INTRODUCTION

The Georgia Registry of Immunization Transactions and Services (GRITS) currently allows two methods of Real time data exchange with public and private providers a Web Service and PHINMS (Public Health Information Network Messaging System). Although both methods provide secure communication with GRITS, choosing the right version for you will vary based on your existing IT infrastructure. Once you've chosen which interface type will work best for you and have completed the necessary forms (described in Section I) please proceed to Section II Web Service Installation or Section III PHINMS Installation. Regardless of which method you choose, HL7 versions 2.4 and 2.5.1 can be utilized for real time communication with GRITS.

To begin sending Real time transactions to GRITS, please follow the steps below:

- **I. Complete the Necessary GRITS Forms**
- **II. GRITS Web Service Installation (if applicable)**
- **III. PHINMS Installation (if applicable)**
  - **i. Export/Install the GRITS SSL Certificate**
  - **ii. Install the PHINMS Client Software**
  - **iii. Import the GRITS SSL Certificate into PHINMS**
  - **iv. Configuring the Test Route**
  - **v. Configure Folder Polling**
  - **vi. Configure GRITS Response File location**
  - **vii. PHINMS Testing**
  - **viii. PHINMS Production Configuration**

I. COMPLETE THE NECESSARY GRITS FORMS

FORM 1 – GRITS SOFTWARE USER AGREEMENT:

If your provider organization is new to GRITS, a GRITS Software User Agreement must be completed. This form will be used to uniquely identify your organization, the organizations primary point of contact, etc. **GRITS credentials will NOT be provided until Software User Agreement has been completed**. Please contact one of the GRITS Business Analysts, **Nikki Griffin at njgriffin@dhr.state.ga.us** or **Patrice Wade at rpwade@dhr.state.ga.us** for the GRITS Software User Agreement.

FORM 2 – BUSINESS ASSOCIATE AGREEMENT (BAA):

If your interface is being installed by a third party vendor or anyone outside of your organization, it is required that a Business Associate Agreement (BAA) be signed. The BAA is between the Immunization Registry (which is a HIPAA Covered Entity) and the software vendor or individual that will be exposed to Protected Health Information (PHI). This agreement is necessary to ensure vendor or individual fulfills the responsibilities associated with protecting this very sensitive information. Although testing can proceed without a BAA, you **will not be allowed to proceed with production installation until a Business Associate Agreement has been completed.** Please contact **Nikki Griffin at njgriffin@dhr.state.ga.us** or **Patrice Wade at rpwade@dhr.state.ga.us** for the Business Associate Agreement.

II. GRITS Web Service Installation (using CDC WSDL)

To utilize the GRITS Web service, you must contact the GRITS Business Analyst who will provide you with the following:

1) Username
2) Password
3) Facility ID (GRITS Org Short Name)

If you plan to utilize the GRITS Web Service, you do **NOT** need to configure PHINMS. Your web service must be configured to send the above values, as well as the appropriate HL7 payload. Although you will not be transmitting information to the CDC, the CDC WSDL is considered a de-facto standard for such interfaces and is the one that GRITS has chosen to utilize.

The URLs for the Web Service in staging (test) and production are:
Test URL: https://www.gritstest.state.ga.us/gritsws/client_Service
Production URL: https://www.grits.state.ga.us/gritsws/client_Service

Using the staging web service you must thoroughly test your interface. Once the GRITS staff approves the messages you've sent to Staging, you will then be granted production privileges. For ease of configuration, your production and staging credentials will be identical; however, we will not activate your production credentials until your testing is complete.

For more information on the CDC WSDL please access the CDC website at http://www.cdc.gov/vaccines/programs/iis/technical-guidance/SOAP/wsdl.html. The CDC WSDL is listed below.

```
<?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">
                  <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="facilityID" 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 -->
```

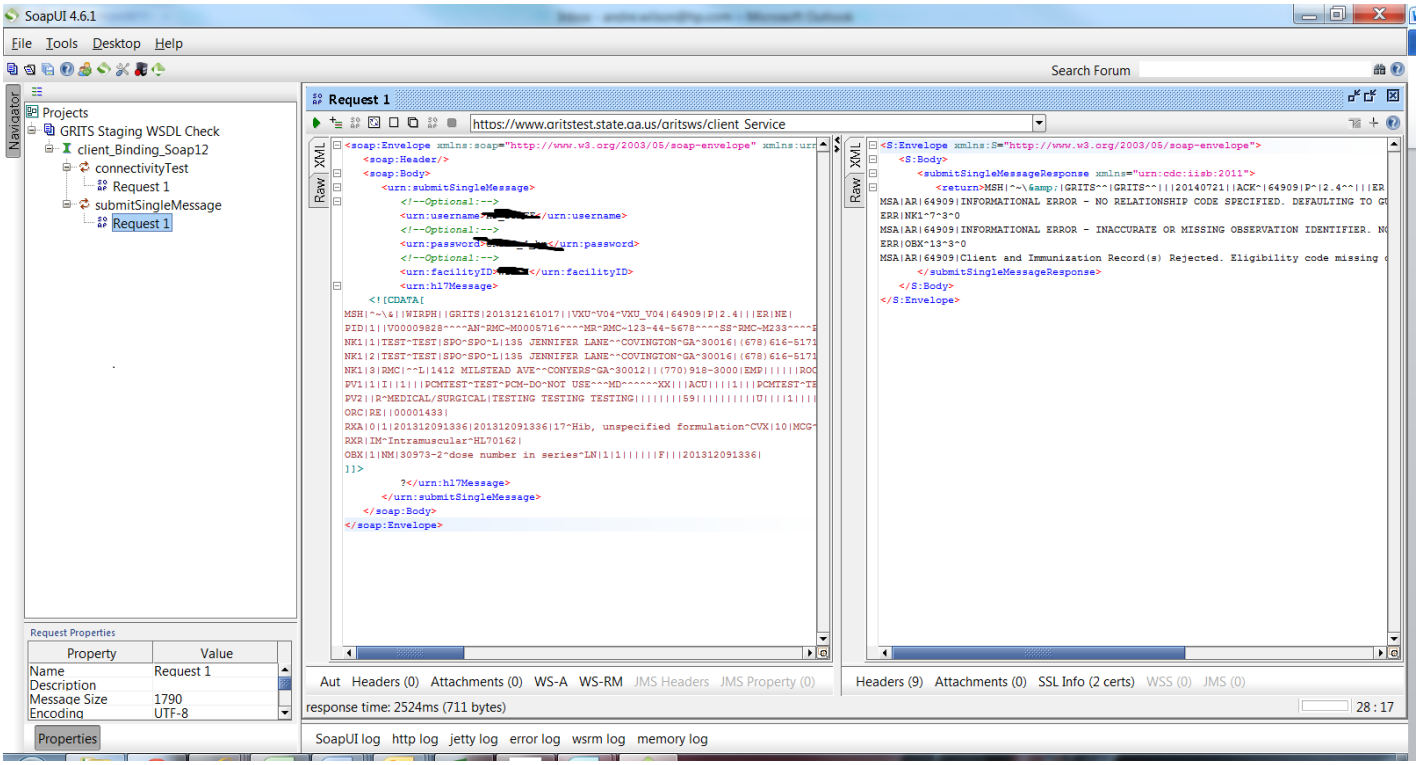
```
      <fault name="UnsupportedOperationFault"
message="tns:UnsupportedOperationFault_Message"/>     <!-- The UnsupportedOperation soap
fault -->
    </operation>

    <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>
      <output><soap12:body use="literal" /></output>
      <fault name="UnknownFault"><soap12:fault use="literal"
name="UnknownFault"/></fault>
      <fault name="SecurityFault"><soap12:fault use="literal"
name="SecurityFault"/></fault>
      <fault name="MessageTooLargeFault"><soap12:fault use="literal"
name="MessageTooLargeFault"/></fault>
    </operation>
  </binding>

  <!-- Service definition -->
  <service name="client_Service">
    <port binding="tns:client_Binding_Soap12" name="client_Port_Soap12">
      <soap12:address location="http://localhost/WebApp/IISService" />
    </port>
  </service>
</definitions>
```

Sending Messages via the Web Service:

The GRITS team uses SoapUI to send messages via the webservice. If you would like to use SoapUI, it can be downloaded at http://sourceforge.net/projects/soapui/files/. Since SoapUI is the product used by the GRITS team, the instructions that follow are based on the use of SoapUI.

Once installed, open and click "File→New SOAP Project". Choose whatever you want for the Project Name (like grits-prod or grits-stage), then for the Initial WSDL enter the URL I had sent you (for either prod or stage) followed by "?wsdl". Then click the Ok button.

Once it opens, in the Projects list on the left of your screen go down to the SubmitSingleMessage→Request 1 and double click it. That should open the Request 1 window. Enter the URL for that regions service where I circled it in RED. Put the three credentials above where I've painted PURPLE, and your HL7 message within a <![CDATA[ ]]> tag like in the picture below. Then click the green button that I circled in YELLOW. It should take a second or two, and your HL7 response should then appear on the right side of the Request 1 window. FYI…The VXQ listed below does not contain PHI.



III. PHINMS Installation

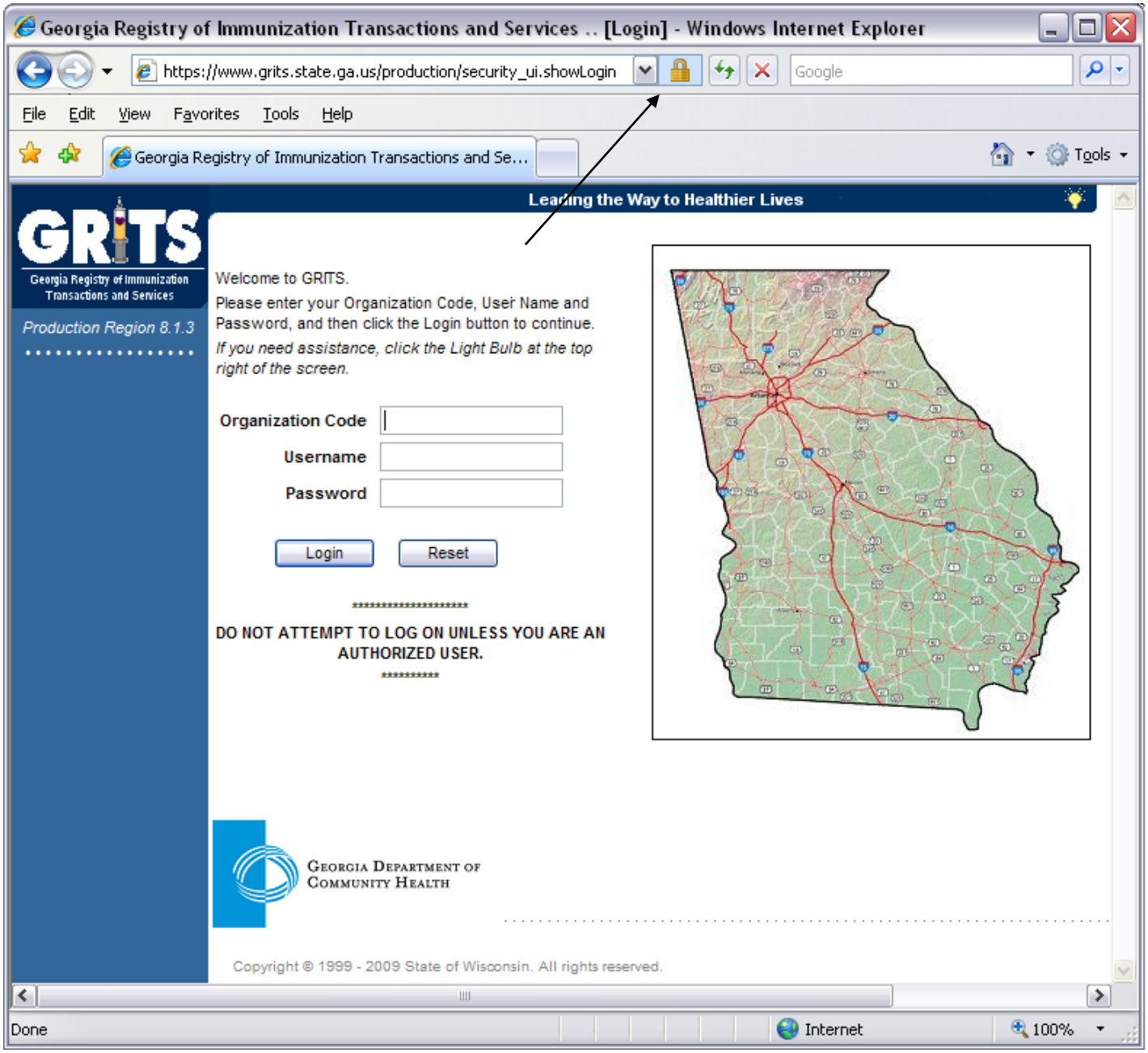
PHINMS is the **P**ublic **H**ealth **I**nformation **N**etwork **M**essaging **S**ystem (pronounced FIN-M-S); it was developed by the CDC as a method to provide secure data transmission. In the state of Georgia, PHINMS is used by public and private providers to securely send HL7 (version 2.4) messages to and from the **G**eorgia **R**egistry of **I**mmunization **T**ransactions and **S**ervices (GRITS). This document contains instructions for installing the PHINMS Client application and configuring it to communicate with GRITS. If you plan to utilize PHINMS, you do **NOT** need to configure the Web Service. The required steps for configuring PHINMS are as follows:

i. EXPORT THE GRITS SSL CERTIFICATE

The following instructions describe the process for obtaining the GRITS SSL certificate using Internet Explorer. Instructions for importing the certificate into the PHINMS client certificate store are also given. If you are not using the PHINMS client software version 2.8.0.1 or higher, follow the export instructions and contact your company technical support team for help with importing the certificate file into your company certificate store.

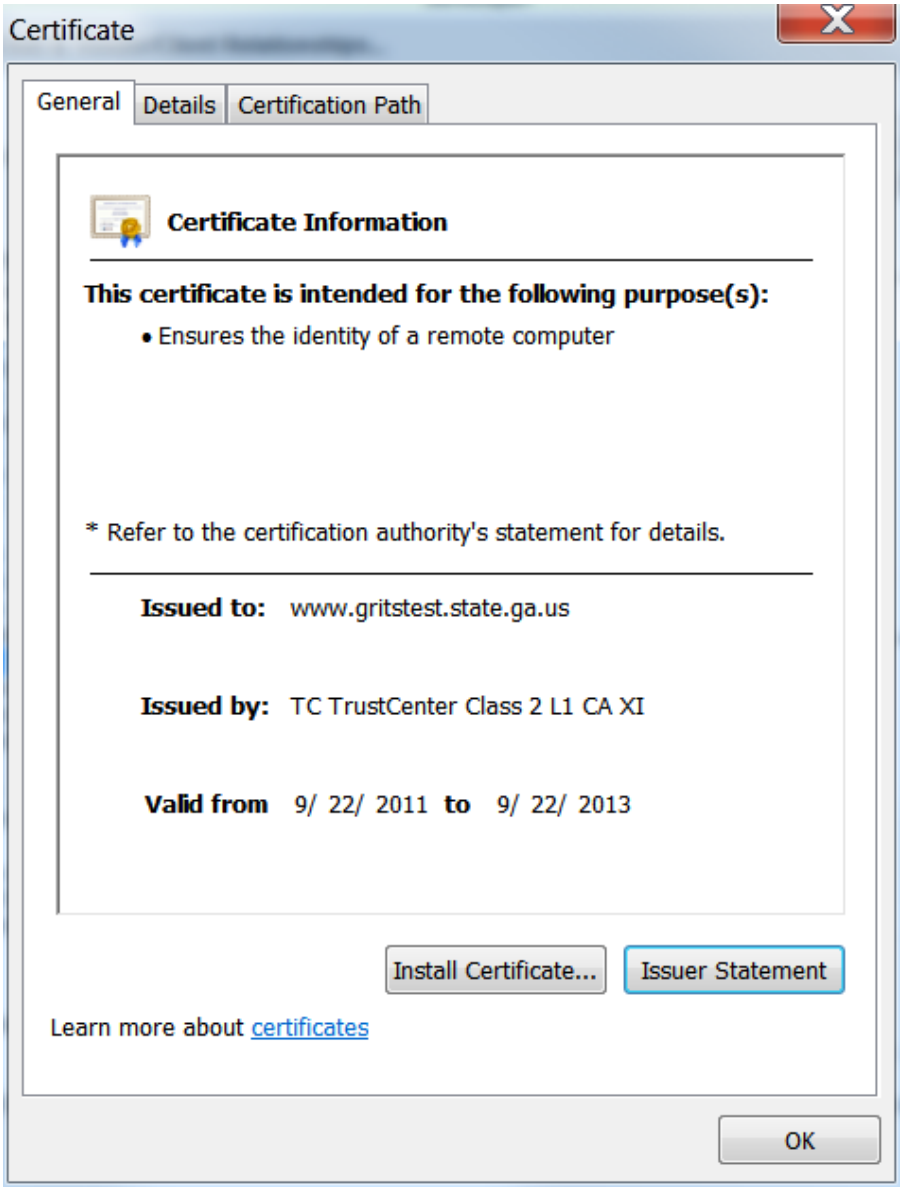
For testing, log into: https://www.gritstest.state.ga.us/

For production, log into: https://www.grits.state.ga.us/



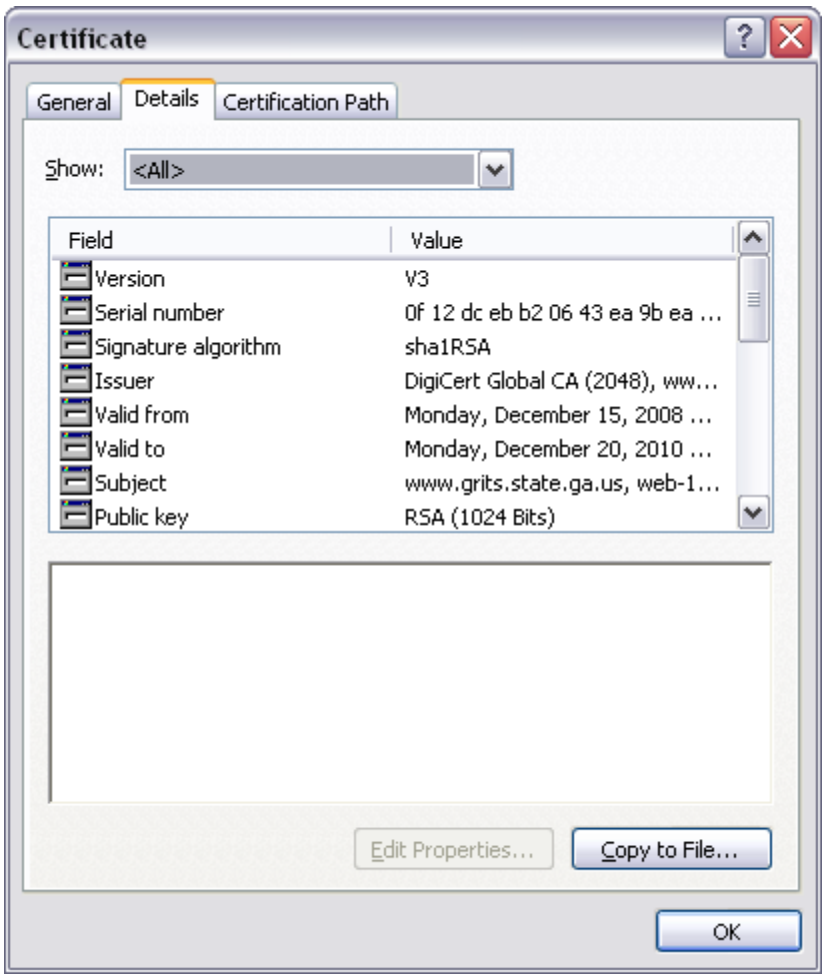
If presented with a Certificate Prompt, select Yes. (This prompt will appear only for first time users.) Double-click on the locked padlock icon on your screen. The location of the padlock will vary depending which browser version is being used. A Certificate window will appear (see below).

Certificate window



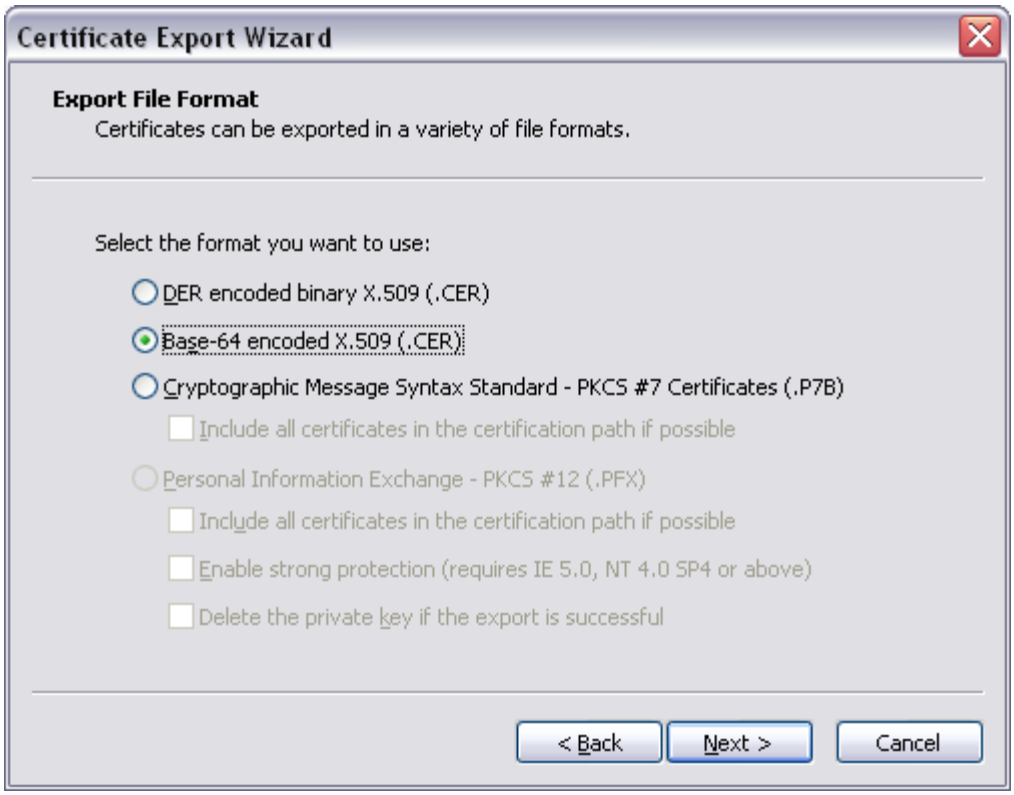
Click on the **Details** tab at the top of the screen.

Details tab



Click on the **Copy to File...** button in the lower-right corner of the screen.

Click **Next >**



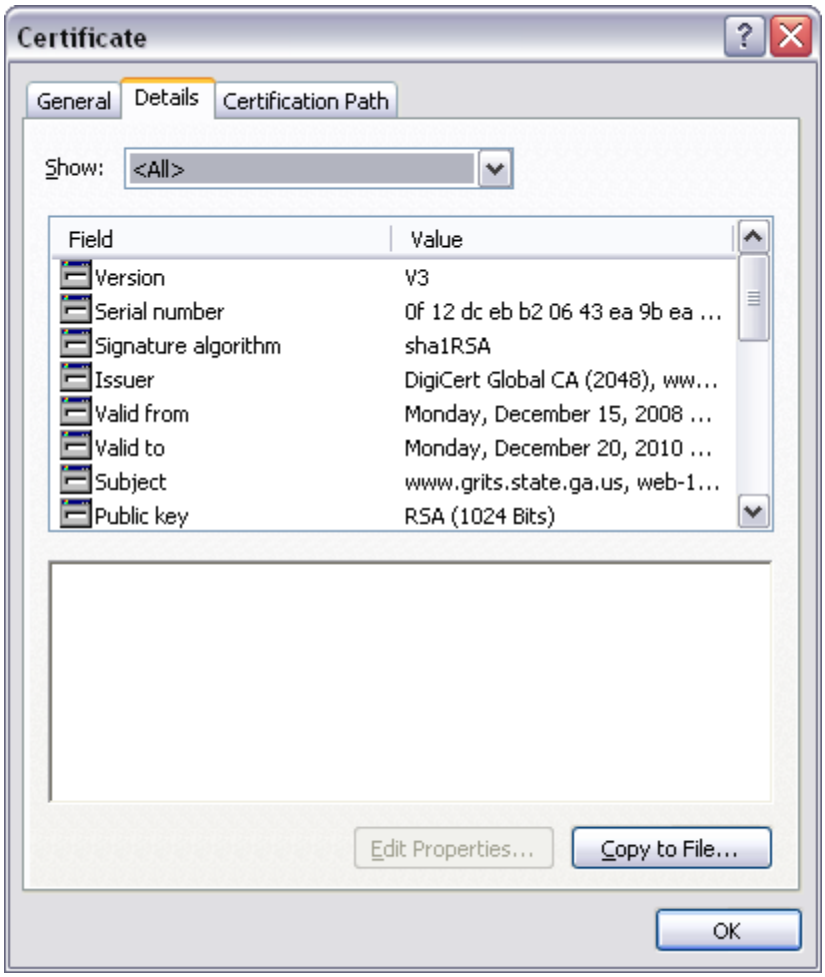
Click the **Base-64 encoded X.509 (.CER)** radio button, then click **Next >**

Type a file name to contain the exported certificate.
(Take note of the path and file name; they will be needed in a later step when importing the certificate.)
Click **Next >**

Click **Finish.**

Click **OK.**

Details screen



Click **OK.**

IF YOU RUN INTO TRANSMISSION ISSUES, PLEASE EXPORT THE GRITS ROOT AND INTERMEDIATE SSL CERTIFICATES BY SELECTING THE APPROPRIATE CERT ON THE CERTIFICATION PATH TAB

ii. INSTALL THE PHINMS CLIENT SOFTWARE

For the purposes of your installation, the process has been streamlined, however, if you'd like to view the information published by the CDC regarding PHINMS please do so via the following link: http://www.cdc.gov/phin/tools/PHINms/installation.html .

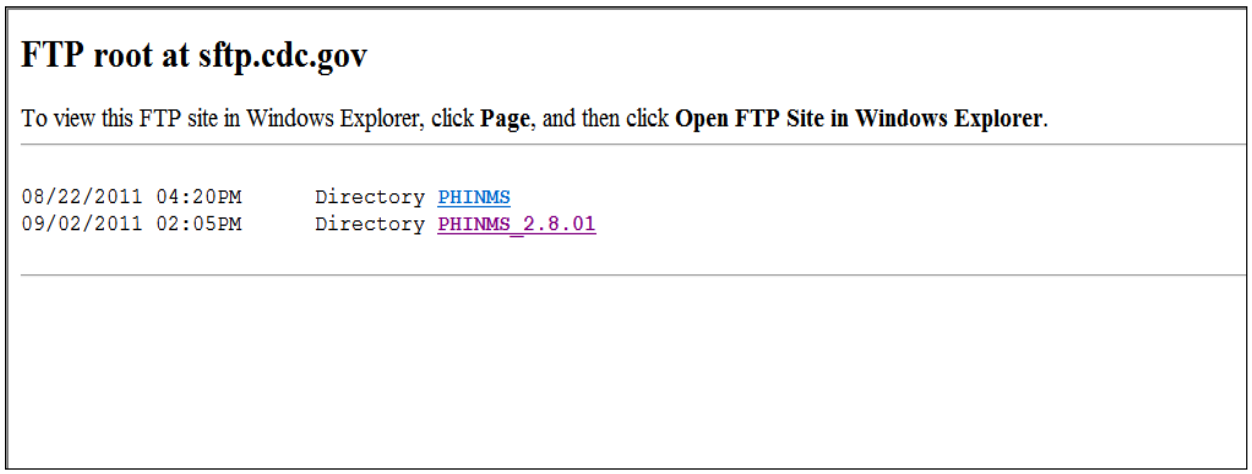
A Java Runtime Environment (JRE) is required for PHINMS. If you don't have Java on your server, please download the **32-bit version** at www.java.com.

Once you've confirmed and/or installed Java, please proceed to log into the CDCs FTP server and download the latest version of PHINMS using the following link:

ftp://sftp.cdc.gov

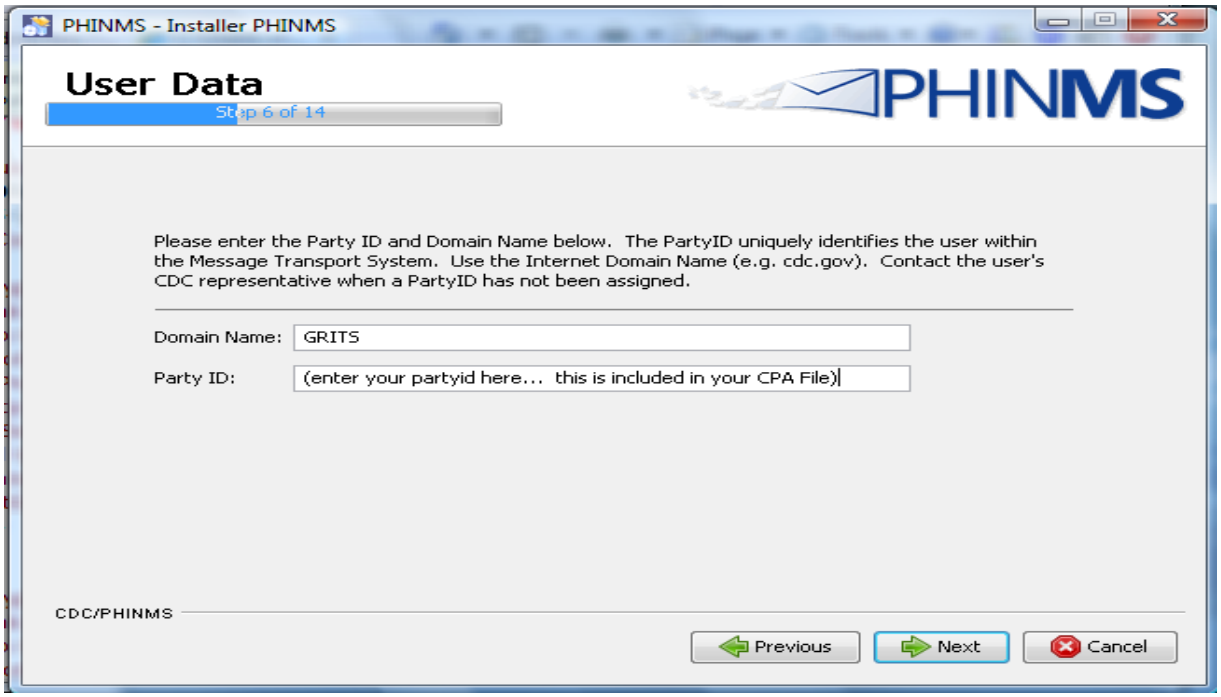
Login: `phinusr` *<case sensitive>*
Password: 8GzGB6TP *<case sensitive>*

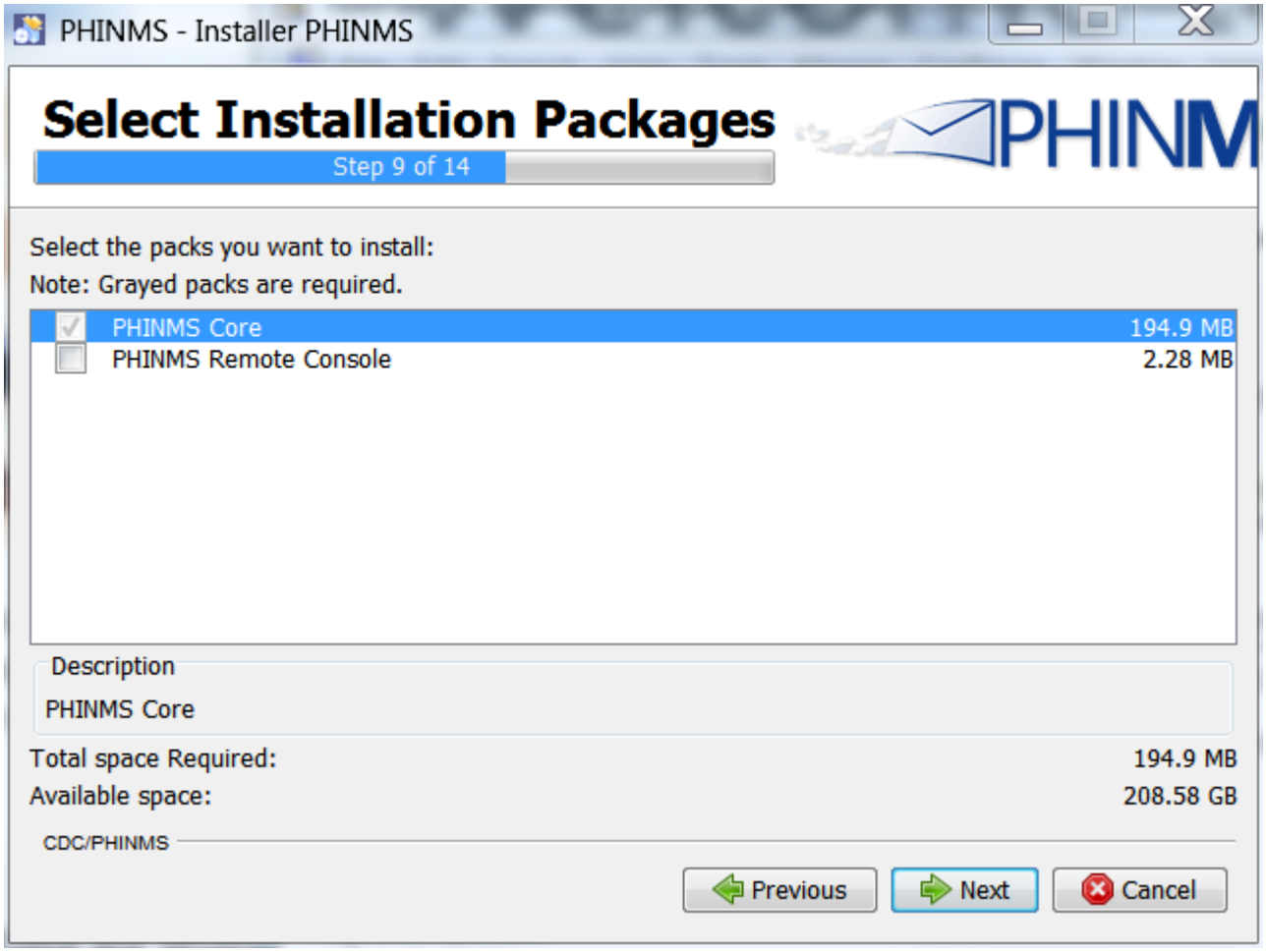
From here, select the latest version of the PHINMS Installs (*at the time of this document 2.8.0.1 was the latest version, but a newer version may now be available)*

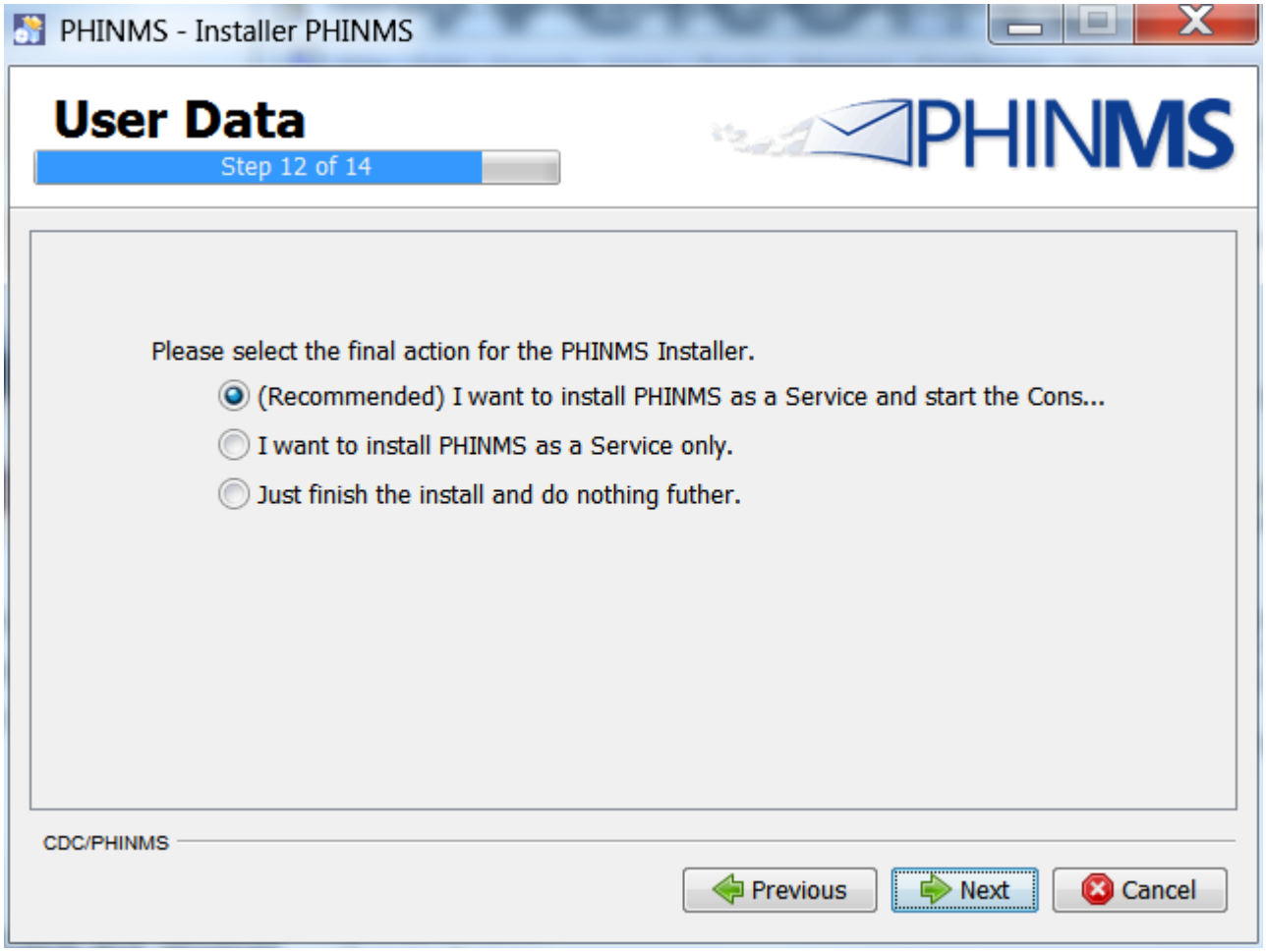


Proceed to installing PHINMS on your server… During the installation process, when prompted for the domain and partyid, please supply the following:

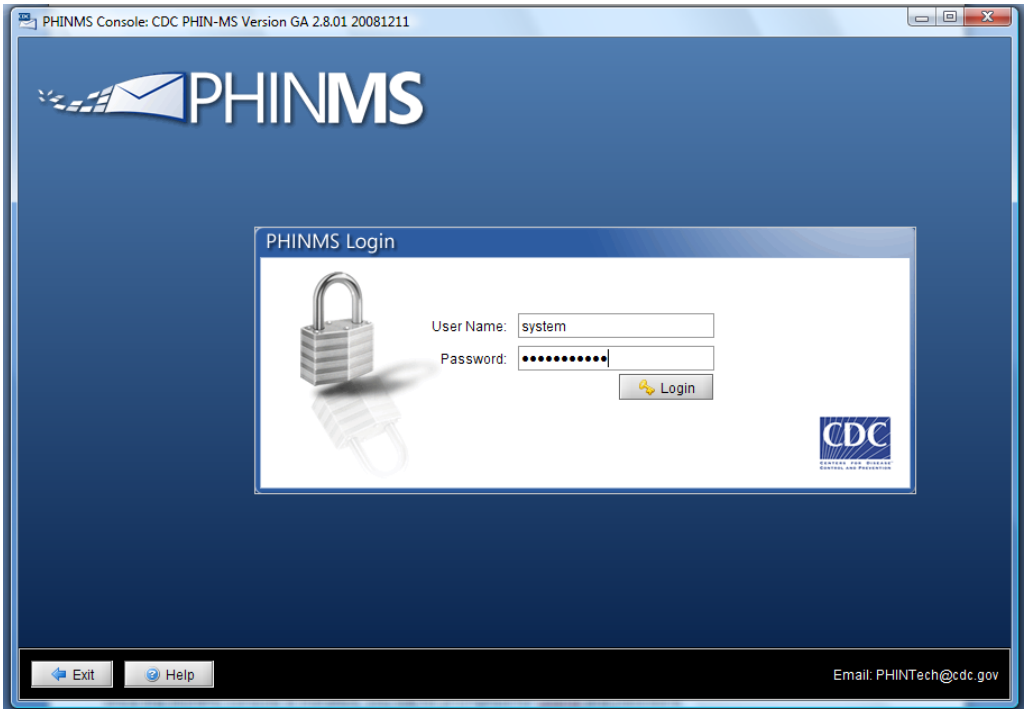
IMPORTANT: ***YOU MUST USE THE PARTYID PROVIDED BY THE GRITS TEAM HERE. USE OF ANY PARTYID OTHER THAN THE ONE SUPPLIED BY GRITS WILL REQUIRE YOU TO UNINSTALL THEN REINSTALL PHINMS USING THE CORRECT PARTY ID.***




PHINMS - Installer PHINMS
Select Installation Packages
Step 9 of 14
PHINM
Select the packs you want to install:
Note: Grayed packs are required.
PHINMS Core 194.9 MB
PHINMS Remote Console 2.28 MB
Description
PHINMS Core
Total space Required: 194.9 MB
Available space: 208.58 GB
CDC/PHINMS
Previous
Next
Cancel


PHINMS - Installer PHINMS
User Data
Step 12 of 14
PHINMS
Please select the final action for the PHINMS Installer.
(Recommended) I want to install PHINMS as a Service and start the Cons...
I want to install PHINMS as a Service only.
Just finish the install and do nothing futher.
CDC/PHINMS
Previous
Next
Cancel

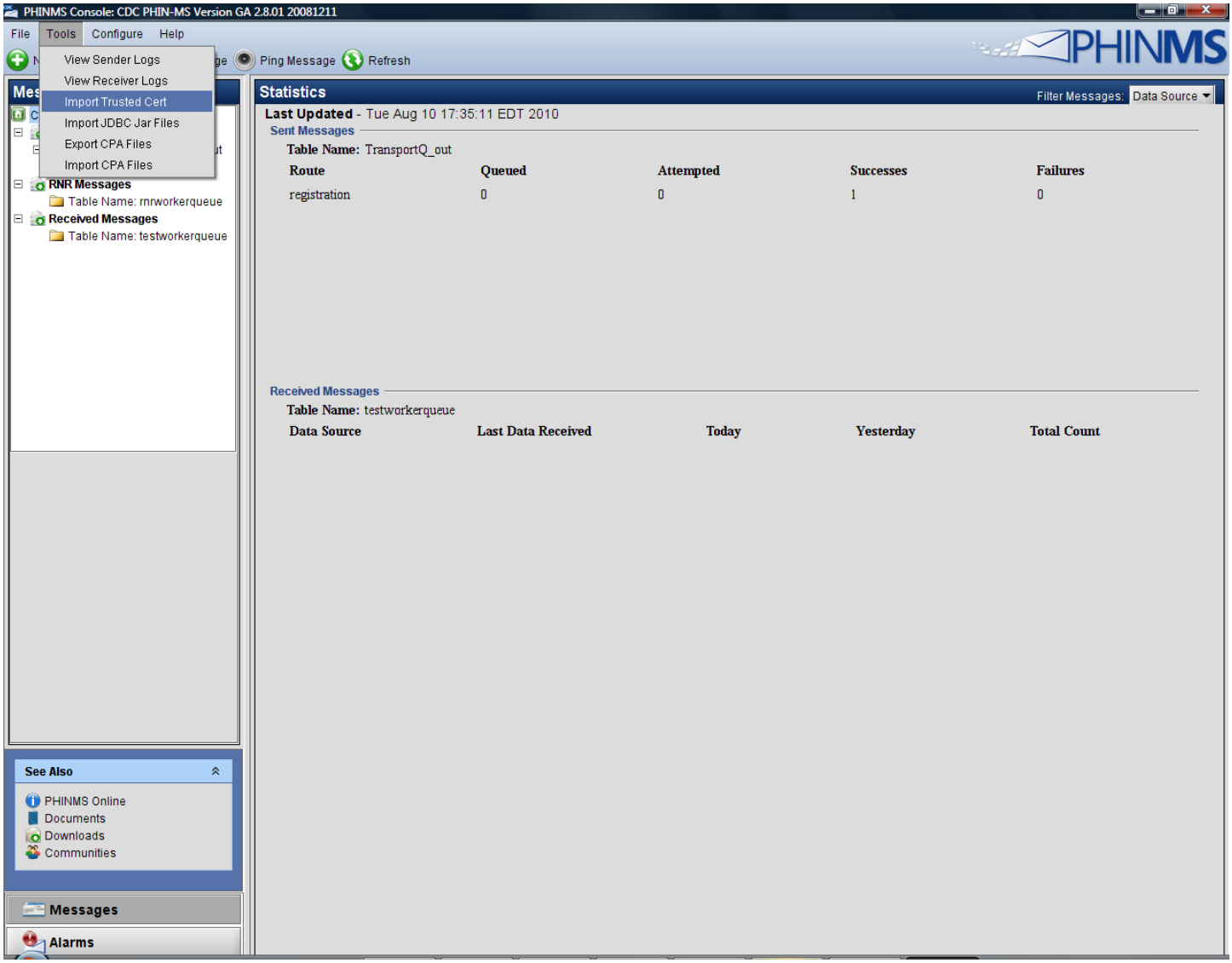
Once the PHINMS console is installed, you will be prompted for userid and password.



Please provide the following:
User Name: system <*case sensitive*>
Password: Phinms123 <*case sensitive*>

iii. IMPORTING THE GRITS SSL CERTIFICATE

Select Tools ->Import Trusted Cert



From here, navigate to the location of the GRITS cert imported in the previous step and select “Ok”.

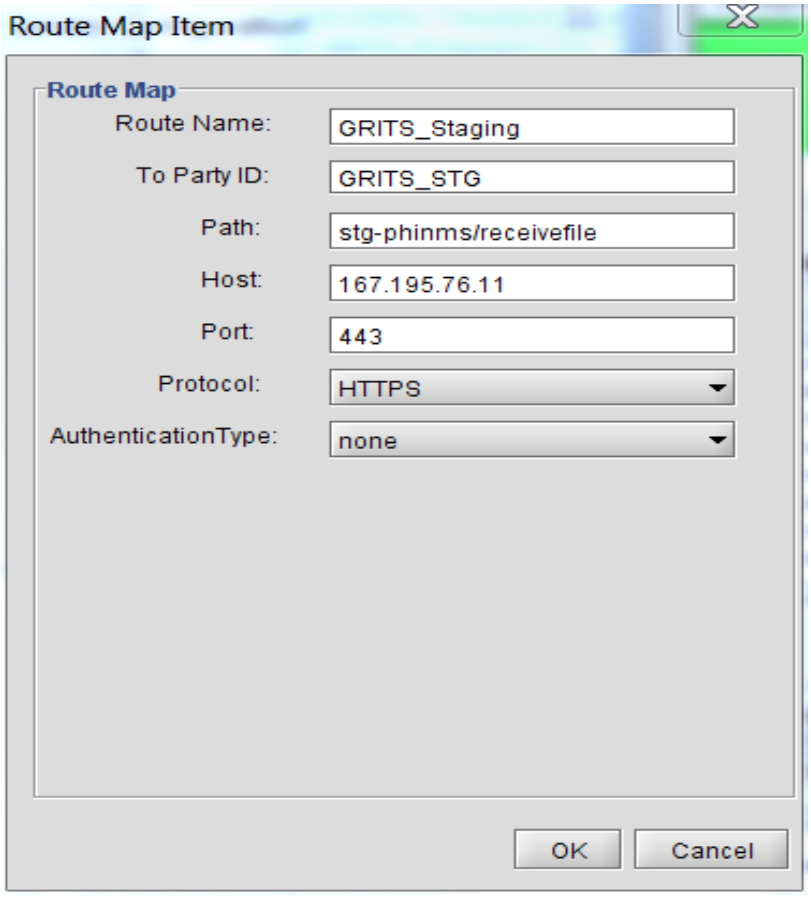
iv. CONFIGURING THE TEST ROUTE

Select configure →sender → routemap → add

Your routemap must configured EXACTLY as follows:

i) For Route Name enter: **GRITS_Staging**
ii) For the "To Party ID" Enter: **GRITS_STG**
iii) For the "Path" enter: **stg-phinms/receivefile**
iv) For the "Host" use the IP address: **167.195.76.11**

Once complete, the staging routemap should look like this:



v. CONFIGURING FOLDER POLLING

When "folder polling" is selected, anytime a new message is placed in the "outgoing" folder (user defined), PHINMS will send the record to GRITS. Once the record is processed, it is moved from the outgoing folder to the processed folder (user defined).

Select configure →sender →folder polling → add

Required fields are indicated by the red asterisk *

i) The Name should be something that indicates you are connection to your respective environment. In this case we used "GRITS_STG_POLL" to indicate Polling properties for the GRITS Staging environment

ii) Select the staging route from the dropdown list

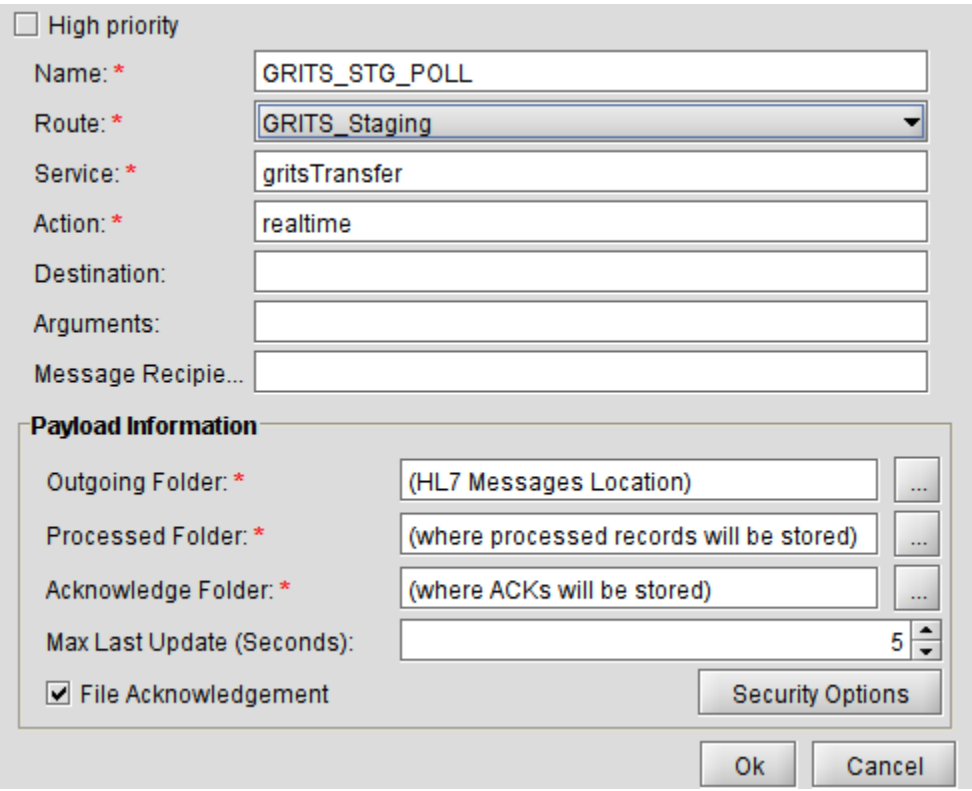
iii) The Service MUST be "gritsTransfer" *<case sensitive>*

iv) The Action MUST be "realtime" *<case sensitive>*

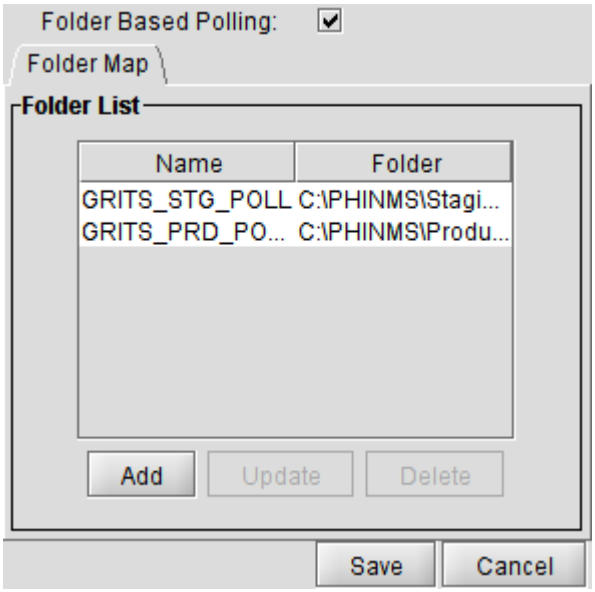
v) Outgoing folder is the location where the messages you intend to send to GRITS from your production system are stored

vi) The Processed Folder is the location where outgoing messages are moved to once they have been processed by PHINMS

vii) Acknowledge Folder is the location where acknowledgement records will be sent. For detailed error messages that come from GRITS, please access messages located in : <installation directory>shared/senderincoming



viii) Confirm that the “Folder Based Polling” box is checked in the Sender Configuration menu

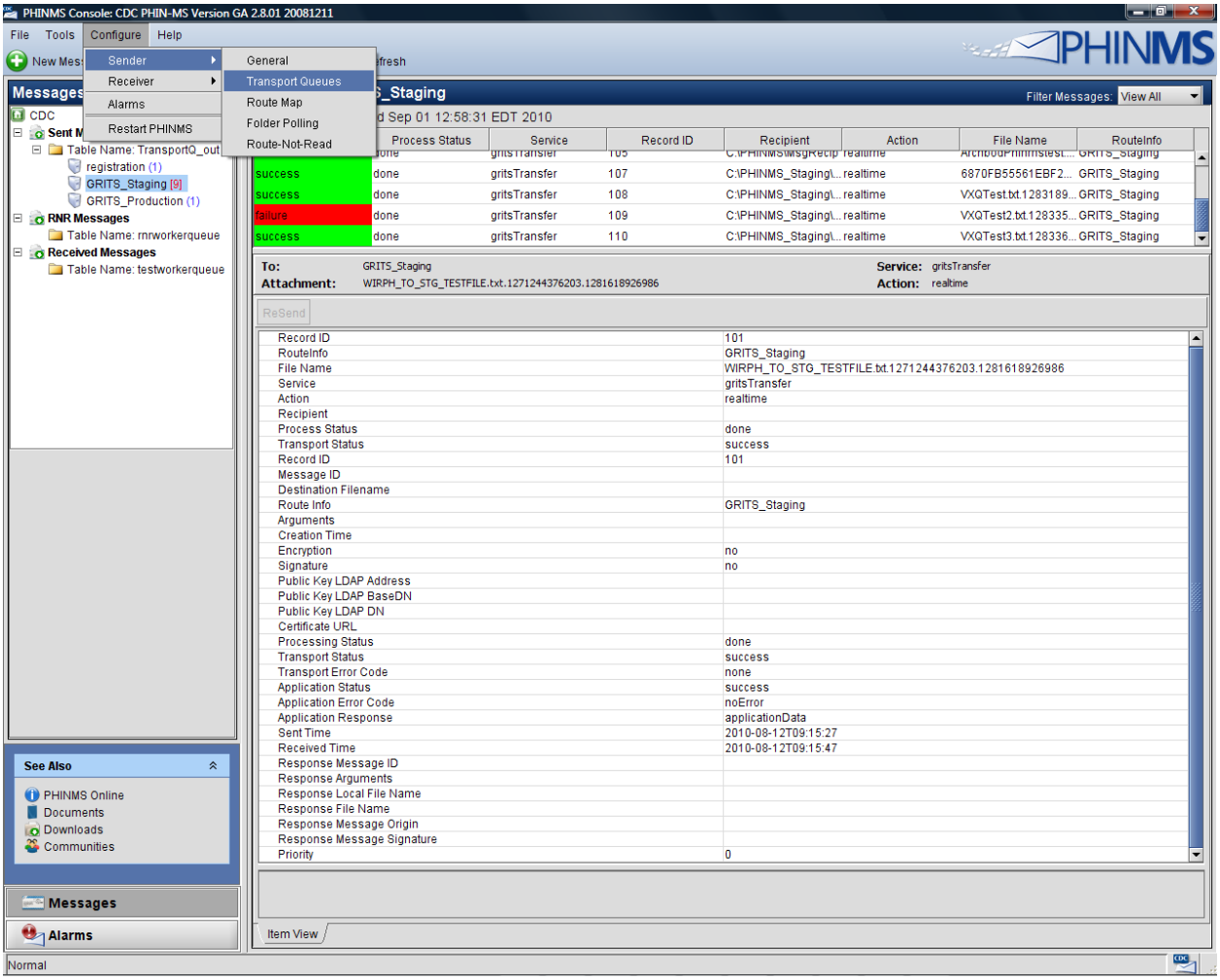


vi. CONFIGURING GRITS RESPONSE FILES IN PHINMS

HSQLDB is the default database and it has default settings. If folder polling is used, and you would like to see the GRITS responses to the HL7 files that are transmited, these defaults need to change. To see GRITS responses, please do the following:

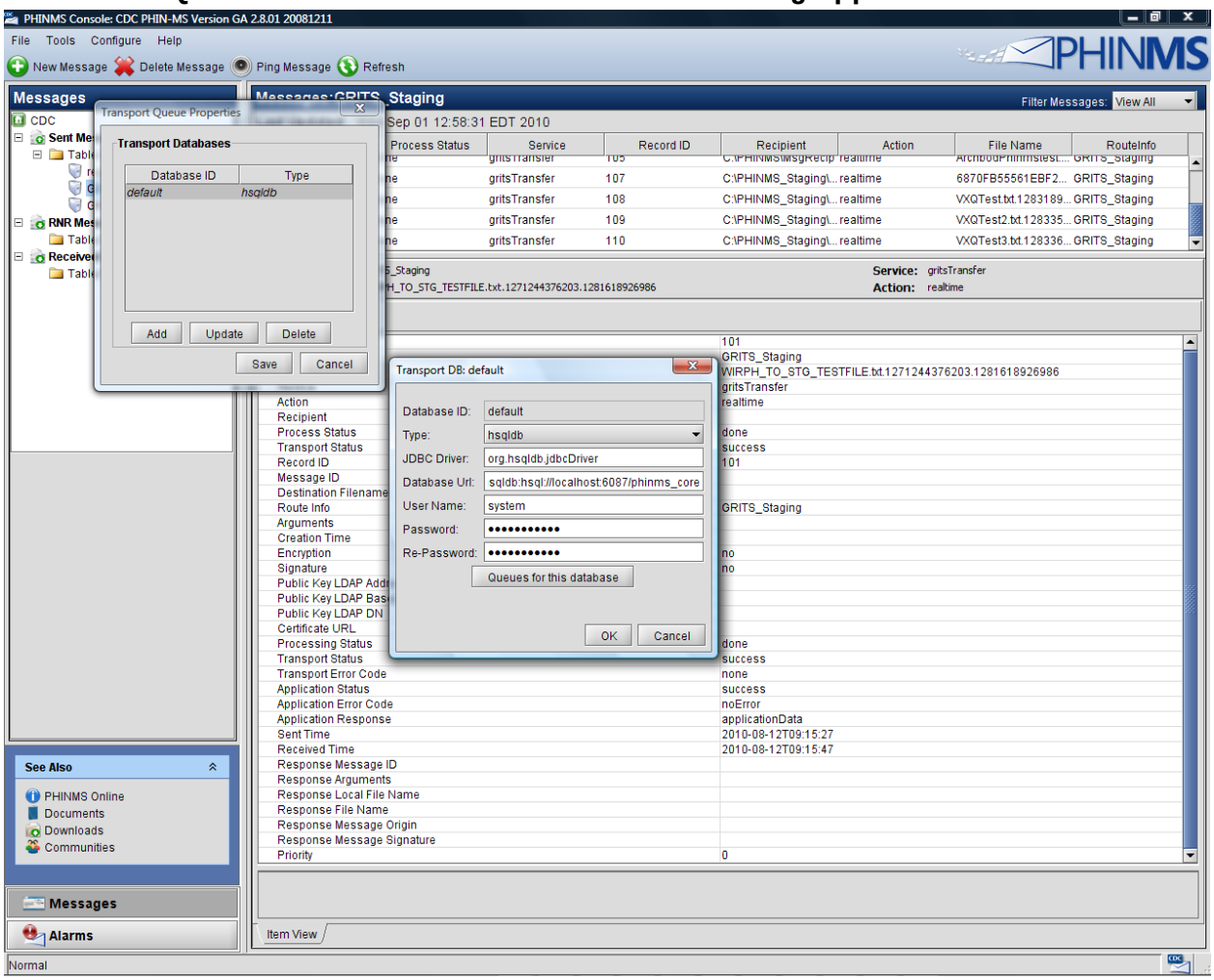
Select Sender →Transport Queues →

The following appears:

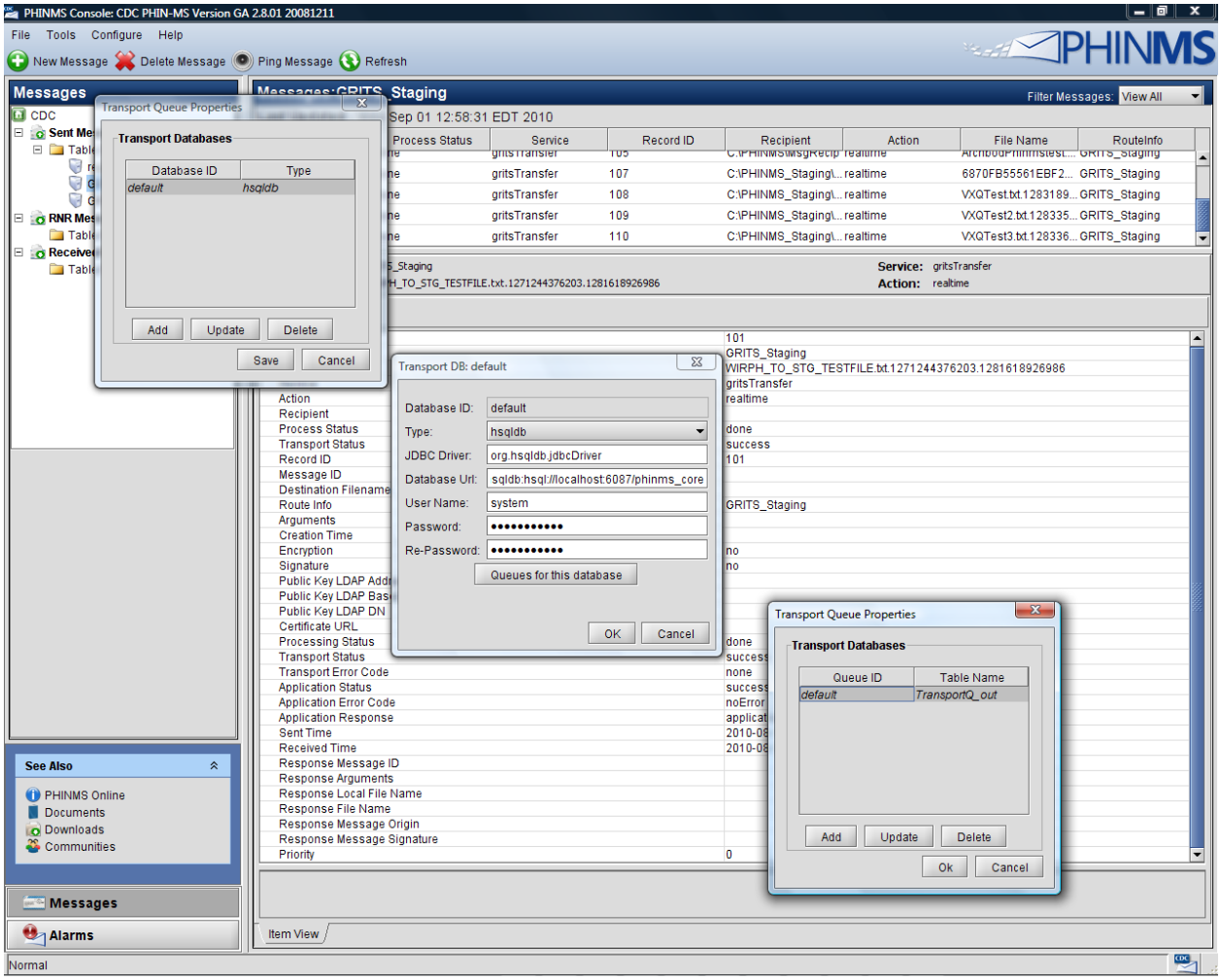


Select the hsqldb default and click "Update"

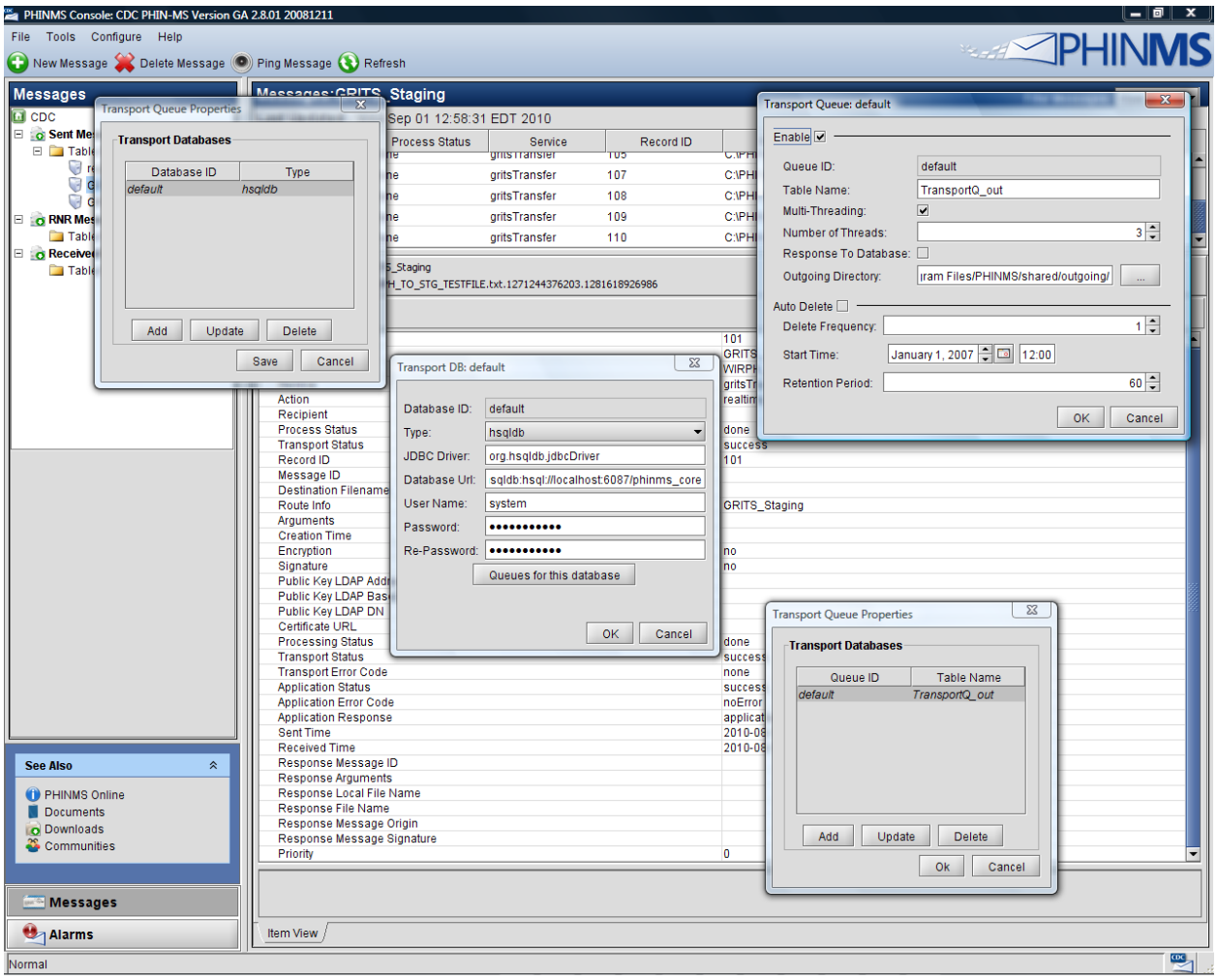
Then click "Queues for this database" and the following appears…



From here, select the transport queue that needs to be updated and select "Update"



The "Response To Database" checkbox is select by default. UNCHECK the "Response To Database" checkbox so that responses are returned to a directory on your server.



Response files are place in the [installation folder]\shared\senderincoming\ directory

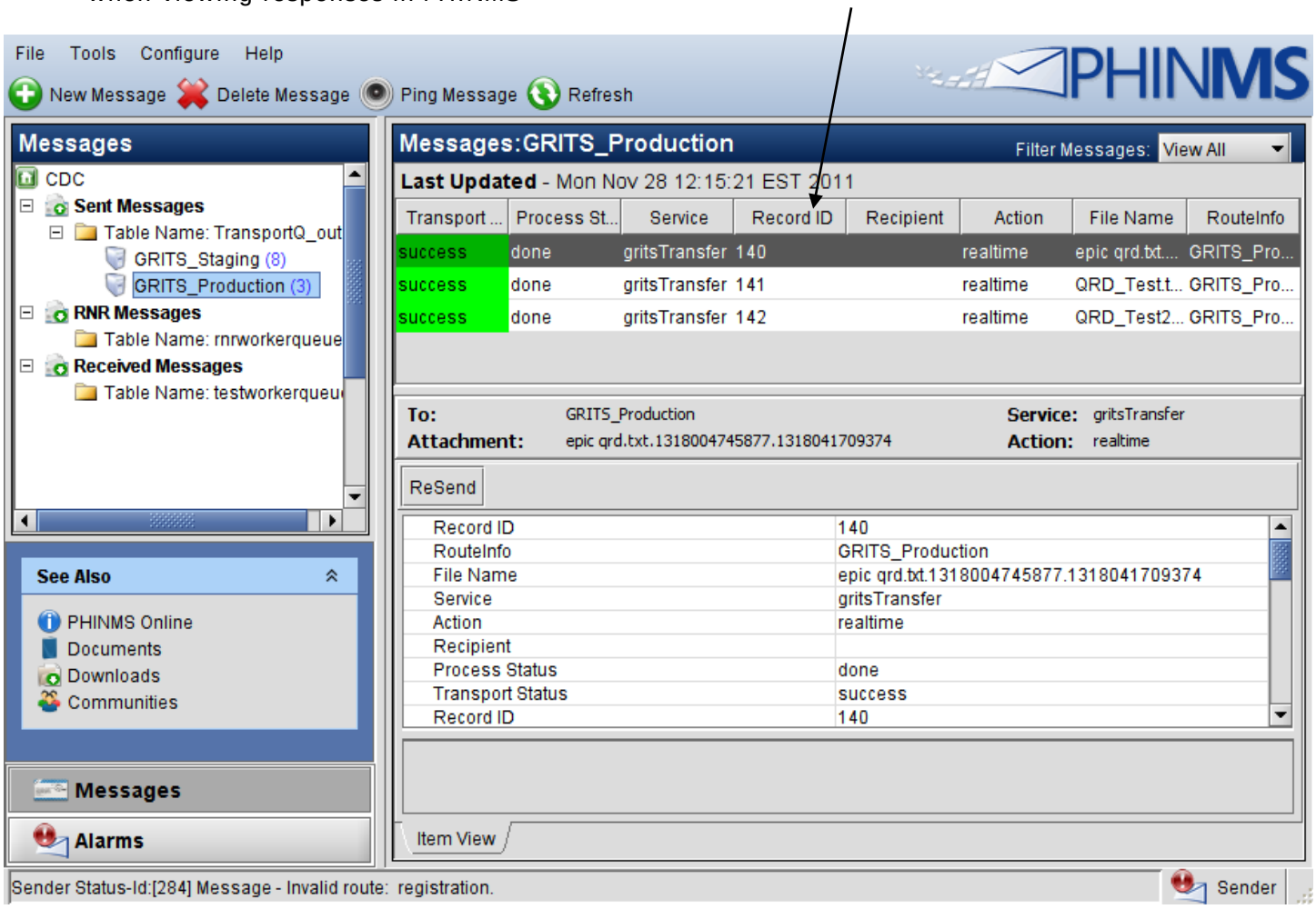
vii. PHINMS TESTING

Once PHINMS has been successfully installed for testing, please ensure the following are accomplished during the testing phase:

1) Confirm that your messages are successfully processed in GRITS. To do this, you'll need to look in your <installation directory>shared/senderincoming directory on your server for GRITS response messages (<installation directory> is the directory where you installed the PHINMS program. By default this is C:/Program Files/PHINMS, however, you may have changed it during installation.

2) Error messages such as "a valid eligibility code is required for new immunization" or any other errors must be resolved during the testing phase. Messages noted in the response file as "Informational" should be resolved, however, resolving these errors is not as critical as non-informational error messages.

3) The format of the response files is <partyid>.default.default.<PHINMS record id>…. Where the PHINMS record can be tied directly to the message sent from the host system and can be seen when viewing responses in PHINMS



4) Be sure your system is equipped to notify the user who initiated the request of the error. For example if GRITS rejects the record because the immunization is administered before the patients date of birth, that transaction was never successfully loaded to GRITS and the end user will not know this unless he or she is notified by your system.

viii. PHINMS PRODUCTION CONFIGURATION

The PHINMS installation used for Staging (test) can also be used for production. However, for production, a few changes are required. ***Please note that you will not be permitted to configure/utilize your production route until testing is complete.***

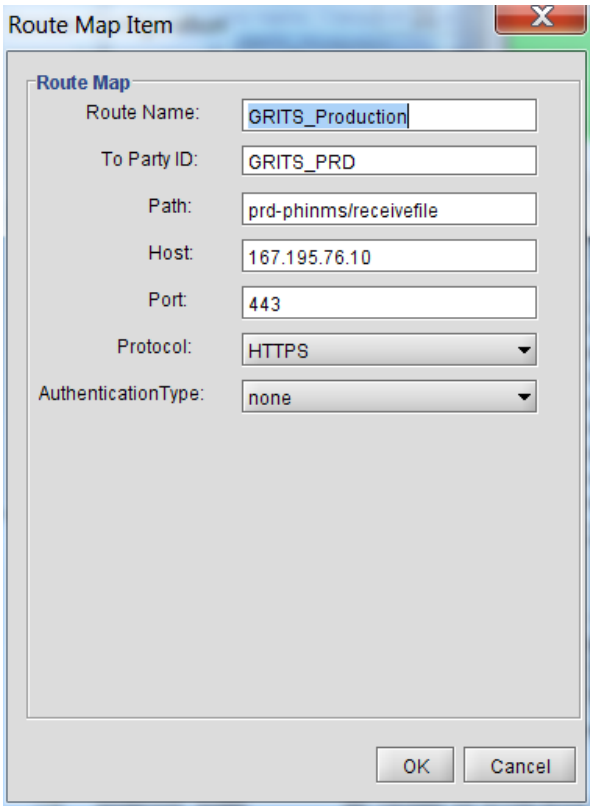
Configure the Production Routemap:

Select configure →sender → routemap → add

Your routemap must configured EXACTLY as follows:

i) For Route Name enter: **GRITS_Production**
ii) For the "To Party ID" Enter: **GRITS_PRD**
iii) For the "Path" enter: **prd-phinms/receivefile**
iv) For the "Host" use the IP address: **167.195.76.10**

Once complete, the staging routemap should look like this:



Configure the Production Folder Poll:

Select configure →sender →folder polling → add

Required fields are indicated by the red asterisk *

i) The Name should be something that indicates you are connection to your respective environment. In this case we used "GRITS_PRD_POLL" to indicate Polling properties for the GRITS Staging environment

ii) Select the production route from the dropdown list

iii) The Service MUST be "gritsTransfer" *<case sensitive>*

iv) The Action MUST be "realtime" *<case sensitive>*

v) Outgoing folder is the location where the messages you intend to send to GRITS from your production system are stored

vi) The Processed Folder is the location where outgoing messages are moved to once they have been processed by PHINMS

vii) Acknowledge Folder is the location where acknowledgement records will be sent. For detailed error messages that come from GRITS, please access messages located in : <installation directory>shared/senderincoming