Programmer's Guide to the DULsnoop Extension of the DUL Facility

For DICOM Communications Monitoring

Nilesh R. Gohel

Mallinckrodt Institute of Radiology
Electronic Radiology Laboratory
510 South Kingshighway Boulevard
St. Louis, Missouri 63110
314/362-6965 (Voice)
314/362-6971 (FAX)

This document describes an extension to the DUL facility for monitoring DICOM associations.

Version 2.10.0

August 3, 1998

---

1 Introduction

This extension of the DUL facility provides a means to parse DICOM PDUs from a TCP data stream and gather the association's service parameters in a data structure known as DUL_ASSOCIATESERVICEPARAMETERS. The extension also tracks the DUL states of an association based on the types of DICOM PDUs encountered.

The application should use *DUL_FileSnoop* or *DUL_NetworkSnoop* depending on whether the input data stream is to be read from a file *DUL_FileSnoop*) or in real-time from the network (*DUL_NetworkSnoop*). In both cases, the DICOM PDUs are passed back to the application via callback functions registered with the facility extension using the *DUL_RegPDUCall* routine. One callback is registered for the PDUs in each direction (one for PDUs from initiator to acceptor and the other for PDUs from acceptor to initiator). Once called, *DUL_FileSnoop* will not return until the input datafiles have been completely parsed for DICOM PDUs. Simlarly, *DUL_NetworkSnoop* will continue parsing files until the required number of associations have been monitored. During this time, callbacks will be called with the DICOM PDUs as they are parsed out of the TCP data stream.

In the case of input from files (*DUL_FileSnoop*), two data files need to be provided. One contains data from the initiator to the acceptor, and the other contains data in the opposite direction. The headers for the data segments in the files should follow that provided in snp.h. Additional information such as initiator name and acceptor name are optional and are only used for reporting purposes.

The other case (*DUL_NetworkSnoop*) uses the SNP facility for real-time snooping on DICOM associations based on TCP/IP over shared media networks. In this case, additional information required by the SNP facility must be provided such as initiator name / address, acceptor name / address, and acceptor TCP port number to identify the association. Other required information required pertains to the network interface for the monitoring, buffer sizes to be used, and number of associations to monitor. Refer to the document *DICOM Test Tools: A Guide to Programs for Testing DICOM Functionality* for examples of facility extension usage in an application context.

2 Include Files

Any code that uses these DUL routines, structure definitions or constants should include the following files in the order given:

```
#include "dicom.h"
#include "condition.h"
#include "lst.h"
#include "snp.h"
#include "dulprotocol.h"
```

3 Return Values

The following returns are possible from the DULsnoop extension:

DUL_NORMAL	Normal return from DUL routine
DUL_SNPFILEOPEN	Error opening file
DUL_SNPCALLBACKUSE	Error using callback
DUL_SNPCALLBACKREG	Error registering callback
DUL_SNPINIT	SNP_Init failure
DUL_SNPPREMATUREEOF	Expecting more data from input files
DUL_SNPSTART	Failure of SNP_Start function
DUL_SNPSTOP	Failure of SNP_Stop function
DUL_SNPTERMINATE	Failure of SNP_Terminate
DUL_SNPNOTALLASSOC	SNP Error with remaining associations
DUL_SNPBADSTATE	SNP facility in bad state
DUL_SNPBADASSOCSTATE	Snooper transition into bad DUL association state

4 DUL Snoop Extension Routines

This section provides detailed documentation for each DULsnoop facility routine.

DUL_RegPDUCall

Name

DUL_RegPDUCall - registers callback functions for delivery of DICOM PDUs to calling entity.

Synopsis

CONDITION DUL_RegPDUCall(void (*callback) (), int callbackType, void *ctx)

callback	Callback function to register
callbackType	Type (direction) of callback function (ITOA or ATOI) as defined in snp.h
ctx	Pointer to context data

Description

DUL_RegPDUCall is used to register callback functions to pass DICOM PDUs to the application. Users should register one callback for PDUs in each direction (initiator to acceptor - ITOA, and acceptor to initiator - ATOI). The form of both callback functions should be:

CONDITION callback_func_name(int ini_state, int acc_state, u_char pdu_type,
char *buf, int len, DUL_ASSOCIATESERVICEPARAMETERS * params, void *ctx);

where:

ini_state	Initiator DUL state
acc_state	Acceptor DUL state
pdu_type	rovide the type of the PDU.

These are (as defined in dulprotocol.h):

```
DUL_TYPEASSOCIATERQ
DUL_TYPEASSOCIATEAC
DUL_TYPEASSOCIATERJ
DUL_TYPEDATA
DUL_TYPERELEASERQ
DUL_TYPERELEASERP
DUL_TYPEABORT
```

buf	Pointer to PDU
len	Entire length of PDU in bytes
params	Pointer to data structure containing DUL association parameters
ctx	Context pointer that is passed in callback

Return Values

DUL_NORMAL

DUL_FileSnoop

Name

DUL_FileSnoop - parses data for DICOM PDUs, gathers association service parameters, and tracks DUL states from files.

Synopsis

CONDITION DUL_FileSnoop(char *itoa_file, char *atoi_file, char *initiator, char *acceptor)

itoa_file	Path of file containing parsed TCP data frominitiator to acceptor.
atoi_file	Path of file containing parsed TCP data from acceptor to initiator.
initiator	(Optional) Name of initiator - for reporting purposes only.
acceptor	(Optional) Name of acceptor - for reporting purposes only.

Description

DUL_FileSnoop parses DICOM PDUs from input files containing data stream segments. This function tracks DUL states and association service parameters in the DUL_ASSOCIATESERVICEPARAMETERS data structure. Upon completion of parsing of a complete DICOM PDU, it is passed back to the application via registered callbacks. Refer to documentation on the *DUL_RegPDUCall* function for information on the form of the callbacks as well as how they may be registered.

Notes

Callbacks to pass back DICOM PDUs have to be registered before this routine may be called. Also, data segments in input files must be preceded by header as specified in snp.h.

Return Values

```
DUL_NORMAL
DUL_SNPFILEOPEN
DUL_SNPFILEREAD
DUL_SNPCALLBACKUSE
DUL_SNPBADASSOCSTATE
DUL_SNPPREMATUREEOF
```

DUL_NetworkSnoop

Name

DUL_NetworkSnoop - Monitors DICOM associations at the DUL level in real-time using SNP facility.

Synopsis

CONDITION DUL_NetworkSnoop(char *device, int ppa, char *initiator, char *acceptor, int port,
int bufsize, int associations)

device	hared media network device driver file name on which to be snooping e.g. Ethernet interface: “/dev/le”
ppa	Physical Point of Access (PPA) e.g. 0 for /dev/le0
initiator	Host name / address of DICOM communication initiator
acceptor	Host name / address of DICOM communication acceptor
port	TCP port number on acceptor that will be used
bufsize	Number of bytes of space used for chunks by STREAMS kernel buffer module
associations	Number of associations to be tracked

Description

DUL_NetworkSnoop monitors DICOM associations with given parameters in real-time. This function parses DICOM PDUs the TCP data stream from the underlying SNP facility. The function also tracks DUL states and association service parameters in the DUL_ASSOCIATESERVICEPARAMETERS data structure. DICOM PDUs are passed back via registered callbacks along with DUL state information and the DUL_ASSOCIATESERVICEPARAMETERS structure. Refer to documentation on the DUL_RegPDUCall function for information on the form of the callbacks as well as how they may be registered. The application is then free to print the information or treat the data in any way that is desired.

Notes

Callbacks to pass back DICOM PDUs have to be registered before this routine may be called. Requires SNP facility which must be able to run on machine. For shared media networks only.

Return Values

```
DUL_NORMAL
DUL_SNPCALLBACKUSE
DUL_SNPBADASSOCSTAT
```