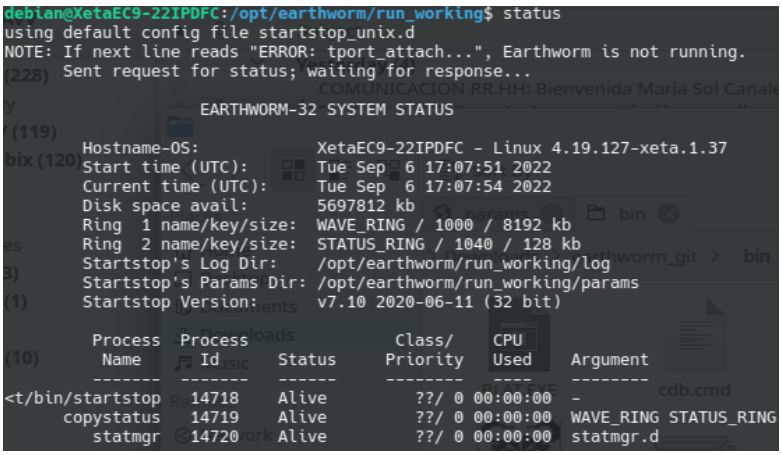


Figura 9: Modulos de earthworm ejecutándose.

9. LIMPIEZA AUTOMÁTICA DE ARCHIVOS

Earthworm genera gran cantidad de archivos de logs. Es necesario configurar una rutina de limpieza diaria.

Crear un archivo *ew_cleanup* en la carpeta *run_working/params:*

```
nano run_working/params/ew_cleanup
```

Contenido del archivo:

```
#!/bin/bash
# Script de bash para limpiar carpetas
# del Earthworm
#
# Propiedad de OAVV - SEGEMAR - 2022
#
```

```
# Load env variables
source /opt/earthworm/run_working/params/ew_linux.bash
# Delete older log files
find ${EW_LOG}/ -mtime +4 -type f -delete
```

Dar permisos de ejecución al archivo:

```
chmod +x run_working/params/ew_cleanup
```

Crear un enlace en cron.daily para que se ejecute diariamente:

```
sudo ln -s /opt/earthworm/run_working/params/ew_cleanup
```

10. SEED-LINK SERVER

El standard para la transmisión de datos sísmicos en tiempo real es el protocolo internacional "SeedLink". Dicho protocolo es compatible con diferentes sistemas, como ser SeisComp o Raspberry Shake, entre otros.

Earthworm permite la instalación de dicho módulo, para así compartir los datos adquiridos mediante SeedLink.

Descargar el código fuente con el siguiente comando:

```
git clone https://github.com/iris-edu/ringserver.git

cd ringserver
```

Compilar con el comando:

```
make

cd ..
```

Si la compilación fue exitosa, al ejecutar el comando *ringserver*, debería arrojar el número de versión:

```
ringserver version: 2020.075
```

Crear el archivo de configuración:

```
nano run_working/params/ringserver.d
```

Contenido del archivo:

```
# Example ringserver configuration file.
#
# Default values are in comments where appropriate.
#
```

```
# Dynamic parameters: some parameters will be re-read by ringserver
# whenever the configuration file is modified.
#
# Comment lines begin with a '#' or '*' character.


# Specify the directory where the ringserver will store
# the packet and stream buffers.  This must be specified.
# DO NOT use environment variables!!!

RingDirectory /opt/earthworm/run_working/data/ring


# Specify the ring packet buffer size in bytes.  A trailing
# 'K', 'M' or 'G' may be added for kilo, mega or giga bytes.

#RingSize 1G


# Specify the maximum packet ID.  The maximum ID for SeedLink
# is 16,777,215 (2^16).

#MaxPacketID 16777215


# Specify the maximum packet data size in bytes.

#MaxPacketSize 512


# Listen for connections on a specified port.  By default all supported
# protocols and network protocol families (IPv4 and IPv6) are allowed
and
# optional flags can be used to limit to specified protocols/families.
# Protocol flags are specified by including "DataLink", "SeedLink"
# and/or "HTTP" after the port.  Network families are specified by
# including "IPv4" or "IPv6" after the port, default is any supported
# by the system.  For example:
# ListenPort <port> [DataLink] [SeedLink] [HTTP] [IPv4] [IPv6]
#  This parameter can be specified multiple times to listen for
connections
# on multiple ports.

#ListenPort 18000


# Listen for DataLink connections on a specified port.  This is an
alias
# for a ListenPort configured with only DataLink allowed.

DataLinkPort 16000


# Listen for SeedLink connections on a specified port. This is an alias
# for a ListenPort configured with only SeedLink allowed.

SeedLinkPort 18000
```

```
# Specify the Server ID as reported to the clients.  The parameter may
# be a quoted string including spaces.  Default is "Ring Server".
# This is a dynamic parameter.

ServerID "OAVV-SEGEMAR SeedLink Server"

# Specify the level of verbosity for the server log output.  Valid
# verbosity levels are 0 - 3.  This is a dynamic parameter.

#Verbosity 3

# Specify the maximum number of clients per IP address, regardless of
# protocol, allowed to be connected concurrently.  This limit does
# not apply to addresses with write permission.  Set to 0 for unlimited.
# This is a dynamic parameter.


MaxClientsPerIP 5

# Specify the maximum number of clients, regardless of protocol,
# allowed to be connected simulataneously, set to 0 for unlimited.
# This is a dynamic parameter.

MaxClients 30

# Specify a timeout in seconds after which to drop client connections
# that have exchanged no packets with the server within the timeout
# window, set to 0 to disable.  This is a dynamic parameter.

ClientTimeout 120

# Control the usage of memory mapping of the ring packet buffer.  If
# this parameter is 1 (or not defined) the packet buffer will be
# memory-mapped directly from the packet buffer file, otherwise it
# will be stored in memory during operation and only read/written
# to/from the packet buffer file during startup and shutdown.
# Normally memory mapping the packet buffer is the best option,
# this parameter allows for operation in environments where memory
# mapping is slow or not possible (e.g. NFS storage).

#MemoryMapRing 1

# Control auto-recovery after corruption detection.  Be default if
# corruption is detected in the ring packet buffer file or stream
# index file during initialization the ring and stream files will be
# renamed with .corrupt extensions and initialization will be
# attempted a 2nd time.  If this option is 0 (off) the server will
# exit on these corruption errors.  If this option is 1 (the default)
# the server will move the buffers to .corrupt files.  If this option
# is 2 (delete) the server will delete the corrupt buffer files.

AutoRecovery 2

# Control reverse DNS lookups to resolve hostnames for client IPs.
# By default a reverse lookup is performed whenever a client connects.
# When a reverse DNS lookup fails a small delay will occur, this can
# be avoided by setting this option to 0 (off).
# This is a dynamic parameter.

ResolveHostnames 0
```

Copiar los archivos de configuración de *ew2ringserver*:

```
cp earthworm_current/params/ew2ringserver.* run_working/params/
```

11. ESTACIONES

La adquisición de datos desde las estaciones se hacen también bajo el protocolo SeedLink, pero en este caso, en modo cliente. En Earthworm, el módulo que realiza dicha función se llama slink2ew.

Copiar el template del módulo *slink2ew.d* y *.desc* a la carpeta *in*. Anexar el nombre de la estación (ej.: _AGS):

```
cp earthworm_current/params/slink2ew.d
run_working/params/slink2ew_AGS.d
```

```
cp earthworm_current/params/slink2ew.desc
run_working/params/slink2ew_AGS.desc
```

Editar el archivo *.desc* y modificar el *modId* por uno nuevo:

```
nano run_working/params/slink2ew_AGS.desc
```

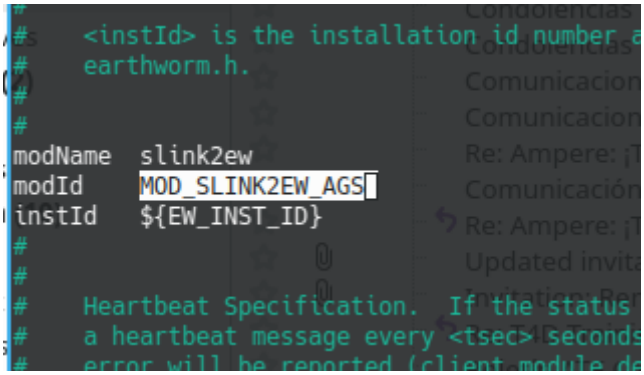


Figura 9: definición de modId en archivo .desc

Nota: Se deberá crear un module ID distinto por cada módulo similar que se utilice. Por ejemplo, si se utilizan 5 módulos slink2ew para adquirir datos de 10 estaciones distintas, se deben crear 5 modId diferentes.

También se deben definir los module ID en la configuración general de Earthworm. Editar:

```
nano run_working/params/earthworm.d
```

Definir los module y asignar un número único a cada uno:

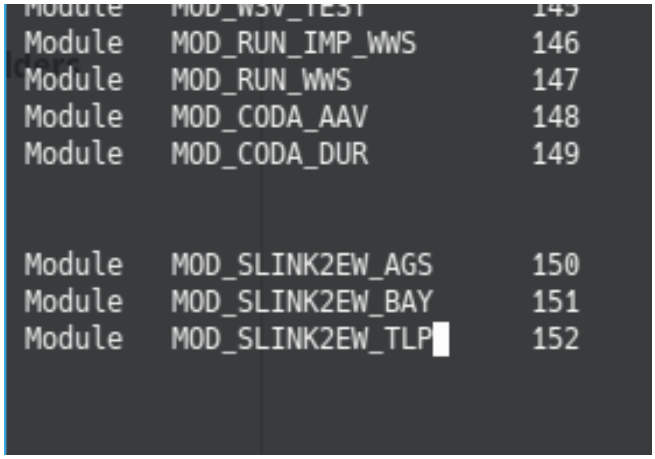


Figura 10: definición de Module en earthworm.d

Editar el archivo slink2ew_AGS.d para definir la IP del equipo al cual me voy a conectar para tomar datos:

```
nano run_working/params/slink2ew_AGS.d
```

Los parámetros más importantes son el *MyModuleId*, *SLhost*, *SLport* y *Selectors*.

- Se debe usar el mismo module ID configurado en el archivo. desc y definido en *earthworm.d*.
- En la línea *Selectors* es importante indicar que solo queremos leer los canales de datos. Eso se hace indicando *Selectors "HH?.**D**"*. Asegurarse de que no haya otra línea que diga “Stream” en otra parte del archivo. Si la hay, comentarla.

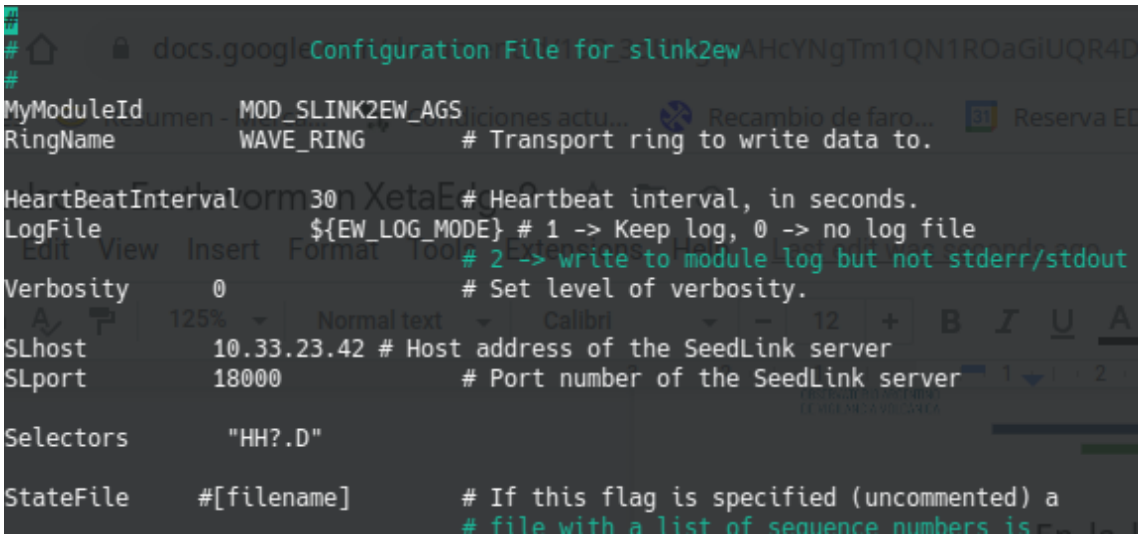


Figura 12: Ejemplo de archivo slink2ew.d para un digitalizador Guralp de 3 canales (HHZ, HHN, HHE)

Editar el archivo *statmgr.d* para monitorear el estado del módulo:

```
nano run_working/params/statmgr.d
```

```
#
##################################
Descriptor   statmgr.desc
Descriptor   startstop.desc
Descriptor   ew2ringserver.desc
Descriptor   slink2ew_AGS.desc
Descriptor   slink2ew_BAY.desc
Descriptor   slink2ew_TLP.desc
##################################
```

Figura 13: Ejemplo de archivo statmgr.d

Finalmente, editar el archivo *startstop_unix.d* para indicar que inicie el módulo creado recientemente cada vez que inicia Earthworm:

```
nano run_working/params/startstop_unix.d
```

```
#
Process           "ringserver ringserver.d"
Class/Priority     OTHER 0
#
Process           "slink2ew slink2ew_AGS.d"
Class/Priority     OTHER 0
#
Process           "slink2ew slink2ew_BAY.d"
Class/Priority     OTHER 0
#
Process           "slink2ew slink2ew_TLP.d"
Class/Priority     OTHER 0
#
```

Figura 14: archivo startstop_unix.d

Reiniciar Earthworm:

```
sudo systemctl restart earthworm
```

Verificar que los módulos hayan iniciado correctamente:

```
status
```

```
        Startstop Version:      v7.10 2020-06-11 (32 bit)

         Process  Process                Class/    CPU
          Name      Id      Status     Priority   Used    Argument
         -------  -------   ------     --------   ----    --------
<t/bin/startstop      832    Alive         ??/ 0 00:00:00  -
      copystatus      833    Alive         ??/ 0 00:00:00  WAVE_RING STATUS_RING
         statmgr      834    Alive         ??/ 0 00:00:00  statmgr.d
   ew2ringserver      835    Alive         ??/ 0 00:00:00  ew2ringserver.d
      ringserver      836    Alive         ??/ 0 00:00:00  ringserver.d
        slink2ew      837    Alive         ??/ 0 00:00:00  slink2ew_AGS.d
        slink2ew      838    Alive         ??/ 0 00:00:00  slink2ew_BAY.d
        slink2ew      839    Alive         ??/ 0 00:00:00  slink2ew_TLP.d
```

Figura 15: Ejemplo de estado de los módulos de adquisición y exportación de datos.

Verificar con *sniffwave* que están llegando datos de las estaciones:

```
debian@XetaEC9-22IPDFC:/opt/earthworm$ sniffwave WAVE_RING
Sniffing WAVE_RING for wild.wild.wild.wild
sniffwave: inRing flushed 207 packets of 354376 bytes total.
  TLP.HHZ.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:18.59 (1662739458.5900) 2022/09/09 16:04:22.70 (1662739462.7000) 0x00 0x00 i186 m153 t19 len1712
  TLP.HHZ.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:22.71 (1662739462.7100) 2022/09/09 16:04:26.82 (1662739466.8200) 0x00 0x00 i186 m153 t19 len1712
  AGS.HHZ.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:19.52 (1662739459.5200) 2022/09/09 16:04:23.63 (1662739463.6300) 0x00 0x00 i186 m151 t19 len1712
  AGS.HHZ.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:23.64 (1662739463.6400) 2022/09/09 16:04:27.75 (1662739467.7500) 0x00 0x00 i186 m151 t19 len1712
  AGS.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:17.76 (1662739457.7600) 2022/09/09 16:04:21.87 (1662739461.8700) 0x00 0x00 i186 m151 t19 len1712
  AGS.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:21.88 (1662739461.8800) 2022/09/09 16:04:25.99 (1662739465.9900) 0x00 0x00 i186 m151 t19 len1712
  AGS.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:26.00 (1662739466.0000) 2022/09/09 16:04:30.11 (1662739470.1100) 0x00 0x00 i186 m151 t19 len1712
  AGS.HHE.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:20.34 (1662739460.3400) 2022/09/09 16:04:24.45 (1662739464.4500) 0x00 0x00 i186 m151 t19 len1712
  AGS.HHE.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:24.46 (1662739464.4600) 2022/09/09 16:04:28.57 (1662739468.5700) 0x00 0x00 i186 m151 t19 len1712
  TLP.HHE.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:23.48 (1662739463.4800) 2022/09/09 16:04:27.59 (1662739467.5900) 0x00 0x00 i186 m153 t19 len1712
  TLP.HHE.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:27.60 (1662739467.6000) 2022/09/09 16:04:31.71 (1662739471.7100) 0x00 0x00 i186 m153 t19 len1712
  BAY.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:22.39 (1662739462.3900) 2022/09/09 16:04:26.50 (1662739466.5000) 0x00 0x00 i186 m152 t19 len1712
  BAY.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:26.51 (1662739466.5100) 2022/09/09 16:04:30.62 (1662739470.6200) 0x00 0x00 i186 m152 t19 len1712
  BAY.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:30.63 (1662739470.6300) 2022/09/09 16:04:34.74 (1662739474.7400) 0x00 0x00 i186 m152 t19 len1712
  TLP.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:22.34 (1662739462.3400) 2022/09/09 16:04:26.45 (1662739466.4500) 0x00 0x00 i186 m153 t19 len1712
  BAY.HHZ.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:23.54 (1662739463.5400) 2022/09/09 16:04:27.65 (1662739467.6500) 0x00 0x00 i186 m152 t19 len1712
  BAY.HHZ.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:27.66 (1662739467.6600) 2022/09/09 16:04:31.77 (1662739471.7700) 0x00 0x00 i186 m152 t19 len1712
  BAY.HHZ.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:31.78 (1662739471.7800) 2022/09/09 16:04:35.89 (1662739475.8900) 0x00 0x00 i186 m152 t19 len1712
  TLP.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:26.46 (1662739466.4600) 2022/09/09 16:04:30.57 (1662739470.5700) 0x00 0x00 i186 m153 t19 len1712
  TLP.HHN.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:30.58 (1662739470.5800) 2022/09/09 16:04:34.69 (1662739474.6900) 0x00 0x00 i186 m153 t19 len1712
  BAY.HHE.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:25.56 (1662739465.5600) 2022/09/09 16:04:29.67 (1662739469.6700) 0x00 0x00 i186 m152 t19 len1712
  BAY.HHE.VV.04 (0x32 0x30) 0 i4 412 100.0 2022/09/09 16:04:29.68 (1662739469.6800) 2022/09/09 16:04:33.79 (1662739473.7900) 0x00 0x00 i186 m152 t19 len1712
```

Figura 16: Ejemplo de salida de sniffwave adquiriendo datos en tiempo real.

12. CONCLUSIONES

Se logró la correcta implementación del software Earthworm dentro de una radio XetaEdge9, permitiendo de esta manera la centralización en la adquisición de datos sismológicos en un sistema portable e instalable en cualquier nodo central de una red de monitoreo volcánico.

Una vez implementado esto, se logró la conectividad con el Earthworm principal del Observatorio Argentino de Vigilancia Volcánica (OAVV) del SEGEMAR.

Fue posible constatar la optimización de recursos en la transferencia de datos entre Earthworms, disminuyendo el ancho de banda de transferencia.

Teniendo en consideración lo anteriormente mencionado, fue posible concluir de forma satisfactoria las pruebas realizadas para la implementación de este nuevo sistema de adquisición de datos sismológicos que posibilitará la optimización de recursos, y robustecerá las capacidades de las redes de monitoreo del OAVV.

13. REFERENCIAS

- Earthworm Documentation V7.10 (2019). http://folkworm.ceri.memphis.edu/ew-doc/
- SeedLink Server. https://github.com/iris-edu/ringserver
- XetaWave. https://www.xetawave.com/products
- Freewave. https://www.freewave.com/products/fgr2-series/
- Terminal PuTTY. https://www.putty.org/
- SeisComP. https://www.seiscomp.de/
- RaspberryShake. https://raspberryshake.org/