

# DMPP 2020 Technical Guide



## Texas Department of Public Safety TLETS Interface Agency Implementation DMPP 2020 Technical Guide



*This document contains information, specifications and diagrams of a highly proprietary and confidential nature. This information is intended only for use by the organization, to which it was distributed directly by Datamaxx Applied Technologies, Inc. Under no circumstances is there to be any duplication, distribution or other release of the information contained in this document to any other organization or person, by any means, without written authorization from Datamaxx Applied Technologies, Inc.*

[www.Datamaxx.com](http://www.Datamaxx.com)

**COPYRIGHT INFORMATION:**

You may copy and/or transmit this Datamaxx document only in its original form and in its entirety, including this notice.

**You may not: (a) modify this document; (b) sell, charge for, or obtain any financial benefit from the copying or transmittal of this document or any copy; (c) remove any Datamaxx branding, or (d) remove any copyright, trademark, or other proprietary notices from this document.**

**This document is provided to you “AS IS” and Datamaxx Applied Technologies, Inc. provides no warranty as to the results you may obtain from using it.**

**Datamaxx™, the Datamaxx logo, Datamaxx Message Processing Protocol®, DMPP-2020®, Datamaxx Standard Embedded Object®, DSEO-2020® and Datamaxx Applied Technologies, Inc. Leading Law Enforcement Technology® are trademarks of Datamaxx Applied Technologies, Inc. Any other product names used within this document are the trademarks of their respective holders.**

**Copyright © 2004 Datamaxx Applied Technologies, Inc. All rights reserved.**

Datamaxx Applied Technologies, Inc.

*This document contains information, specifications and diagrams of a highly proprietary and confidential nature. This information is intended only for use by the organization to whom it was distributed directly by Datamaxx Applied Technologies, Inc. Under no circumstances is there to be any duplication, distribution or other release of the information contained in this document to any other organization or person, by any means, without written authorization from Datamaxx Applied Technologies, Inc.*

**Published by:**

Datamaxx Group, Inc. d/b/a  
Datamaxx Applied Technologies, Inc.  
2001 Drayton Drive  
Tallahassee, Florida, 32311  
(850) 558-8000  
[www.datamaxx.com](http://www.datamaxx.com)

**Revision History:**

Version	Date	Notes
Version 1.0	09/2004	

Datamaxx Applied Technologies, Inc.

*This document contains information, specifications and diagrams of a highly proprietary and confidential nature. This information is intended only for use by the organization to whom it was distributed directly by Datamaxx Applied Technologies, Inc. Under no circumstances is there to be any duplication, distribution or other release of the information contained in this document to any other organization or person, by any means, without written authorization from Datamaxx Applied Technologies, Inc.*

## Table of Contents

<b>1</b>	<b>INTRODUCTION .....</b>	<b>1</b>
<b>2</b>	<b>IMPLEMENTATION OVERVIEW .....</b>	<b>3</b>
<b>3</b>	<b>DMPP-2020 CONNECTION MANAGEMENT IMPLEMENTATION .....</b>	<b>4</b>
<b>4</b>	<b>DMPP-2020 PROTOCOL IMPLEMENTATION .....</b>	<b>5</b>
<b>5</b>	<b>DMPP-2020 FRAME FIELDS .....</b>	<b>6</b>
5.1	DMPP-2020 Extended Message Header Fields .....	6
5.2	Keep Alive .....	7
5.3	Message Segmentation.....	7
<b>6</b>	<b>ENCRYPTION IMPLEMENTATION.....</b>	<b>8</b>
6.1	Binary Objects .....	8
6.2	Message Examples.....	8
6.3	Message Routing Header Format.....	13
6.4	Transaction Data Message Format.....	14

## 1 INTRODUCTION

This document is meant to offer a brief discussion of the benefits of utilizing this protocol within law enforcement systems employing a server interface, as distinct from a standalone device. For purposes of this document, a server interface is a remote server (or similar device) within the network or subnetwork that manages its own security schema for user authentication and information access. References to TLETS will be in relation to the implementation of the new OpenFox™ TLETS message switch architecture unless otherwise specified.

This protocol has been chosen as the mechanism to provide compliance with national standards while also providing an environment that is not burdensome for regional/remote systems to implement.

Control Terminal Agencies (CTAs) within each State must deploy systems that comply with the requirements of national agencies such as NCIC and NLETS. Some of these requirements are listed below:

- Implementing a protocol that provides guaranteed message delivery
- Identification of users responsible for transactions
- Ability to process transactions that meet national standards
- Ability to process image data

All of these requirements are implemented in the system with this protocol. The DMPP-2020<sup>®1</sup> protocol is an application-to-application protocol over TCP/IP that guarantees delivery of messages.

The CTA must be able to determine the individual operator responsible for any message that transverses through the system. The TLETS system as implemented in conjunction with the DMPP-2020 protocol and the inclusion of the Message Routing Header achieves this requirement.

The system supports all standard NLETS, NCIC-2000 and existing legacy TLETS transactions. In addition, the system will also be able to process images.

The combination of DMPP-2020 and the OFML<sup>®2</sup> (XML based message formats) defines a strategy that meets all requirements, and can be implemented on any platform that supports TCP/IP. This approach also allows use of existing agency routing methodologies.

In addition to the national standards, the CTA has an obligation to provide an environment to regional systems that is consistent and can be implemented with a reasonable amount of effort

---

<sup>1</sup> DMPP-2020 is a registered trademark of Datamaxx Applied Technologies, Inc.

<sup>2</sup> OFML is a registered trademark of Datamaxx Applied Technologies, Inc.

while using state-of-the-art protocols and network facilities. The DMPP-2020 protocol as implemented in the TLETS system provides such an environment.

The combination of DMPP-2020 and OFML defines a strategy that meets all requirements, and can be implemented on any platform that supports TCP/IP.

## 2 IMPLEMENTATION OVERVIEW

The implementation has finite aspects that fit into a structured programming model, as described below. Each aspect is intentionally disassociated from the others to permit a simple structured implementation. The DMPP-2020 protocol specification is provided in a separate document.

- DMPP-2020 Protocol Implementation
- OFML Implementation

The DMPP-2020 implementation is further divided into three (3) subsections described below:

- Connection Management
- Data Stream Processing
- Encryption Processing

The encryption processing uses the National Institute of Standards and Technology (NIST) “AES” algorithm. This algorithm meets the requirements of national security policies for Law Enforcement and Criminal Justice systems, and is public domain with source code available for a variety of languages via a free download.

Information is available at: <http://csrc.nist.gov/>.

DMPP-2020 is designed to be processed using “State Machine” programming techniques. It has fully deterministic processing characteristics, which also allows for a very robust implementation and easy error detection.

The OFML data structures permit the use of existing control information, and add new fields for user and device identification. The interface server concept permits flexibility of operation at the local agency, yet provides TLETS with the necessary monitoring and control mechanism.

All aspects are described in detail in this document.

### 3 DMPP-2020 CONNECTION MANAGEMENT IMPLEMENTATION

The Interface Agency will open a connection to the TLETS system. TLETS will be waiting, listening for the connection. The connection will be made to an IP address and port number assigned by the TLETS staff.

The IP address and port number should be configurable, **not** hard coded. This will expedite any changes that may occur and reduce potential cost to the agency.

When a connection request is received from an Interface Agency on the assigned port number, the IP address of the requester (i.e., the Interface Agency address) will be validated. If the IP address fails validation, the TLETS system will close the connection.

Once the connection has been established, data flow may commence. Either TLETS or the Interface Agency may close the connection at any time. The Interface Agency must be able to recognize and handle the close function, and automatically try to reconnect. All reconnect attempts should be defined, configurable cycle to minimize network contention. Suggested default connection cycle time is 30 seconds, TLETS will recognize when the connection has been closed by the Interface Agency, and hold all messages on the queue for that agency.

An idle time (also known as “Keep Alive”) has been defined. This is used to detect a “half session failure” in which one or other of the end-points has shutdown without an appropriate notification. This can also occur if there is a communications failure. Therefore, the Interface Agency must send a “Keep Alive” on a timed interval after the last message sent or received. If this message is not received, TLETS will consider that the Interface Agency has lost communications, close its side of the connection, and await a new connection request.

If the Interface Agency sends a “Keep Alive” and does *not* receive a response within a configurable time-period (suggested default 30 seconds), then it should assume that communications with TLETS has been lost, close its connection, and attempt a new connection.

Please note that the “Keep Alive” is a true “idle” timer in that its period commences from the start of the last message processed, sent or received. It is **not** a periodic message, synchronized to clock time.

It is important that all configuration values are easily set, and not “hard coded”.

The message contents are described in the following section.

## 4 DMPP-2020 PROTOCOL IMPLEMENTATION

This section of the document provides practical examples for the implementation of the DMPP-2020 protocol in the TLETS environment. It will document the specific technique that TLETS uses to provide reliable, binary object capable communications.

TLETS supports the widely accepted standards of communication put forth in the NCIC and NLETS TCP/IP specifications, and therefore implements DMPP-2020 in a manner complying with these national standards.

**TLETS will implement DMPP-2020 with the encryption as defined in the DMPP-2020 specification.**

The system uses the application level acknowledgements to deliver messages reliably, as well as the status checking function to implement an idle line timer (“Keep Alive”) as described above.

The TLETS system will require the segmentation of large messages and an indication of which message segments, if any, contain image data. TLETS will also require client data messages to present images in the DSEO-2020 format.

TLETS will send messages to the remote systems using segmentation. The remote systems must support the ability to receive multiple segments in one message. An image sent from the server will not span multiple segments.

## 5 DMPP-2020 FRAME FIELDS

The DMPP-2020 protocol defines a message by virtue of its Start Pattern, length field, and Stop Pattern. These values create a fully deterministic message frame, which can then be processed.

It cannot be over emphasized that TCP/IP data is presented as a stream, and therefore a “read” may contain a message fragment, a whole message, multiple messages, or one or more messages and a fragment. The programmer must be aware that a DMPP-2020 defined pattern (notably the “Stop Pattern”) may be fragmented across received data blocks. Thus, once a valid start pattern has been received, the end pattern must **not** be inspected until the complete length of the frame has been received.

The use of a “State Machine” in the programming facilitates the processing of DMPP-2020 frames.

Once the frame has been detected and processed, the DMPP-2020 header can be processed, as described in the next section.

### 5.1 DMPP-2020 Extended Message Header Fields

Six (6) fields in the DMPP-2020 extended message header will be used by TLETS. The fields and their appropriate values appear below.

Header Length	This field is always set to 16 (hex 0010) or 42 (hex 002A), if the message is encrypted.
Function	The functions supported are listed below (hex): 0001 - Data Message, no ACK, Final Block 0002 - Data Message, ACK requested, Final Block 0003 - Data Message, no ACK, More to Follow 0004 - Data Message, ACK requested, More to Follow 0011 - Positive ACK to data message 0012 - Negative ACK to data message 0021 - Request system status 0022 - Status response
Validation	The contents of this field are returned by TLETS, or must be returned to TLETS when a message is received.
Data Length	This field represents the data length as an unsigned 32-bit number. Please note that no single block may be larger than 32K.
Status	The TLETS uses this field as documented in the DMPP-2020 specification.
Destination	The value is always set to hex 0001 on outgoing messages, and ignored on inbound messages.

The following are also present when encrypted data is transmitted:

Header Length	This field will always be 26 (hex 0001a).
Encryption Type	This field will always be 1 (hex 00001).
Book Id	This field will contain the 2-byte book Id
Key Id	This field will contain the 2-byte key Id
CRC value	This field will contain the 16-bit CRC-16 value
IV Value	This field will contain the 16-byte AES Initialization vector

## 5.2 Keep Alive

TLETS will use the status request/response function to act as an idle line timer (“Keep Alive”). TLETS will terminate a connection that has had no activity for 60 seconds. To prevent an idle connection from terminating, clients are expected to issue a system status request message before 60 seconds of idle time. It is recommended that a request is sent every 30 seconds if no other traffic has been sent during that time. TLETS will respond with a “system available” status response and reset the idle timer for the connection.

## 5.3 Message Segmentation

To maximize resource efficiency at the central site and to manage a large number of client connections, the TLETS system requires that messages larger than 32K bytes be segmented. The term segment and block are used interchangeably. If a DSEO-2020 object is present in a message, it must be completely contained within a single message segment. Please note that a single message segment may contain multiple DSEO-2020 objects (so long as their combined size is under the 32K byte limit). A message may be broken into any number of segments, and each segment need not attain the 32K byte maximum. If the function code for a data message requests an ACK, and is not the final block, the next block should not be sent until the ACK for the prior block is received. Likewise, after sending a final block requesting an ACK, the next message should not be started until the ACK is received. If no ACK is received for a data block within 60 seconds, the connection should be closed and a new connection attempted. Any partially completed message (some blocks sent and ACKed, but not all) should be resent in its entirety upon successful establishment of the new connection.

## 6 ENCRYPTION IMPLEMENTATION

The implementation supports encryption, with dynamic keys. The keys are organized into “books”. The sender selects a key from the assigned book at random and indicates the values in the header. The data is then encrypted using the NIST “AES” 256 bit key algorithm.

The NIST AES algorithm is public domain. Source code may be freely downloaded, as desired. Information is available at: <http://csrc.nist.gov/>.

### 6.1 Binary Objects

As documented above, TLETS will require that all inbound and outbound objects be wrapped in DSEO-2020 format. An object when present must be completely contained with a single segment. The status field in the DMPP-2020 header should reflect the content of the block. The two (2) status codes used are listed below.

- 01 Message segment contains no object data
- 05 Message segment contains at least one DSEO-2020 formatted image

Please note that in the DMPP-2020 specification status code 01 states “Message may contain binary object in Unisys format”. Since TLETS does not support Unisys formatted objects this code is used to indicate no object is present. **TLETS will only scan segments for DSEO-2020 objects if they have the status code set to 05.** TLETS will ensure that all segments bound for the peer have the status code set correctly, so the peer need not scan for DSEO-2020 objects if the segment status code is set to 01.

### 6.2 Message Examples

The following message examples are taken from a live system TCP/IP trace and are presented to demonstrate the appearance of messages using DMPP-2020. Since they are taken from a system that is **not TLETS**, some of the features (e.g. XML) are not indicated.

The message examples include:

- Data message with acknowledgement from the Interface Agency.
- Data message with acknowledgement to the Interface Agency.
- Status request exchange (Keep Alive) from the Interface Agency.

These examples represent the normal operation of an interface. The two systems exchange data messages, and during idle periods, a status request/response (“Keep Alive”) is conducted.

**Data message with acknowledgement from the Interface Agency to TLETS.**

The following is an example of a message sent from the Interface Agency to TLETS.

Offset	Hex Data	ASCII Equivalent
00000000	ff00aa55 00000053 00100002 32313230	...U...S....2120
00000010	00000037 00210001 464f5859 2e52512e	...7!...FOXY.RQ.
00000020	2a484152 52592e4f 4b4f4850 30303339	*HARRY.INWYP0039
00000030	2e53442e 4c49432f 33413236 3739362e	.SD.LIC/3A26796.
00000040	4c49592f 31393939 2e4c4954 2f504355	LIY/1999.LIT/PCU
00000050	aa00ff	...

The message breakdown is:

FF00AA55	Start Pattern
00000053	Message Length
0010	Extended Header length
0002	Function - Data message, ACK requested, final block.
32313230	Validation Code
00000037	Data length - length of the actual message data (from FOXY to LIT/PC).
0021	Status Code - ignored
0001	Destination - ignored
464F thru 5043	Message Data (text)
55AA00FF	Stop Pattern

TLETS will respond with :

Offset	Hex Data	ASCII Equivalent
00000000	ff00aa55 0000001c 00100011 32313230	...U.....2120
00000010	00000000 00010001 55aa00ff	.....U...

The message breakdown is:

FF00AA55	Start Pattern
0000001C	Message Length
0010	Extended Header length
0011	Function - Positive ACK
32313230	Validation Code - echoed from the input message
00000000	Data length - zero
0001	Status Code (meaningless)
0001	Destination (always 1)
55AA00FF	Stop Pattern

**Data message with acknowledgement from TLETS to the Interface Agency.**

TLETS would send the following message to the Interface Agency.

Offset	Hex Data	ASCII Equivalent
00000000	ff00aa55 000000b4 00100002 00000001	...U.....
00000010	00000098 00010001 464f5859 2e2a4841	.....FOXY.*HA
00000020	5252592e 4e434943 20202020 20202031	RRY.NCIC 1
00000030	33363731 2031373a 33353a32 32204d52	3671 17:35:22 MR
00000040	49203039 30313831 0d0a464f 58592020	I 090181..FOXY
00000050	20202020 20303030 30392031 373a3335	00009 17:35
00000060	3a323220 30362f31 362f3230 30300d0a	:22 06/16/2000..
00000070	0d0a314c 3031464f 58592c4d 52494430	..L01FOXY,MRID0
00000080	39303138 300d0a30 4b304850 30303339	90180..0K0HP0039
00000090	0d0a4e4f 20524543 4f524420 4c49432f	..NO RECORD LIC/
000000A0	33413236 37393620 4c49532f 53440d0a	3A26796 LIS/SD..
000000B0	55aa00ff	U...

The message breakdown is:

FF00AA55	Start Pattern
000000B4	Message Length
0010	Extended Header length
0002	Function - Data message, ACK requested, final block
00000001	Validation Code - should be returned in the client's ACK
00000098	Data Length - from FOXY to 'CR'LF'
0001	Status Code (meaningless)
0001	Destination (always 1)
464F thru 0D0A	Message Data
55AA00FF	Stop Pattern

The client responded with the following acknowledgment:

Offset	Hex Data	ASCII Equivalent
00000000	ff00aa55 0000001c 00100011 00000001	...U.....
00000010	00000000 00010001 55aa00ff	.....U...

The message breakdown is:

FF00AA55	Start Pattern
0000001C	Message Length
0010	Extended Header length
0011	Function - Positive ACK
00000001	Validation Code - returned from the TLETS output
00000000	Data Length - zero
0001	Status Code - ignored
0001	Destination - ignored
55AA00FF	Stop Pattern

There is a consideration when data messages may be sent with a function code of "0001 - data message, no ACK, final block". If TLETS sets this value in the function field, it is not expecting an ACK to the message, and one should not be sent to TLETS. Likewise, if a client device sets this value in the function field, TLETS will honor it and not send an ACK. TxDPS recommends the use of message ACKs for all standard user transactions.

**Status request exchange (Keep Alive) from the Interface Agency to TLETS.**

Offset	Hex Data	ASCII Equivalent
-----	-----	-----
00000000	ff00aa55 0000001c 00100021 31363138	...U.....!1618
00000010	00000000 00210001 55aa00ff	.....!...U...

The message breakdown is:

FF00AA55	Start Pattern
0000001C	Message length (total length of this data message)
0010	Extended Header length (always 16 - hex 10)
0021	Function - Request system status
31363138	Validation Code - this will be returned (see response below)
00000000	Data Length - this is zero for status messages
0021	Status Code - ignored
0001	Destination - ignored
55AA00FF	Stop Pattern

This message caused TLETS to reset the idle timer for this connection, and respond with the following message:

Offset	Hex Data	ASCII Equivalent
-----	-----	-----
00000000	ff00aa55 0000001c 00100022 31363138	...U....."1618
00000010	00000000 00210001 55aa00ff	.....!...U...

The message breakdown is:

FF00AA55	Start Pattern
0000001C	Message length (total length of this data message)
0010	Extended Header length
0022	Function - Status Response
31363138	Validation Code - echoed from the status request
00000000	Data Length - this is zero for status messages
0021	Status Code - Available and ready
0001	Destination - always set to 1
55AA00FF	Stop Pattern

### 6.3 Message Routing Header Format

The Message Routing Header concept permits an interface agency to manage their own users and devices, yet provides enough information to TLETS so that national policies can be met, with regards to device and user identification.

In this model, the interface agency has assumed the responsibility of user and device management in a manner acceptable to TLETS. A “Device Address Code (DAC)” identifies a device that has been defined this way.

The TLETS specification uses this exact same control field, but adds other data to it to construct the information needed to process the message. The other data fields include the terminal device name and user identification data.

All message data, including the header, is encapsulated in XML following the OFML specification. A complementary document *Omnixx<sup>TM</sup> Force/OpenFox<sup>TM</sup> Markup Language OFML Interface in the Law Enforcement Environment* is provided that defines the OFML data structures.

The fields from the OFML specification required for the Message Routing Header are as follows (in **bold**):

```
<OFML>
  <HDR>
    <ID>(Contents)</ID>
    <DAC>DeviceId</DAC>
    <DAT>(Contents)</DAT>
    <REF>Reference</REF>
    <MKE>(Contents)</MKE>
    <USR?>UserId</USR>
    <ORI?>(Contents)</ORI>
    <DST*>(Contents)</DST>
    <CTL?>(Contents)</CTL>
    <SUM?>(Contents)</SUM>
  </HDR>
  <TRN|RSP>
    <MFC*>(Contents)</MFC>
  </TRN|RSP>
</OFML>
```

---

<sup>3</sup> Omnixx is a trademark of Datamaxx Applied Technologies, Inc.

<sup>4</sup> OpenFox is a trademark of Computer Projects of Illinois, Inc.

## 6.4 Transaction Data Message Format

In the new TLETS implementation, all message formats will use XML following the OFML specification. The XML formatted data streams may be processed with industry standard programming tools such as style sheets and parsers.

The field contents will change very little or not at all. The packaging and identification is what will change.

In the current implementation, fields must be submitted in a specified order, separated by a delimiter (hex “.” character) or with identifiers attached (e.g. “LIC/”). In the new implementation with XML, each field is identified as an element within a well-formed XML document. Order is not important, except for those considerations noted in Section 3.1 of the OFML specification “*Additional requirements for MFC elements*”.

The following example shows a query using OFML. It is a Motor Vehicle query from a client assuming the switch spawns DQ to NLETS and QV to NCIC:

```
<OFML>
  <HDR>
    <ID>0204002453</ID>
    <DAC>DEVA</DAC>
    <DAT>20010824073119</DAT>
    <REF>12345ABCDE</REF>
    <MKE>DQ</MKE>
    <ORI>ID1234567</ORI>
    <DST EID="DRI">VA</DST>
    <CTL>ABC1234567</CTL>
    <SUM>DQ: SMITH, GEORGE</SUM>
  </HDR>
  <TRN>
    <NAM>SMITH, GEORGE</NAM>
    <DOB>19511205</DOB>
    <SEX>M</SEX>
  </TRN>
</OFML>
```

An agency also needs to consider new functions such as image entry and retrieval. At initial rollout, all devices associated with an interface agency will be marked as “Non image capable” to avoid the transmission of image data that might not be acceptable. Once the agency has implemented image capability (via the DSEO specification) TLETS will define the device as “Image capable”.

None of the image support changes are required for initial implementation, but should be considered in the planning process.