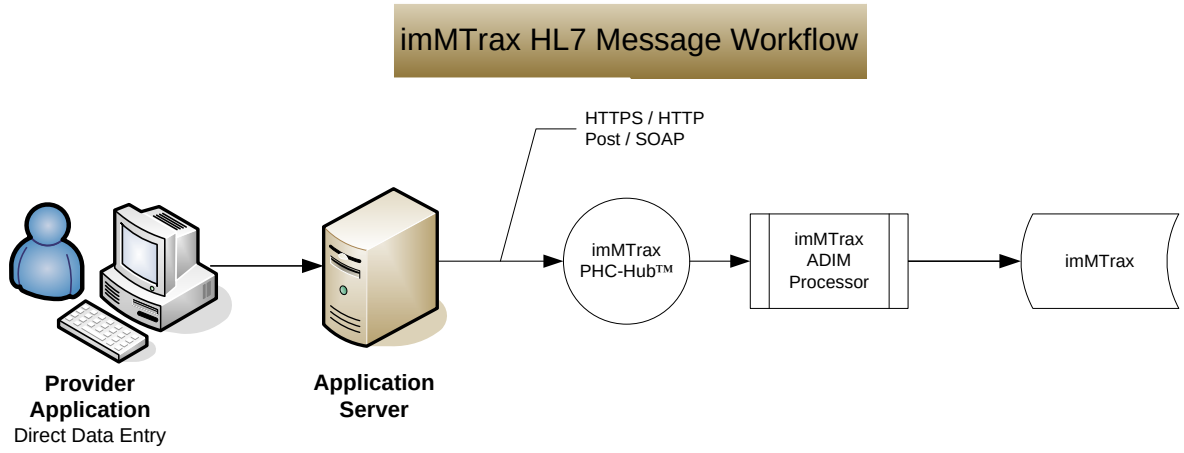
HL7 messages are received by the imMTrax inbound interface and processed in a two-stage manner. The first stage accepts the message content and performs field level validation in real-time as the HL7 messages are received. This is performed by PHC-Hub™. For coded fields, this includes database lookups for valid values. An ACK message is issued after this processing stage is complete.

The second stage, the ADIM processor, applies imMTrax business rules and updates the imMTrax patient and immunization data. The diagram below indicates how HL7 messages flow when sent inbound to imMTrax.

imMTrax HL7 Message Workflow



TRANSPORT PROTOCOL

imMTrax supports the user ID and password for security (HTTP POST) or SOAP protocol to connect to PHC-Hub. Each transport option will be discussed in the sections below.

HTTPS & HTTP POST

An HL7 Immunization Registry task force (Rockmore, Yeatts, and Davidson), produced a specification titled "**Transport of Immunization HL7 transactions over the Internet Using Secure HTTP**". The imMTrax HTTP transport protocol follows this specification. The process is described as follows:

The EHR application (or other sending system) initiates the HTTP POST transaction to imMTrax with the following data fields that populate the message envelope:

- USERID – This is the imMTrax-assigned User ID.
- PASSWORD – imMTrax will assign a temporary password for the user. The password will be changed by the user to ensure it is kept confidential. The password must contain at least one number, alphabetic characters, and be at least eight characters long.
- MESSAGEDATA – The HL7 message is ASCII text. The message must begin with the character string "MSH".

The sending application will construct the HTTP transaction using the supplied values and including the actual HL7 message as the MESSAGEDATA parameter. The USERID and PASSWORD values will be used to insure the POST transaction is authentic. The imMTrax organization associated with the USERID will be used to associate the inbound message with a processing profile. Each unique site for which a message is sent must be identified within the message.

Once the message has been sent, the EHR should expect a standard acknowledgement (ACK) message in response. The ACK message is part of the single HTTP transaction and is returned over the same connection without the message envelope used for the original message. Error messages with details on why the inbound HL7 message was not successfully processed are also returned.

Once the ACK message has been received, the next HL7 messages should represent all patients and patient vaccination activity that has occurred since the last communication with imMTrax. The process will repeat until all HL7 message activity has been sent. imMTrax can support and prefers real time HL7 messages, which in reality ranges from a few seconds to a few minutes after the event has been documented in the EHR. **NOTE:** The process or mechanism that triggers the vaccination message to be sent to imMTrax should occur within 24 hours. If messages are sent in a batch, the sending system must be set to send data to imMTrax at a time /day/cycle that imMTrax approves.

Additional detail on HTTP Post is contained in Appendix 1.

SOAP (SIMPLE OBJECT ACCESS PROTOCOL)

PHC-Hub, imMTrax's integration tool, transports HL7 messages utilizing the following elements:

- Username and password elements for authentication
- HL7Message element to transmit the actual message

SOAP and POST methods are both transported over HTTPS. All of the elements listed above are used in both SOAP and POST methods. The HL7 data are communicated differently using SOAP vs HTTP Post. The SOAP/WSDL uses the *USERNAME*, *PASSWORD*, and *HL7Message* to pass the messages whereas the HTTP POST method uses the *USERNAME*, *PASSWORD*, and *MESSAGEDATA*.

A sending application can use the SOAP transport protocol[1] to send messages to imMTrax. SOAP is an XML based protocol which consists of three parts:

- Envelope – A required element that identifies the XML document as a SOAP message.
- Header - An optional element that contains application-specific information (like authentication, payment, etc.) about the SOAP message.
- Body - A required element that contains call and response information. The Body contains the actual SOAP message intended for the ultimate endpoint of the message.

SOAP may be used over HTTPS with either simple or mutual authentication. It is the advocated method to provide web service security.

[1] http://en.wikipedia.org/wiki/SOAP

A SOAP message with the parameters needed for a search is sent to imMTrax PHC-Hub (with web services enabled). PHC-Hub then returns an XML-formatted document with the resulting data, e.g., patient demographic and vaccination information. The data being returned (in a standardized machine-parsable format) is then integrated directly into a third-party web site or application.

Processing Model

The SOAP processing model describes a distributed processing model, its participants, the SOAP nodes, and how a SOAP receiver processes a SOAP message. The following SOAP nodes are defined:

SOAP sender: A SOAP node that transmits a SOAP message.

SOAP receiver: A SOAP node that accepts a SOAP message.

SOAP message path: The set of SOAP nodes through which a single SOAP message passes.

Initial SOAP sender (Originator): The SOAP sender that originates a SOAP message at the starting point of a SOAP message path.

SOAP intermediary: A SOAP intermediary is both a SOAP receiver and a SOAP sender and is targetable from within a SOAP message. It processes the SOAP header blocks targeted at it and acts to forward a SOAP message towards an ultimate SOAP receiver.

Ultimate SOAP receiver: The SOAP receiver, in this case PHC-Hub, is a final destination of a SOAP message. It is responsible for processing the contents of the SOAP body and any SOAP header blocks targeted at it. In some circumstances, a SOAP message might not reach an ultimate SOAP receiver, for example because of a problem at a SOAP intermediary. An ultimate SOAP receiver cannot also be a SOAP intermediary for the same SOAP message.

Examples of SOAP messages can be found in Appendix 2 of this document. Reference documents on SOAP can be found in Appendix 1 of this document.

SECURITY

The interface needs to use the Internet public network for communication, and then the transaction must be encrypted using the HTTPS protocol.

The Montana Immunization Program has a PHC-Hub security certificate. The provider's vendor or information technology support person may need information about the certificate so that they can set their system to recognize PHC-Hub's certificate (install the certificate). Once these settings are complete the EHR will accept the PHC-Hub certificate and then be able to send their messages to the PHC-Hub web servlet. Information on the PHC-Hub certificate can be obtained by contacting MT immunization program staff at **(406) 444-5580.**

APPENDIX 1 – HL7 MESSAGE TRANSPORT TO PHC-HUB

This appendix is a copy of the document "*Transport of Immunization HL7 transactions over the Internet Using Secure HTTP*" Version 1.0, September 17, 2002. Authored by The HL7 Immunization Registry Task Force sub group on HTTP message transport (Joseph Rockmore – IBM Corporation, Andrey Yeatts – Scientific Technologies Corporation, Kevin Davidson – QS Technologies, Inc).

This document discusses conventions that may be used to transport Health Level Seven (HL7) messages over the Internet using Secure HTTP (HTTPS). It is the intent of sub group to use existing standards wherever possible.

PRIVACY

When transporting identifiable health information, the privacy of the information must be insured. Privacy may be insured by encrypting the message or transmitting the message over a secure channel. The HTTPS protocol, widely used for secure transactions in eCommerce, provides encryption and is recommended by this standard. The HTTPS protocol is defined in RFC 2660 (*http://www.ietf.org/rfc/rfc2660.txt*);

AUTHENTICATION

Health information messages state important facts about personal information. Because of this, it is necessary to provide assurance of the identity of party asserting the facts in these messages. Authentication provides such assurance.

Two authentication methods are proposed.

1. User Name/Password. imMTtrax provides each of its clients (other immunization registries and data providers) a User ID and a temporary password. The client will need to login and change the temporary password to a password only the client knows. The client present this User ID and password whenever sending transactions.
2. The HL7 message will be digitally signed using X.509 certificates and formatted according to the S/MIME standard. X.509 is a standard of the International Telecommunications Union.

Method 1 is considered primarily as a means whereby immunization data providers may authenticate with PHC-Hub. Method 2 is the preferred means for authentication between state immunization information systems. However, either method is allowed in either situation subject to Montana law and imMTrax policy.

The sub group also recognizes that the complexity of implementing the digital signature may result in the User ID/Password method being the first deployed.

The S/MIME standard provides a structure to format messages that are digitally signed using an X.509 certificate. Encryption is an optional component of S/MIME. This standard assumes that encryption through HTTPS or other secure channel will be used, and therefore use of the encryption facility of S/MIME is not required.

In order to use S/MIME, both the sender and the receiver must obtain X.509 digital certificates from agreed-upon Certificate Authority(s). The presentation of a message from a recognized Certificate Authority insures the identity of the sender and the integrity and non-deniability of the message. It does not, in and of itself, determine whether the sender is someone that PHC-Hub should talk to.

imMTrax uses an SSL certificate. The certificate can be found at the following websites: https://immtraxtest.org/phc/HL7Server (imMTrax test application) and https://immtrax.org/phc/HL7Server (imMTrax production). Install the certificate on the server that will be sending the messages. If you need assistance with this process, contact the imMTrax office.

Transport Protocol for HL7 Messages over HTTPS using User ID/Password Authentication

When using User ID/Password Authentication, application programs will contact the IIS server by issuing an HTTP POST transaction with the following data fields:

- USERID – imMTrax assigns the user ID.
- PASSWORD –imMTrax assigns a temporary password that must be changed by the provider site. Implementations must support passwords of at least 8 characters and have a combination of letters and digits and will not allow any repeating characters.
- MESSAGEDATA – The HL7 message as ASCII text. The message must begin with the character string "MSH".

The response content to the HTTP POST will be the appropriate HL7 message as required by *Implementation Guide for Immunization Data Transactions using Version 2.3.1 of the Health Level Seven (HL7) Standard Protocol.* The HL7 message will not be encapsulated in any way.

Transport Protocol for HL7 Messages over HTTPS using Digital Signatures

When using Digital Signatures for Authentication, application programs will contact the PHC-Hub server by issuing an HTTP POST transaction with the following data fields:

- USERNAME/PASSWORD – Each user needing HL7 upload access to PHC-Hub is given a specific user name and password. This user name and password is created in the imMTrax system as a user with this permission under the organization /site that is the sending entity. The user name and password is used to authenticate the sending system's access to PHC-Hub.
- MESSAGEDATA – The Message content will be the digitally signed HL7 message formatted in accordance S/MIME Version 2 specification available at http://www.ietf.org/rfc/rfc2311.txt.

The response content to the HTTP POST will be the appropriate HL7 message as required by *Implementation Guide for Immunization Data Transactions using Version 2.3.1 of the Health Level Seven (HL7) Standard.* Message content will be the digitally signed HL7 message formatted in accordance S/MIME Version 2.

HTTP VERSION AND RECOMMENDED HEADERS

Where possible, HTTP version 1.1 (http://www.ietf.org/rfc/rfc2616.txt) should be used for all client messages.

When HTTP messages are sent, intervening servers may cache responses to improve overall network response. Because the messages discussed here are dynamic queries and updates, cached results are likely to be incorrect or out of date. HL7 query id's should be unique and so should not be cached, but to avoid any possible interaction with caching servers, the no-cache directives should be used in all HTTP headers. In HTTP version 1.1, these take the form:

Cache-control: no-cache
In version 1.0, the equivalent is:
Pragma: no-cache

IIS SERVER LOOKUP SERVICE

Both public key infrastructure and IIS-to-IIS communication require a lookup service to link IIS with their public keys and http addresses.

Such a lookup (or directory) service should provide sufficient information to a client that the client could adequately determine the likely authoritative IIS given address information in an HL7 query message or "other previous residence" address hints.

The information returned should include addresses for the HL7 HTTP server and human technical contact, and the public key used to communicate authentication messages to the IIS.

The search information schema should include for each IIS:

A printable name for the IIS (ex: Montana State Immunization Information System)
The country the covered by the IIS's domain of service (ex: USA)
The state the IIS's domain of service covers (ex: MT)
If the IIS is not authoritative for the entire state:
    The list of counties the IIS is authoritative for (ex: Yellowstone)
    If the IIS is not authoritative for the entire county, or if there are cities outside the jurisdiction of any county for which the IIS is authoritative:
        The list of cities the IIS is authoritative for (ex: Billings, MT)

The returned data for a matching IIS should include:

The HTTP/HTTPS URL for the HL7 service
The X509 public key for the service
A human technical contact email address
A human technical contact telephone

BATCH UPLOADS VIA HTTPS

When batches of HL7 messages are sent via HTTP, they should be combined according to the HL7 Batch Protocol as described in by *Implementation Guide for Immunization Data Transactions using Version 2.3.1 of the Health Level Seven (HL7) Standard Protocol.* Batch uploads use the same specifications above, except that instead of the messages starting with "MSH", batches start with "FHS".

SOAP TRANSPORT

PHC-Hub is able to accept messages that are sent using the SOAP protocol.

Example I is a message sent with PHC-Hub specific information.

EXAMPLE I


```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelop
   <soap:Header/>
   <soap:Body>
      <urn:submitSingleMessage>
         <!--Optional:-->
         <urn:username>josh.akin3</urn:username>
         <!--Optional:-->
         <urn:password>tnt12345</urn:password>
         <!--Optional:-->
         <urn:facilityID>14629</urn:facilityID>
         <urn:hl7Message>MSH|^~\&amp;|TMI NextGen Rosetta|MT0001
PID|1||3300.3300^^^^MR||AKINOME^BAHRAINE^^^^MS||20010101|F||2028-
PD1|||^^|||||||||Y||
RXA|0|1|20130519||03^^CVX^90707^^CPT|1|1||01^New Immunization|la
RXR|ID|RA|
OBX|1|CE|30963-3^Vaccine purchased with^LN||PUBLIC^Public funds^
OBX|1|CE|VFC-STATUS^VFC Status^STC||V01^Medicaid^HL70064||||||F<
      </urn:submitSingleMessage>
   </soap:Body>
</soap:Envelope>
```

Example II is a sample from the CDC's EHR-IIS Interoperability Enhancement Project: SOAP Implementation Guide[2]

EXAMPLE II

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd"
             xmlns:wsp="http://www.w3.org/ns/ws-policy"
             xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
             xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
             xmlns:wsaw="http://www.w3.org/2005/08/addressing"
             xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
             xmlns:tns="urn:cdc:iisb:2011"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             targetNamespace="urn:cdc:iisb:2011"
             name="IISServiceNew">

  <!-- schema for types -->
  <types>
        <xsd:schema elementFormDefault="qualified" targetNamespace="urn:cdc:iisb:2011">

              <xsd:complexType name="connectivityTestRequestType">
                    <xsd:sequence>
                          <xsd:element name="echoBack" type="xsd:string" minOccurs="1"
maxOccurs="1" nillable="true"/>
                    </xsd:sequence>
              </xsd:complexType>

              <xsd:complexType name="connectivityTestResponseType">
                    <xsd:sequence>
                          <xsd:element name="return" type="xsd:string" minOccurs="1"
maxOccurs="1" nillable="true"/>
                    </xsd:sequence>
              </xsd:complexType>

              <xsd:complexType name="submitSingleMessageRequestType">
```

[2] http://www.cdc.gov/vaccines/programs/iis/technical-guidance/SOAP/downloads/transport-specification.pdf

```
                  <xsd:sequence>
                        <xsd:element name="username" type="xsd:string" minOccurs="0"
maxOccurs="1" nillable="true"/>
                        <xsd:element name="password" type="xsd:string" minOccurs="0"
maxOccurs="1" nillable="true"/>
                        <xsd:element name="null" type=”xsd:string"  minOccurs="0"
maxOccurs="1" nillable="true"/>
                        <xsd:element name="hl7Message" type="xsd:string" minOccurs="1"
maxOccurs="1" nillable="true"/>
                  </xsd:sequence>
            </xsd:complexType>

            <xsd:complexType name="submitSingleMessageResponseType">
                  <xsd:sequence>
                        <xsd:element name="return" type="xsd:string" minOccurs="1"
maxOccurs="1" nillable="true"/>
                  </xsd:sequence>
            </xsd:complexType>

            <xsd:complexType name="soapFaultType">
               <xsd:sequence>
                     <xsd:element name="Code" type="xsd:integer" minOccurs="1"/>
                     <xsd:element name="Reason" type="xsd:string" minOccurs="1"/>
                     <xsd:element name="Detail" type="xsd:string" minOccurs="1"/>
               </xsd:sequence>
            </xsd:complexType>

            <xsd:complexType name="UnsupportedOperationFaultType">
               <xsd:sequence>
                     <xsd:element name="Code" type="xsd:integer" minOccurs="1"/>
                     <xsd:element name="Reason" fixed="UnsupportedOperation"/>
                     <xsd:element name="Detail" type="xsd:string" minOccurs="1"/>
               </xsd:sequence>
            </xsd:complexType>

            <xsd:complexType name="SecurityFaultType">
               <xsd:sequence>
                     <xsd:element name="Code" type="xsd:integer" minOccurs="1"/>
                     <xsd:element name="Reason" fixed="Security"/>
                     <xsd:element name="Detail" type="xsd:string" minOccurs="1"/>
               </xsd:sequence>
            </xsd:complexType>

            <xsd:complexType name="MessageTooLargeFaultType">
               <xsd:sequence>
                     <xsd:element name="Code" type="xsd:integer" minOccurs="1"/>
                     <xsd:element name="Reason" fixed="MessageTooLarge"/>
                     <xsd:element name="Detail" type="xsd:string" minOccurs="1"/>
               </xsd:sequence>
            </xsd:complexType>

            <xsd:element name="connectivityTest" type="tns:connectivityTestRequestType"/>
            <xsd:element name="connectivityTestResponse"
type="tns:connectivityTestResponseType"/>
            <xsd:element name="submitSingleMessage"
type="tns:submitSingleMessageRequestType"/>
            <xsd:element name="submitSingleMessageResponse"
type="tns:submitSingleMessageResponseType"/>
            <xsd:element name="fault" type="tns:soapFaultType"/>
            <xsd:element name="UnsupportedOperationFault"
type="tns:UnsupportedOperationFaultType"/>
            <xsd:element name="SecurityFault" type="tns:SecurityFaultType"/>
            <xsd:element name="MessageTooLargeFault" type="tns:MessageTooLargeFaultType"/>

    </xsd:schema>
  </types>
```

```
  <!-- Message definitions -->
  <message name="connectivityTest_Message">
    <documentation>connectivity test request</documentation>
    <part name="parameters" element="tns:connectivityTest" />
  </message>

  <message name="connectivityTestResponse_Message">
    <documentation>connectivity test  response</documentation>
    <part name="parameters" element="tns:connectivityTestResponse" />
  </message>

  <message name="submitSingleMessage_Message">
    <documentation>submit single message request.</documentation>
    <part name="parameters" element="tns:submitSingleMessage" />
  </message>

  <message name="submitSingleMessageResponse_Message">
    <documentation>submit single message response</documentation>
    <part name="parameters" element="tns:submitSingleMessageResponse" />
  </message>

  <message name="UnknownFault_Message">
    <part name="fault" element="tns:fault"/>
  </message>

  <message name="UnsupportedOperationFault_Message">
       <part name="fault" element="tns:UnsupportedOperationFault"/>
  </message>

  <message name="SecurityFault_Message">
    <part name="fault" element="tns:SecurityFault"/>
  </message>
  <message name="MessageTooLargeFault_Message">
    <part name="fault" element="tns:MessageTooLargeFault"/>
  </message>

  <!-- Operation/transaction declarations -->
  <portType name="IIS_PortType">
    <operation name="connectivityTest">
      <documentation>the connectivity test</documentation>
      <input message="tns:connectivityTest_Message"
wsaw:Action="urn:cdc:iisb:2011:connectivityTest"/>
      <output message="tns:connectivityTestResponse_Message"
wsaw:Action="urn:cdc:iisb:2011:connectivityTestResponse"/>
      <fault name="UnknownFault" message="tns:UnknownFault_Message"/>    <!-- a general soap
fault -->
      <fault name="UnsupportedOperationFault"
message="tns:UnsupportedOperationFault_Message"/>    <!-- The UnsupportedOperation soap
fault -->
    </operation>

    <operation name="submitSingleMessage">
      <documentation>submit single message</documentation>
      <input message="tns:submitSingleMessage_Message"
wsaw:Action="urn:cdc:iisb:2011:submitSingleMessage"/>
      <output message="tns:submitSingleMessageResponse_Message"
wsaw:Action="urn:cdc:iisb:2011:submitSingleMessageResponse"/>
      <fault name="UnknownFault" message="tns:UnknownFault_Message"/>    <!-- a general soap
fault -->
      <fault name="SecurityFault" message="tns:SecurityFault_Message"/>
      <fault name="MessageTooLargeFault" message="tns:MessageTooLargeFault_Message"/>
    </operation>
  </portType>

  <!-- SOAP 1.2 Binding -->
```

```
  <binding name="client_Binding_Soap12" type="tns:IIS_PortType">
    <soap12:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
    <operation name="connectivityTest">
      <soap12:operation soapAction="urn:cdc:iisb:2011:connectivityTest" />
      <input><soap12:body use="literal" /></input>
      <output><soap12:body use="literal" /></output>
      <fault name="UnknownFault"><soap12:fault use="literal" name="UnknownFault"/></fault>
      <fault name="UnsupportedOperationFault"><soap12:fault use="literal"
name="UnsupportedOperationFault"/></fault>
    </operation>
    <operation name="submitSingleMessage">
      <soap12:operation soapAction="urn:cdc:iisb:2011:submitSingleMessage" />
      <input><soap12:body use="literal" /></input>
```

REFERENCE IMPLEMENTATIONS

The working group proposed the creation of reference implementations demonstrating the protocols described herein. The purpose of the reference implementation is to provide examples that may be used as starting points by registry developers in implementing the protocols in this standard. The following are general principles for the reference implementations:

1. The reference implementations shall be open source.
2. The reference implementations should avoid, to the extent possible, registry-specific business logic, and should concentrate on the protocols.
3. The reference implementations should provide simple interfaces for authentication and message logging by external routines to be provided by the specific registry implementers.