

## 1. Purpose

Integration with Loftware is possible through the Loftware Sockets Interface. Loftware has a registered port with Internet Assigned Numbers Authority (IANA). The WatchDogNT interface to the Loftware Print Server (LPS) will listen for both Sock Stream and UDP on port 2723. All integration is done via a session (Sock Stream) with the Loftware Socket Server that is initiated on port 2723. All session based communication rides on the Loftware Session Protocol that is based on a Request - Response as well as an Unpended Response design. With the Synchronous interface, there is limited Unpended Responses sent.

Loftware only supports the TCP/IP protocol at this time. This document serves as a guide to the protocol basics as well as the integration methods. All mentions to 'Server' in this document refers to the Loftware Print Server.

## 2. Session Overview

There are three basic types of communications between a client and a server:

- 1) **Request** - A client *Requests* something of the server
- 2) **Response** - The server *responds* to the client's request (To include errors)
- 3) **Unpended Response** - Any information sent to a client from the server which was not expected (not an immediate response to a pending request).

In order to begin a session, the client must connect to the server. A connection is initiated by a call to a Berkley compliant connect() or a WSAConnect() for Win32.

The next step after connecting to the server is to issue a Login Request to allow the server to identify the type of connection as well as the connection's identifier (ComputerName). Once connected, the client is free to issue Requests of the server. During the session's lifetime, Keep Alive requests must be issued to the server during periods of inactivity. The server monitors the connection status of its connected clients, a client which has not generated any activity over a predefined period of time will be assumed invalid and will be closed. A graceful exit from the server involves issuing a Logout request, and following the provider-specific steps to gracefully shutdown a socket.

Please note that as of release 5.5 Service Pack 11 of LPS there are two new commands added which allow for the login and sending of the print request in one request. Systems that are unable to maintain a connection to the Server should utilize this new functionality to limit I/O overhead.

### 3. Loftware Messaging Protocol

All information passed between the client and the server is done in the form of a messaging protocol. The Loftware Messaging Protocol is used to identify the start of a Loftware Message or Packet, the type of request/response (Command), the Message's Sequence, and the length of the variable length data portion that can be used to include request parameters as well as response data. There are actually two parts to a message, the header portion, which includes information about the message, and the optional data portion that contains information mentioned above. Expressed as a pair of c-style struct, a Loftware Message or Packet looks like this:

```
typedef struct WDPProtocolHeader
{
    DWORD      magic;
    DWORD      cmd;
    DWORD      seq;
    DWORD      len;
};

typedef struct WDPProtocol
{
    struct WDPProtocolHeader* header;
    unsigned char*          pData;
};
```

*(A DWORD is a win32 defined type which is a C/C++ 4 byte unsigned long)*

The data portion of the Message contains Command-Specific data in a Command-Specific alignment. Because a Loftware Message has a mandatory header which has a length of 16 bytes (4 DWORDs), when 16 bytes have arrived at the client side (assume the client is in a wait-state for a response) the first byte is checked for an out-of-frame condition (invalid packet received), this magic number is (0x1AFBECFD) which will not change. If the magic check passes, the Command can be checked for a response (Loftware Responses have the 0x8000 bit set). The sequence is a client-issued value to represent the order of the request. Valid client-side sequence values are in the range (0x00000000 – 0x10000000). At this time, client's issuing a Request and expecting a response must block until the response is received (a response with a matching sequence number). It is the client's responsibility to increment this value, the server will only echo the sequence back on a Request/Response. It is also the client's responsibility to ensure the sequence is reset to 0x00000000 when 0xFFFFFFFF is reached). The length represents the length of the data portion and can be 0 (zero) representing no data. The length member **MUST** be accurate as this represents the number of bytes to read in from the socket/buffer for the request/response in question.

#### 4. Data Section

Data is stored in the data portion of the Loftware Message as an array of unsigned characters (Bytes). Any type can be stored in the data area so long as both sides (Client AND Server) know the alignment. Strings are stored as NUL (0x00) terminated values. Any amount of data can [and] is stored in the data portion. Objects, structures, data types, etc can all be marshaled into the buffer and transported.

#### 5. Byte Order

LPS runs on a Little Endian platform, care must be taken when connecting to LPS from a Big Endian platform. Lets look at the magic value 0x1AFBECFD as an example. The following table represents a block of memory, say address 1000. The column headers represent the actual location(s) in memory.

On a Little Endian machine, the Magic would be stored as:

1000	1001	1002	1003
0xFD	0xEC	0xFB	0x1A

On a Big Endian machine, the Magic would be stored as:

1000	1001	1002	1003
0x1A	0xFB	0xEC	0xFD

If a header were created on a Big Endian machine without any byte order conversion, LPS would fail the packet during the magic check "Invalid Header received".

#### 6. Error Messages

For all requests to the server an Error Response (0x8001) can be sent by the server to indicate an error with either the server or the processing of the request. Clients MUST be able to properly handle errors returned by the server. The data portion of the Message will contain a plaintext message describing the error stored as a NUL-terminated string.

```
#define RSP_SNDERRORMSG 0x8001
```

## 7. Login to Server

The login operation to the server is a Request with a specific command and data. The data required for the login is:

Field Num	Field Desc	Type
1	ComputerName	Nul Terminated String
2	Version	"5.5.2.15" Nul Terminate String
3	UserName	Nul Terminate String

- ComputerName - Client's ComputerName or unique machine identifier
- Version - A string representation of a the version of software that the client, for administrative reporting only, however, must exist.
- UserName - Client's UserName or Application name, where the ComputerName identifies the machine, the UserName identifies the user or application on the particular machine (It is possible to have multiple sessions from the same client machine).

The command to set in the header is REQ\_LPSLOGIN (0x0205). The data buffer will contain three NUL-terminated strings, the Length member of the header will represent the total length of the three strings plus three bytes for the NUL bytes (0x00). The following are the defines for the Request and Response for Login:

```
#define REQ_LPSLOGIN 0x0205
#define RSP_LPSLOGIN 0x8205
#define UNP_MAXCLIENTS 0xFF00
```

In response to the Login, the server will either issue an Error Response which will have a command of 0x8001, UNP\_MAXCLIENTS (0xFF00), or the Login Response (0x8205) which will contain the Server Name (String).

If the UNP\_MAXCLIENTS message is returned the server is rejecting the login request due to the client licenses limit. Restarting the Loftware Print Server will reset the license count, it may be necessary to purchase more licenses based on the per-seat usage.

If either an error or the UNP\_MAXCLIENTS response is returned, the client is NOT logged into the server and any attempts at further communications during the current session will result in error returns.

## 8. Unpended Server Shutdown message

During normal server shutdown, the server will issue an unpended Shutdown response to all Logged in, connected clients before attempting to gracefully shutdown the connection. It is the client's responsibility to handle this message. Upon receipt of this message, the client cannot send any more requests and can assume that there will be no more responses on this connection. It is up to the client to initiate a login retry loop to reconnect with the server. When the server is restarted, a UDP message (broadcast) is sent out on port 2723 to indicate it's online status.

```
#define RSP_SVRSHUTDOWN          0x0002
```

## 9. Get Tab File Information Request (Field List)

To obtain tab file information for a particular label, send a REQ\_GETTAB (0x0013) with the label path/filename in the data section.

Field Num	Field Desc	Type
1	Path/LabelName	Nul Terminated String

?? If the label is located in the default Loftware Labels path, only the filename need be sent ("UPS4x6.lwl")

?? If the path is relative to the Labels Path (Located in a sub-directory of the Labels Path) then a relative path can be sent (ie "\Shipping\UPS4x6.lwl")

?? If the path is included, the path MUST be correct and reachable by the server ([\\FS01\Labels\Shipping\UPS4x6.lwl](#)).

The tab file information must exist in the label file (Label must have been saved running a version of Loftware 4.2 or greater). If the label is not found, or the tab file marker is not located an error (RSP\_SNDERRORMSG) is returned with the error string in the data section. On a successful lookup, the tab file information is returned in the response. Each field row is terminated by a Carriage-Return/Line feed pair (0x0D 0x0A).

See chapter 8 "On-Demand Printing" in the Loftware User's Guide for more information on Tab Files.

```
#define REQ_GETTAB          0x0013  
#define RSP_SNDTAB         0x8013
```

## 10. Sending a print job

```
#define REQ_SNDJOB          0x0150
```

To send a print job to the server there are six pieces of information that must be provided:

Field Num	Field Desc	Type
1	Request Type*	Byte (8bits)
2	Printer Number	WORD (16 Bits) (0 if not used)
3	Printer Alias	Nul-Terminated String
4	Job Name	Nul Terminated String
5	Job Data	Nul-Terminated String

\* Request Types:

- 0 – Pass File
- 1 – CSV File
- 2 – XML File

If the Loftware Printer Number is known, the Printer Number field is set otherwise send a PrinterNumber of zero (0x00). The Printer Number field is a WORD, or 16 bit value. If the PrinterNumber field is nonzero, the Printer Alias field is ignored, if the PrinterNumber value is zero (0x00) the PrinterAlias field is used.

The Printer Alias field is a NUL-Terminated (0x00) string value, represented the printer's alias assigned during Loftware Printer Configuration.

The JobName field is a unique, client-assigned, information string value. The JobName will follow the job throughout its existence in the Loftware Printing System (LPS).

The job data is the job itself in the same format it would be stored on disk as. Lines must be Carriage Return/Line-Feed terminated. Due to the requirement of the Printer Information outside the scope of the job itself, multi-printer or 'stacked' job requests are NOT allowed.

The server will return a response upon completion of the job or a printer error that are covered in the following sections.

**Due to the synchronous nature of this integration method, no other jobs may be submitted until a response is received for this particular request.**

## 11. Job Status Response

Status Update Responses are sent upon completion, failure, or delay (due to printer error) of a job request.

```
#define RSP_JOBUPDATES    0xF230
```

When the client logs into the server, the provided ComputerName is used as a filter for print jobs. Any status update with the matching ComputerName will be routed to ALL connected clients with the same ComputerName. There are five pieces of information provided in a Status Update Response (in the data portion):

Field Num	Field Desc	Type
1	Update Type	Byte
2	Printer Number	WORD (16 Bits)
3	Job Number	DWORD (32 Bits)
4	Current Request	DWORD (32 Bits)
5	Text String	Nul Terminated String

The Update Type is a bit wise character field that can represent multiple results in one message (i.e. job completed with errors). The following is a list of the bit representations:

```
#define STAT_CRIT_FAIL        0x02  
#define STAT_PRINTED         0x04  
#define STAT_PRINTERERROR    0x08  
#define STAT_DELETED         0x10  
Reserved                    0x20
```

A Printed update represents a completed job, if the Critical Failure bit is also set, the job completed with errors (at least one request printed, and at least one request failed). An update with the only the Critical Failure bit set represents a totally failed job (nothing printed). The Deleted bit represents a user deleting the request through the status program. Bits 0x08 & 0x20 are reserved and not sent to the client.

The Printer Number field represents the Loftware Printer number the job is printing on, whether the printer number was supplied in the original request. The printer alias is not reflected in the update due to size constraints. The Printer Number is a WORD (16 bit value).

The Job Number references the job number returned during the sending of the print job. The Job Number is a DWORD (32 bit value).

The text string field can be an error message if the job failed, or an update status if the job is printing.

## 12. Client Logout

When the client has completed its session with LPS, the first part of a graceful shutdown entails sending a REQ\_LOGOUT (0x000B) message. Upon receipt of the client's logout, the server will NOT send any more responses. It is up to the Client to perform a graceful shutdown after receipt of the RSP\_LOGOUT response.

There are no parameters in either message. Once a logout has occurred, the session MUST be closed, a client issuing a Login Request AFTER a logout has undefined behavior.

```
#define REQ_LOGOUT          0x000B
#define RSP_LOGOUT         0x800B
```

## 13. Enhanced Mode Login And Send Print Request

For cases where holding the socket open for multiple requests is not feasible/possible, there is a request that handles the login and the sending of a print request all in one transaction

```
#define REQ_LOGINSENDWAIT  0x0206
```

Field Num	Field Desc	Type
1	Computername	Nul-Terminated String
2	Version	6.0.0.0
3	Username	Nul-Terminated String
4	Request Type*	Byte (8bits)
5	Printer Number	WORD (16 Bits) (0 if not used)
6	Printer Alias	Nul-Terminated String
7	Job Name	Nul Terminated String
8	Job Data	Nul-Terminated String

\* Request Types:

- 0 – Pass File
- 1 – CSV File
- 2 – XML File

## 14. Retrieval of Configured Printer Information

To retrieve a current list of the configured printers in csv format, use the REQ\_GETPRINTERINIEX command (no parameters).

```
#define REQ_GETPRINTERINIEX 0x0208
```

Printer information is returned in csv format without headers. The header is provided here for explanation.

```
Printer Number,Printer Name,Port,Loftware Printer ID,Printer Alias(Name)  
1, "Loftware Intermec Easy Coder 3400", "112.34.111.222:9100", 3000, "SHIPPING1"
```

In the above example the Loftware Printer Number is 1 (This is the value used in the "Printer Number" field in Job Request), the Printer Name is "Loftware Intermec Easy Coder 3400", the Port is "112.34.111.222:9100" (In this case the printer is IP addressable), the Alias is "SHIPPING1" (This is the value used in the "Printer Alias" field in the Job Request).