USER MANUAL

RTPSdump

Contenido

[1. Overview 2](#_Toc270935119)

[2. Uso 2](#_Toc270935120)

[3. Base de datos 2](#_Toc270935121)

[3.1. Tabla de typecodes 2](#_Toc270935122)

[3.2. Tabla de entidades 3](#_Toc270935123)

[3.3. Tabla de mensajes de descubrimiento 3](#_Toc270935124)

[3.4. Tabla de datos por tópico 4](#_Toc270935125)

[4. Tipos de datos 4](#_Toc270935126)

[4.1. Tipos simples 4](#_Toc270935127)

[4.2. Arrays 5](#_Toc270935128)

[4.3. Secuencias 6](#_Toc270935129)

[4.4. Estructuras 6](#_Toc270935130)

[4.5. Uniones 7](#_Toc270935131)

# Overview

RTPSdump es una aplicación que vuelca información sobre los mensajes RTPS que se han enviado por la red en una base de datos SQLITE. Previamente estos mensajes RTPS deben haber sido capturados por un sniffer, por ejemplo el sniffer Wireshark, y guardados en un fichero PCAP, formato que entiende la aplicación RTPSdump.

Dentro de los mensajes RTPS que pueden ser enviados, RTPSdump analiza los mensajes de descubrimiento de *datawriters* y *datareaders*, además de los mensajes de datos que se envían entre estas entidades. Los mensajes de descubrimiento contienen información del tópico que usa cada *datawriter* y *datareader*. A través del *typecode* del tópico RTPSdump puede deserializar los mensajes de datos que se envían entre *datawriters* y  *datareaders.*

# Uso

RTPSdump se ejecuta a través de línea de comandos:

*RTPSdump.exe [--db base\_de\_datos] fichero\_pcap*

El comando *--db* sirve para asignarle un nombre a la base de datos SQLITE que usará la aplicación. Por defecto el nombre de la base de datos creada es *dump.db*.

# Base de datos

RTPSdump crea tres tablas predeterminadas en la base de datos, además de crear una tabla por cada tópico que se ha descubierto en las trazas. En este apartado se describen las diferentes tablas y sus campos.

## Tabla de typecodes

RTPSdump crea una tabla llamada *typecodes* donde guarda información sobre los tópicos que se han descubierto en el fichero de trazas. Los campos que tiene esta tabla son:

* **topic\_name:** El nombre del tópico.
* **type\_name:** El nombre del tipo de datos.
* **typecode:** Descripción del *typecode* del tipo de datos.

## Tabla de entidades

RTPSdump crea una tabla llamada *entities* donde guarda información de los *datawriters* y *datareaders* que se han descubierto en el fichero de trazas. Los campos que tiene esta tabla son:

* **host\_id, app\_id, instance\_id, entity\_id:** estos 4 campos definen un identificador único para cada entidad.
* **type:** determina si la entidad es un *datareader* o un *datawriter*.
* **topic\_name:** nombre del tópico.
* **type\_name:** nombre de tipo de datos.
* **exists\_typecode:** este campo está a ‘1’ si al encontrar la entidad está tenía información del typecode, o el valor ‘0’ en caso contrario.

## Tabla de mensajes de descubrimiento

RTPSdump crea una tabla llamana *entities\_topic\_messages* donde guarda todos los mensajes de descubrimiento que se han encontrado en el fichero de trazas. Cada entrada de esta tabla es un mensaje de descubrimiento. Los campos que tiene esta tabla son:

* **wireshark\_timestamp\_sec, wireshark\_timestamp\_usec:** estos dos campos definen el timestamp del momento en que el wireshark capture el mensaje de descubrimiento.
* **ip\_src:**IP desde la que se envió el mensaje de descubrimiento.
* **ip\_dst:** IP a la que va dirigido el mensaje de descubrimiento.
* **host\_id, app\_id, instance\_id, reader\_id, writer\_id:** identificadores únicos del *datawriter* de la aplicación que envía el mensaje de descubrimiento y del *datareader* de la aplicación que puede recibir los mensajes de descubrimiento.
* **writer\_seq\_num:** número de secuencia del mensaje de descubrimiento.
* **sourcetimestamp\_sec, sourcetimestamp\_nanosec:** es el timestamp del mensaje de descubrimiento con respecto al instante en que fue enviado.
* **dest\_host\_id, dest\_app\_id, dest\_instance\_id:** identificador único del participante al que se envía el mensaje de descubrimiento. Estos campos pueden estar vacios en caso de que el mensaje de descubrimiento no vaya dirigido solo a un participante.
* **entity\_host\_id, entity\_app\_id, entity\_instance\_id, entity\_id:** identificador único del *datawriter* o *datareader* del que tiene información el mensaje de descubrimiento.
* **type:** determina si la entidad de la que tiene información el mensaje de descubrimiento es un *datareader* o un *datawriter*.
* **topic\_name:** nombre del tópico que usa la entidad.
* **type\_name:** nombre de tipo de datos que usa la entidad
* **exists\_typecode:** este campo está a ‘1’ si al analizar el mensaje de descubrimiento tenía información del typecode, o el valor ‘0’ en caso contrario.

## Tabla de datos por tópico

RTPSdump crea una tabla por cada tópico encontrado al analizar el fichero de trazas. Cada entrada en la tabla es un mensaje de datos encontrado en el fichero de capturas. Estás tablas tienen campos en común. El resto de campos se crean en base a la estructura del tipo de datos del tópico. En esta sección se describen los campos comunes:

* **wireshark\_timestamp\_sec, wireshark\_timestamp\_usec:** estos dos campos definen el timestamp del momento en que el wireshark capture el mensaje de datos.
* **ip\_src:**IP desde la que se envió el mensaje de datos.
* **ip\_dst:** IP a la que va dirigido el mensaje de datos.
* **host\_id, app\_id, instance\_id, reader\_id, writer\_id:** identificadores únicos del *datawriter* de la aplicación que envía el mensaje de datos y del *datareader* de la aplicación que puede recibir los mensajes de datos.
* **writer\_seq\_num:** número de secuencia del mensaje de datos.
* **sourcetimestamp\_sec, sourcetimestamp\_nanosec:** es el timestamp del mensaje de datos con respecto al instante en que fue enviado.
* **dest\_host\_id, dest\_app\_id, dest\_instance\_id:** identificador único del participante al que se envía el mensaje de datos. Estos campos pueden estar vacios en caso de que el mensaje de datos no vaya dirigido solo a un participante.

# Tipos de datos

En este apartado se describirán los tipos de datos que puede tener un tópico y que son soportados por la aplicación RTPSdump y como se guardan en la base de datos.

## Tipos simples

Los tipos simple que soporta RTPSdump son guardados como un campo más en la tabla del tópico. El nombre del campo será el nombre de tipo dentro del tópico. Por ejemplo para el tipo ‘long contador’, el campo creado se llamará ‘contador’.

|  |  |
| --- | --- |
| Tipo simple | Tipo del campo en SQLITE |
| Octet | TINYINT |
| Short | SMALLINT |
| Unsigned short | SMALLINT UNSIGNED |
| Long | INT |
| Unsigned long | INT UNSIGNED |
| Long long | BIGINT |
| Unsigned long long | BIGINT UNSIGNED |
| Char | CHARACTER(1) |
| String | TEXT |
| Float | FLOAT |
| Double | DOUBLE |
| Boolean | TINYINT |
| Enumeration | TEXT |

**Nota:** SQLITE no soporta datos ‘unsigned’. Esto implica que los datos unsigned, tales como ‘unsigned short’ o ‘unsigned long’, puedes ser mostrados como un número negativo si su bit más significativo es 1.

## Arrays

RTPSdump soporta arrays multidimensionales. Cuándo un array es detectado en el tópico se crea un campo en la tabla del tópico y se crea otra tabla aparte para guardar los datos que se lean del array. El campo creado en la tabla del tópico (*nombre\_del\_array\_ref*) contiene un entero que hace referencia a las entradas en la tabla del array donde se guardan los datos leídos del array.

Cada entrada de la tabla de un array es un dato único dentro de un array. La tabla creada para el array contiene un campo *ref* que hace referencia a una entrada de la tabla del tópico a la que pertenece el dato guardado. Los siguientes campos son los índices del dato dentro del array. Y el último campo es el valor del dato.

Por ejemplo, para un array ‘long example[10]’ en la tabla del tópico existirá un campo llamado *example\_ref* y se habrá creado otra tabla con los siguientes campos: *ref*, *index\_0*, *value*. Un mensaje de datos tiene su entrada en la tabla del tópico y su campo *example\_ref* tendrá un valor que se corresponde con diez entradas de la tabla del array que contienen los 10 valores del array para ese mensaje de datos.

Los tipos simple que son soportados por RTPSdump para un array son:

|  |  |
| --- | --- |
| Tipo simple para un array | Tipo del campo en SQLITE |
| Octet | TINYINT |
| Short | SMALLINT |
| Unsigned short | SMALLINT UNSIGNED |
| Long | INT |
| Unsigned long | INT UNSIGNED |
| Long long | BIGINT |
| Unsigned long long | BIGINT UNSIGNED |
| Char | CHARACTER(1) |
| Float | FLOAT |
| Double | DOUBLE |
| Boolean | TINYINT |
| Enumeration | TEXT |

## Secuencias

RTPSdump soporta secuencias. Cuándo una secuencia es detectada en el tópico se crea un campo en la tabla del tópico y se crea otra tabla aparte para guardar los datos que se lean de la secuencia. El campo creado en la tabla del tópico (*nombre\_de\_la\_secuencia\_ref*) contiene un entero que hace referencia a las entradas en la tabla de la secuencia donde se guardan los datos leídos de la secuencia.

Cada entrada de la tabla de una secuencia es un dato único dentro de una secuencia. La tabla creada para la secuencia contiene un campo *ref* que hace referencia a una entrada de la tabla del tópico a la que pertenece el dato guardado. Los siguientes campos es el indice del dato dentro de la secuencia. Y el último campo es el valor del dato.

Por ejemplo, para una secuencia ‘sequence<long> example’ en la tabla del tópico existirá un campo llamado *example\_ref* y se habrá creado otra tabla con los siguientes campos: *ref*, *index*, *value*. Un mensaje de datos tiene su entrada en la tabla del tópico y su campo *example\_ref* tendrá un valor que se corresponde con entradas de la tabla de la secuencia que contienen los valores de la secuencia para ese mensaje de datos.

Los tipos simple que son soportados por RTPSdump para una secuencia son:

|  |  |
| --- | --- |
| Tipo simple para un sequencia | Tipo del campo en SQLITE |
| Octet | TINYINT |
| Short | SMALLINT |
| Unsigned short | SMALLINT UNSIGNED |
| Long | INT |
| Unsigned long | INT UNSIGNED |
| Long long | BIGINT |
| Unsigned long long | BIGINT UNSIGNED |
| Char | CHARACTER(1) |
| Float | FLOAT |
| Double | DOUBLE |
| Boolean | TINYINT |
| Enumeration | TEXT |

## Estructuras

RTPSdump crea un campo dentro de la tabla del tópico por cada tipo que hay dentro de una estructura. El nombre del campo estará compuesto por el nombre de la estructura y el nombre del tipo. Por ejemplo para la estructura:

*struct ejemplo  
{  
 long contador;  
 string message;  
};*la tabla del tópico tendrá los campos: *ejemplo\_\_contador* y *ejemplo\_\_message*.

## Uniones

RTPSdump crea un campo dentro de la tabla del tópico por cada tipo que hay dentro de una estructura. El nombre del campo estará compuesto por el nombre de la estructura y el nombre del tipo. Por ejemplo para la estructura:

*union ejemplo  
{  
 case 1:  
 long contador;  
 case 2:  
 string message;  
};*la tabla del tópico tendrá los campos: *discriminator, ejemplo\_\_contador* y *ejemplo\_\_message*. Pero a diferencia de las estructuras el campo *discriminator* tiene el valor del discriminador de la unión y no todos los campos se rellenan en cada entrada de la tabla del tópico.