Canada Savings Bonds

Obligations d'épargne du Canada

FTPS Server
User Guide

Version 2.2

April 19, 2013

FTPS Server User Guide – Revision History

Use the following table to track the revision history for this document. Please ensure that the date, phase and contact information are provided so that questions regarding the content may be directed to the appropriate individual.

Revision Date	Phase	Reason	Ver.
May 9, 2007		Draft	1.0
May 24, 2007		Second Draft	1.1
September 11, 2007		Final	1.2
October 22, 2007		Added contact information, modified file name and document title. Added a Preface to summarize details to be provided at implementation.	1.3
January 31, 2008		Replacement of temporary snapshots	1.4
February 19, 2008		Added a note on configuration of organization's firewall.	1.5
March 18, 2008		Expanded the accepted data file naming standard (section 5.1.3) to support an alphanumeric sequence number of any length.	1.6
March 24, 2008		Changed the recommended file transfer mode.	1.7
April 15, 2008		Added a File Confirmation step for Processing Agents in Section 3.3 . Removed references to FundServ.	1.8
May 21, 2008		Added support for explicit mode.	1.9
July 6, 2009		Updated 5.1.3 Data File Naming to separate the explanation for the sequence numbering for payroll contribution files and purchase files as requirements are not the same. And to update links to documentation on CSB website.	2
May 11, 2010		Changed RPAC-PFT email address from EDS to csb.gc.ca, changed campaign period from 6 to 2 months and changed EDS to HP.	2.1
April 19, 2013		Changed Processing Agents contact information on preface page.	2.2

TABLE OF CONTENTS

TRADEMARKS

GLOSSARY OF ACRONYMS

API	Application Programmer's Interface
ASCII	American Standard Code for Information Interchange
BOC	Bank of Canada
CA	Computer Associates
CD	Compact Disk
CPB	Canada Premium Bond
CPU	Central Processing Unit
CSB	Canada Savings Bond
DB	Database
HP	Hewlett Packard
EMC	EMC Corporation
FTP	File Transfer Protocol
FTPS	FTP over SSL
GB	Gigabyte
GHz	Gigahertz
GUI	Graphical User Interface
I/O	Input / Output
MFC	Microsoft Foundation Class library
N/A	Not Applicable
OS	Operating System
RDMS	Retail Debt Management System
RDO	Retail Debt Operation
SDK	Software Development Kit
SPOC	Secure Posting Of Contribution and other files
SSL	Secure Socket Layer

PREFACE

This document describes the implementation of the Bank of Canada's secure FTPS solution.

CONTACT INFORMATION

Organizations transmitting **payroll contribution files**

- Please call the Employer Help Line at 1 888 467-5999 Monday to Friday, 8 am to 6 pm, Eastern Time.

Processing Agents transmitting Canada Savings Bond **purchase files**

- Please call 1 888 646-2626 Monday to Friday, 8 am to 8 pm, Eastern Time.

1. INTRODUCTION

1.1. Purpose

This document is a guide that can be used by organizations using the FTPS server to upload data files (such as purchase and payroll contribution files). It is intended for a fairly technical audience that is already familiar with FTP and is already aware of the files to be uploaded (see references in section 1.5 'Referenced Documents').

1.2. Scope of this Document

This document provides the technical information required to use the FTPS system. It is not intended to document the business processes related to the FTPS system. It is intended to complement, not replace, the file specifications referenced in section 1.5 'Referenced Documents'.

1.3. Background

The FTPS system supports two types of data file uploads:

- Processing Agent organizations transmit files containing details of bond purchases made by the general public. These purchase files are received during the Canada Savings Bond sales campaign.

- Employer organizations transmit files containing employee contribution details for non-certificated purchases taking place through payroll deduction. Payroll contribution files are transmitted on a regular basis, such as weekly or every two weeks, to coincide with employer payroll cycles. In both cases, the data is classified Protected B[1] and as such special measures are needed to ensure the data is transmitted and handled securely.

1.4. Organization of This Document

Section 1 Introduction. This section describes the purpose, scope and organization of this document. This section also identifies all documents referenced within this document.

[1] "Protected B" (particularly sensitive) is a Government of Canada designation that applies to information that, if compromised, could reasonably be expected to cause serious injury outside the national interest and often include information, which if released, would reasonably compromise individual privacy e.g., loss of reputation or competitive advantage.

Section 2 Overview. This section provides an overview of the network architecture and the security features of the FTPS system.

Section 3 Using a FTPS Client GUI. This section provides instructions to users wishing to use the FTPS system through a GUI-based client software.

Section 4 Using a FTPS Client API. This section provides instructions to users wishing to use the FTPS system through a custom-built client software.

Section 5 Upload File Specifications. This section provides or references the specifications of the files that may be uploaded through the FTPS system.

1.5. Referenced Documents

The following documents are referenced within, or have been used in the preparation of this deliverable.

Technical Specifications for Proprietary Payroll System Users

http://www.csb.gc.ca/wp-content/uploads/2009/02/s3conv-technicalspecifications.pdf

Retail Debt Management System (RDMS) Purchase File Specifications

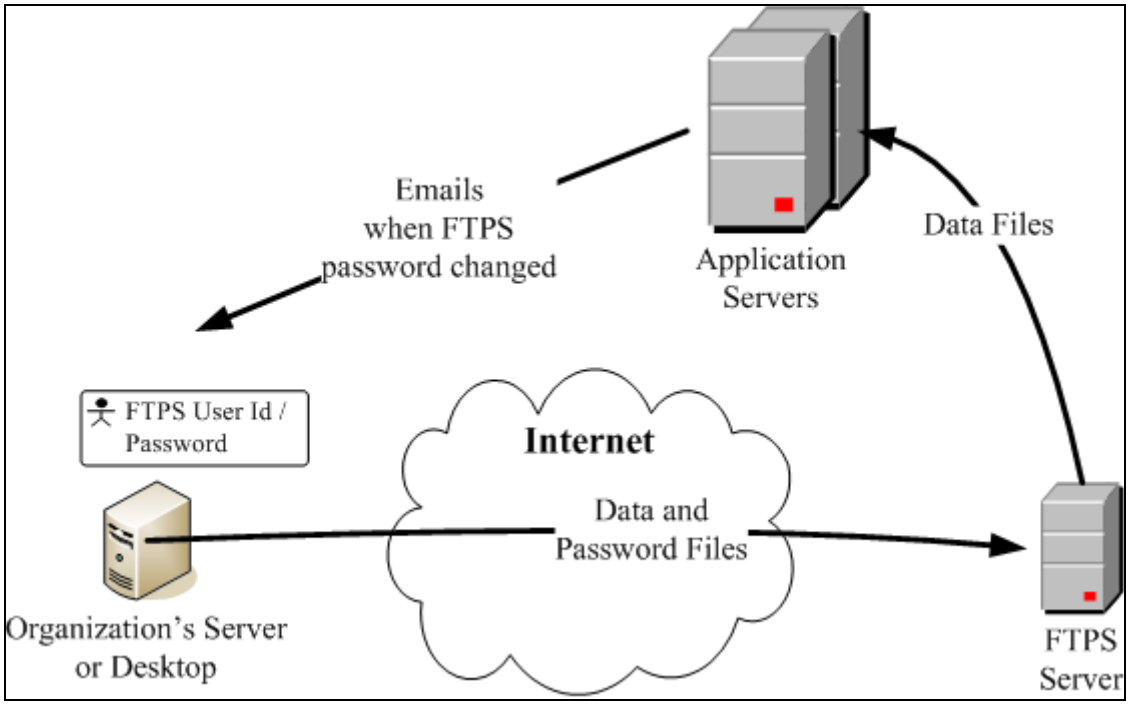
http://csb.gc.ca/fis/selling-and-processing-s42/?language=en

2. OVERVIEW

This section provides an overview of the network architecture and the security features of the FTPS system.

2.1. Network Architecture

The following diagram shows the network architecture of the FTPS system. An organization uses the public Internet to connect to the FTPS server and upload data and password files.



Organizations that use a firewall that restricts outgoing connections must configure their firewalls to allow outgoing connections to IP address 204.104.133.46 (csb-oec.bpmca.com) on ports 990 and 23001 to 23100 if using "implicit" mode, or ports 21021 and 23001 to 23100 if using "explicit" mode.

2.2. Steps for Using FTPS

The following are the basic steps that are followed to use the FTPS system. More details on these steps are provided in the following sections of this document:

1. Establish a connection to the Internet.

2. Connect to the FTPS server. In this step, the user has two options (for either option, data encryption must be enabled):

    a. Implicit mode with passive connection type. This option must use port 990.

    b. Explicit mode with passive connection type. This option must use port 21021.

3. Login to the FTPS server using the user id assigned to the organization and by entering the organization's secret password.

4. If prompted, accept the FTPS server's certificate.

5. Set the transfer mode to ASCII or binary. On the first attempt to use the FTPS server with a test file, users should use ASCII; if this mode is not successful (which may happen if the file contains special characters) then binary should be used.

6. Repeat the following step as many times as required:

    a. Upload a data file or a password file (the password file is used to change the organization's secret password).

7. Close the FTPS connection.

2.3. Security

The data files uploaded through the FTPS system are not encrypted before being transmitted but are protected by the following measures:

- Files are transmitted using FTP with SSL, which ensures that the files are encrypted while in transit on the Internet.

- Each organization is allowed and encouraged to change its password frequently. A minimum frequency of 3 months is recommended.

- Once an organization changes its password, that password is known only by the organization. The Bank of Canada help desk cannot see the password.

- By enforcing stringent criteria for its passwords, the FTPS server ensures that only strong passwords are used (see section 5.2.1 'Password File Contents' for a description of the criteria).

- Users of the FTPS server have upload capability but no download, delete, list, or rename capability.

- The FTPS system verifies the data in the data files against the user id of the sending organization. A data file is accepted only if it contains data that the organization is permitted to submit (see sections 5.1.1 'Payroll Contribution Data File Contents' and 5.1.2 'Purchase Data File Contents' for more details).

3. USING A FTPS CLIENT GUI

This section provides instructions to users wishing to use the FTPS system through a GUI-based client software.

3.1. Recommended FTPS Client GUI

The FTPS system has been thoroughly tested using Glub Tech Secure FTP v2.5.12 on Windows XP. This software is available on Windows, Mac OS X, and any Unix platform where a Java runtime environment (version 1.4+) is present. It can be purchased online from Glub Tech (http://www.glub.com/store/) for US$25[2].

3.1.1. Installation of Glub Tech Secure FTP

The following steps may be followed to install Glub Tech Secure FTP on a Windows desktop (similar instructions would be used for other platforms):

- If you don't have the Java Runtime Environment (JRE) version 1.4 or greater, you must first download and install it. This can be done, for example, from the following web address using a web browser (such as Microsoft Internet Explorer): http://java.sun.com/javase/downloads/index_jdk5.jsp (download Java Runtime Environment 5.0 Update 12). Note that this is an example only – any version that is equal to 1.4 or greater can be used.

- If you had to download the Java JRE, install it by running the downloaded file (e.g., jre-1_5_0_12-windows-i586-p.exe).

- Purchase a license through the following web address using a web browser, and follow the vendor's instructions: http://www.glub.com/store/

- Download the Glub Tech Secure FTP software from the following web address using a web browser: http://www.glub.com/store/download.jsp?shortname=secureftp_2_5

- Install Glub Tech Secure FTP by running the downloaded file (e.g., secureftp2_5_13_setup.exe).

- The start-up program for Glub Tech Secure FTP should be at the following location: C:\Program Files\Secure FTP 2.5\ secureftp.bat.

[2] Note that prices are the prerogative of the vendor and are not guaranteed by the Bank of Canada.

3.2. Other FTPS Client GUIs

Organizations may also use other FTPS client GUI software since the FTPS solution is standards based and not reliant on a specific vendor. Software packages that are expected to work are as follows. Note that none of these packages has been tested with the Bank of Canada FTPS solution.

- edtFTPj/PRO (http://www.enterprisedt.com/products/edtftpjssl/overview.html). This is Java-based GUI usable on any platform that supports Java 1.5.x or above.

- WS_FTP Professional 2007 (http://www.ipswitch.com/products/ws_ftp/index.asp). This product can be used on Windows platforms, including Windows XP and Windows Vista.

- jMethods JFTP (http://www.jmethods.com/products/). This is Java-based GUI usable on any platform that supports Java 1.4.2 or above.

- FTP Voyager (http://www.ftpvoyager.com/). This product can be used on Windows platforms, including Windows XP and Windows Vista.

3.3. Steps to Upload a File

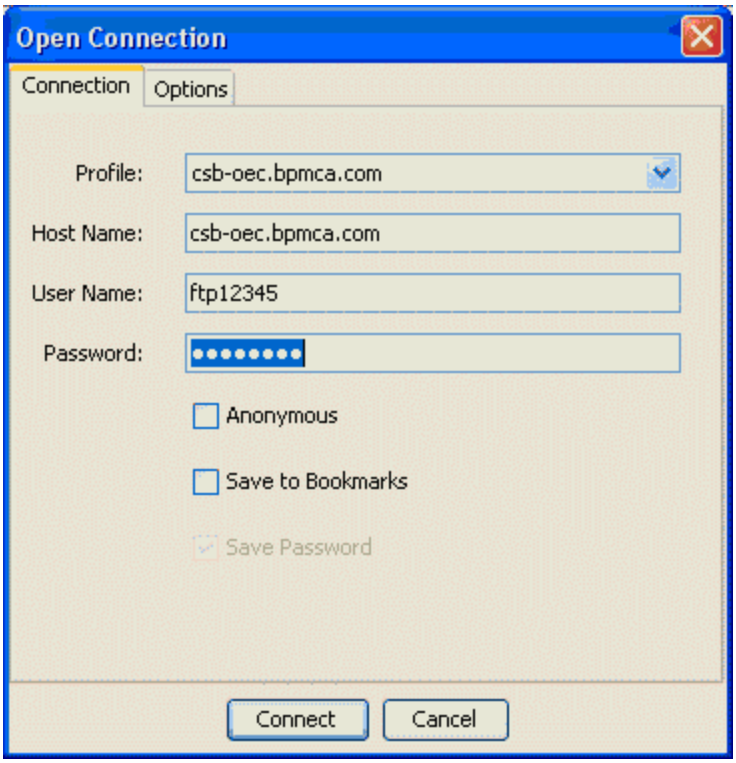
The following are the standard steps expected to be used to upload a file to the FTPS server. The snapshots provided assume the use of Glub Tech Secure FTP client GUI (see section 3.1 'Recommended FTPS Client GUI'). The steps should be similar if using different client software.

Notes:

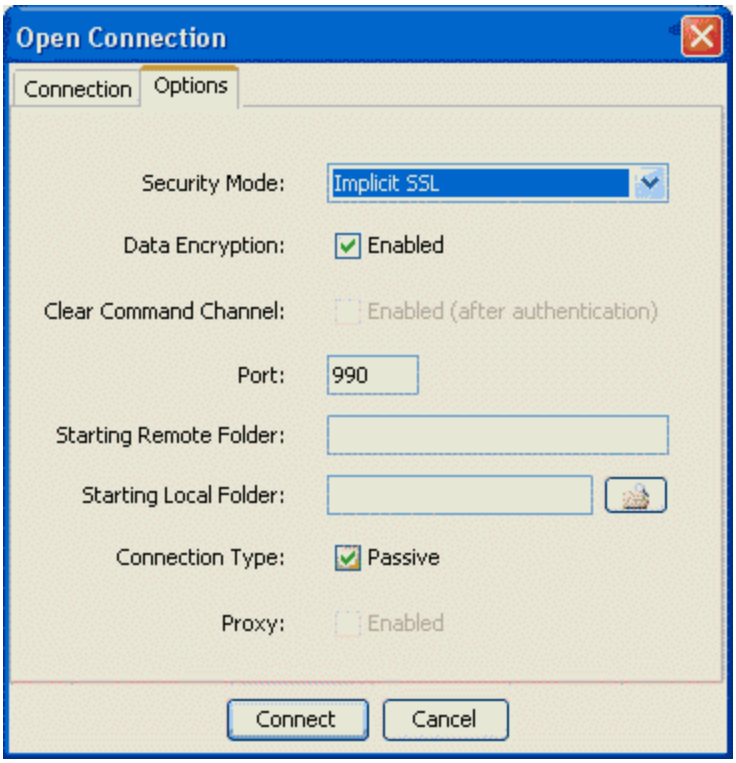
- If your organization uses a proxy server, use menu option File/Preferences to set the proxy server's parameters in Glub Tech Secure FTP.

- Organizations that use a firewall that restricts outgoing connections must configure their firewalls to allow outgoing connections to IP address 204.104.133.46 (csb-oec.bpmca.com) on ports 990 and 23001 to 23100 if using "implicit" mode, or ports 21021 and 23001 to 23100 if using "explicit" mode.

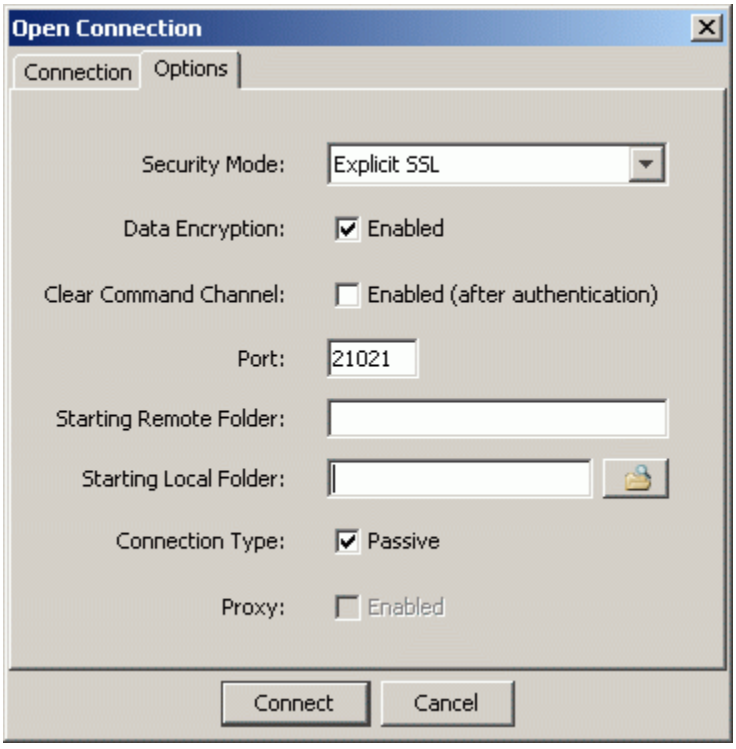
1. Start the client GUI. A screen like the following should come up (otherwise click on "Connect").



2. Enter the following, select “Save to Bookmarks” (this ensures that your entries are saved for future re-use) then click on tab “Options”:
- The host name csb-oec.bpmca.com or 204.104.133.46 .
- Your organization’s user id (which is “ftp” followed by the Organization Id).
- Your secret password.


Open Connection
Connection
Options
Security Mode:
Implicit SSL
Data Encryption:
Enabled
Clear Command Channel:
Enabled (after authentication)
Port:
990
Starting Remote Folder:
Starting Local Folder:
Connection Type:
Passive
Proxy:
Enabled
Connect
Cancel

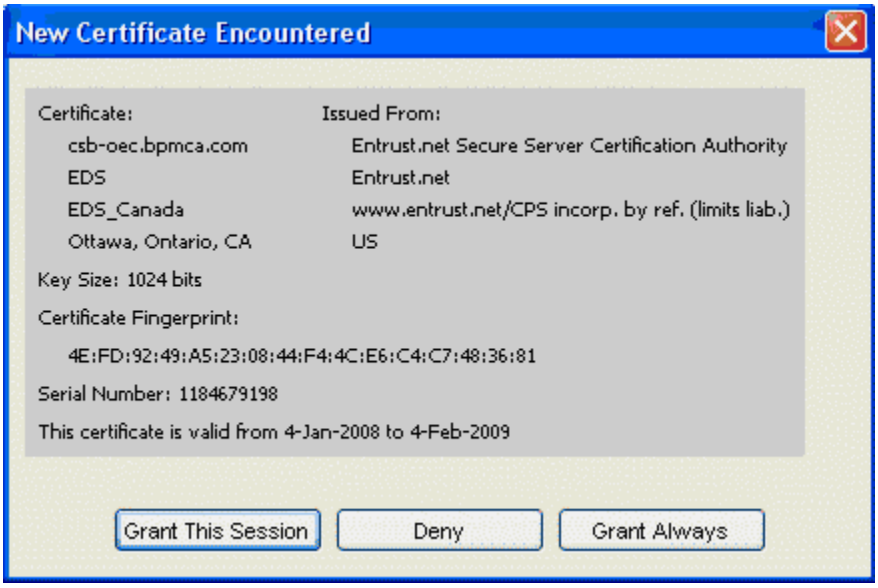
Or


Open Connection
Connection
Options
Security Mode:
Explicit SSL
Data Encryption:
Enabled
Clear Command Channel:
Enabled (after authentication)
Port:
21021
Starting Remote Folder:
Starting Local Folder:
Connection Type:
Passive
Proxy:
Enabled
Connect
Cancel

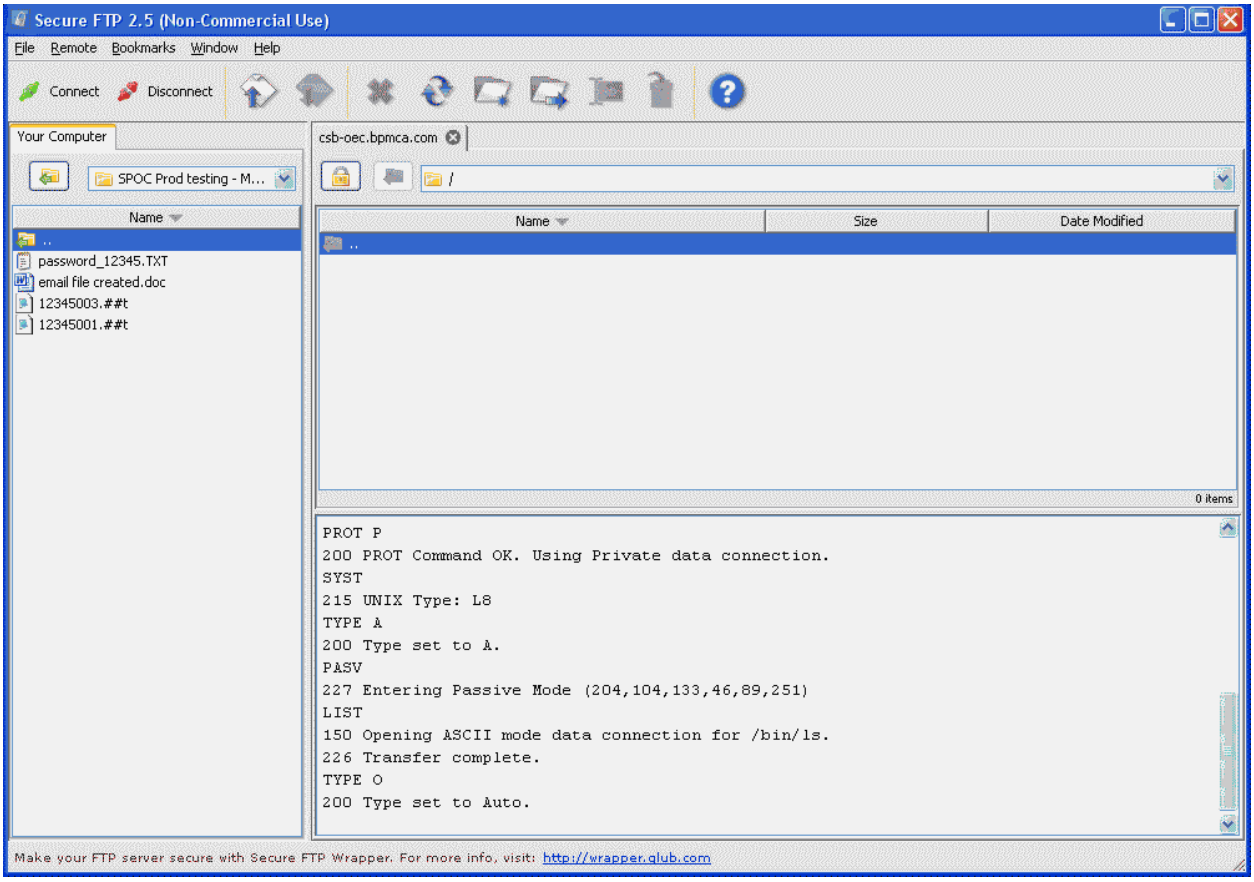
3. You have two options:

"Implicit" option: Select "Implicit SSL" (port should change to 990) and ensure that "Data Encryption" is checked (Enabled) and "Connection Type" is checked (Passive).

"Explicit" option: Select "Explicit SSL" (port should change to 21 but you must change it to 21021) and ensure that "Data Encryption" is checked (Enabled) and "Connection Type" is checked (Passive).

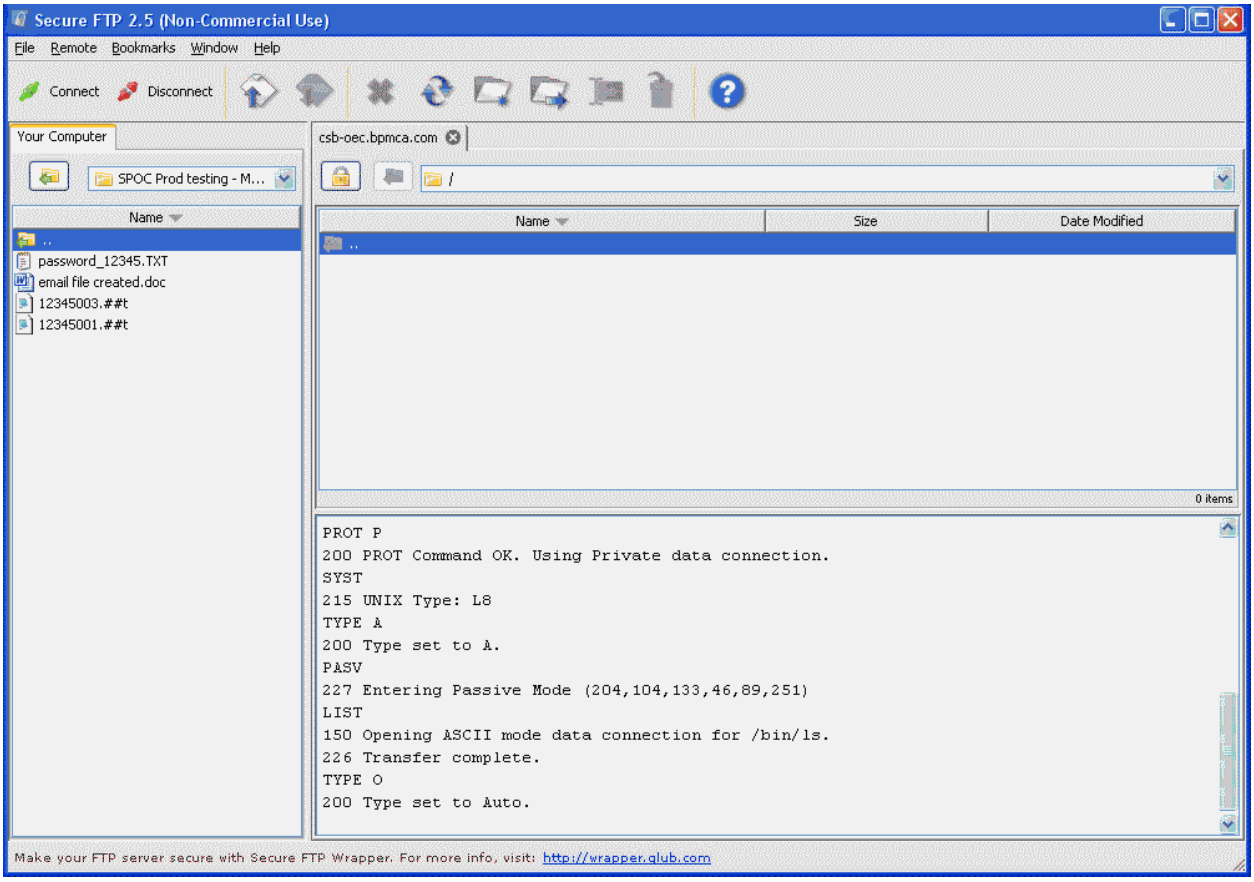


4. Accept the certificate by selecting either "Grant This Session" or "Grant Always".



5. The left side of the screen shows your local folder. Change to the folder that contains the files to be uploaded.

6. Set the transfer mode (use “Remote / Transfer Mode / Text” to use ASCII or use “Remote / Transfer Mode / Binary” to use binary). Drag and drop the file(s) to be uploaded from the left side of the screen to the right side. Note that the file still does not appear in the file list even though it was successfully copied – this is because the content of the folder is blocked for security reasons.



7. Disconnect from the FTPS server by clicking on “Disconnect”.

8. Note: File confirmation (for Processing Agents transmitting Canada Savings Bond purchase files only). Processing Agents must provide advanced notice prior to submitting a purchase file. This can be done by sending an e-mail with the file information to rpac-pft@csb.gc.ca. When the file is received and processed, a reply will be sent to the submitting organization. This confirmation should be received within 24 hours of processing. If not, please e-mail rpac-pft@csb.gc.ca for follow-up.

4. USING A FTPS CLIENT API

This section provides instructions to users wishing to use the FTPS system through custom-built client software.

4.1. Recommended FTPS Client API

The FTPS system has been thoroughly tested using Glub Tech Secure FTP Bean v2.5.6 with Java 5. This software is available on Windows, Mac OS X, Linux, and any Unix platform where a Java runtime environment (version 1.4+) is present. It can be purchased online from Glub Tech (http://www.glub.com/store/). The price of the license varies depending on the intended use – for an organization that intends to use the software internally (i.e., not for resale), the price is US$500 per developer[3].

4.1.1. Installation of Glub Tech Secure FTP Bean

The following steps may be followed to install Glub Tech Secure FTP Bean on a Windows desktop (similar instructions would be used for other platforms):

- If you don't have the Java Development Kit (JDK) version 1.4 or greater, you must first download and install it. This can be done, for example, from the following web address using a web browser (such as Microsoft Internet Explorer): http://java.sun.com/javase/downloads/index_jdk5.jsp (download JDK 5.0 Update 12).

- If you had to download the Java JDK, install it by running the downloaded file (e.g., jdk-1_5_0_12-windows-i586-p.exe).

- Purchase a license of Secure FTP Bean through the following web address using a web browser and follow the vendor's instructions: http://www.glub.com/store/

- Download the Glub Tech Secure FTP Bean software from the following web address using a web browser: http://www.glub.com/store/download.jsp?shortname=bean_2

- Install Glub Tech Secure FTP Bean by unzipping the downloaded file (e.g., gtftps2_5_6.zip) into a folder on your desktop.

- The root folder for Glub Tech Secure FTP Bean should be at the following location: C:\Program Files\SecureFtpBean.

[3] Note that prices are the prerogative of the vendor and are not guaranteed by the Bank of Canada.

4.2. Other FTPS Client APIs

Organizations may also use other FTPS client API software since the FTPS solution is standards based and not reliant on a specific vendor. Software packages that are expected to work are as follows. Note that none of these packages has been tested with the Bank of Canada FTPS solution.

- edtFTPj/PRO (http://www.enterprisedt.com/products/edtftpjssl/overview.html). This is a Java-based API usable on any platform that supports Java 1.5.x or above.

- WS_FTP Professional SDK (http://www.ipswitch.com/products/ws_ftp/devkit/index.asp). This language-independent API can be used on Windows platforms. It uses the Windows Component Object Model (COM) architecture.

- Secure FTP Factory (http://www.jscape.com/sftp/index.html). This is Java-based API usable on any platform that supports Java 1.2.2 or above.

- jMethods Secure FTP API for Java (http://www.jmethods.com/products/). This is a Java-based API usable on any platform that supports Java 1.4.2 or above.

- FTP Voyager SDK (http://sdk.ftpvoyager.com/). This API is a Windows MFC C++ library.

4.3. Sample Code

The following is an example of a Java class that uses Glub Tech Secure FTP Bean.

```
/*
 * Example of FTPS client with Glub Tech Secure FTP Bean
 * Written by HP
 * Last updated in April 2007
 *
 * Notes:
 * - Libraries required (included with Glub Tech Secure FTP Bean):
 *   Gtftps.jar, jakarta-regexp-1.4.jar.
 * - This example is for learning only.  It is NOT a fully-functional
 *   production application.
 */

package ftpsexamples;

import com.glub.secureftp.bean.SSLCertificate;
import com.glub.secureftp.bean.SSLFTP;
import com.glub.secureftp.bean.SSLSessionManager;

import java.io.File;

public class FtpsClientExample implements SSLSessionManager {
```

```
    private SSLCertificate currentCert = null;

    public static void main(String[] args) {

        // This call allows the java.security.SecureRandom object to be
        // generated prior to being used.  Class SecureRandom provides a
        // cryptographically strong pseudo-random number generator (PRNG).
        // This secure random number generator is used by Glub Tech Secure
        // FTP Bean.
        SSLFTP.preSeed();

        // Obtain FTPS parameters
        String hostNameOrIpAddress = null;
        int hostPortNumber = 0;
        String ftpsUserId = null;
        String ftpsPassword = null;

        if (args.length < 4) {
            hostNameOrIpAddress = "111.111.111.111"; // Not a real IP address
            hostPortNumber = 990;
            ftpsUserId = "ftp12345";
            ftpsPassword =
                    "ftp12345XXXX"; // Not a recommended password in production
        } else {
            hostNameOrIpAddress = args[0];
            hostPortNumber = new Integer(args[1]).intValue();
            ftpsUserId = args[2];
            ftpsPassword = args[3];
        }

        // Perform the upload
        FtpsClientExample ftps = new FtpsClientExample();
        ftps.uploadFile(hostNameOrIpAddress, hostPortNumber, ftpsUserId,
                        ftpsPassword, "D:\\tempo\\password.txt");

        // Prints if no exceptions
        System.out.println("Done.");
    }

    // Connect to FTPS server, upload file, then disconnect

    public void uploadFile(String host, int port, String user, String pass,
                           String fileName) {

        // Instantiate the SSLFTP object and provide FTPS parameters
        SSLFTP sslFTP =
            new SSLFTP(this, host, port, SSLFTP.IMPLICIT_CONNECTION,
                       System.out, System.out);
        try {
            // Connect and login to FTPS server
            sslFTP.connect();
            sslFTP.login(user, pass, null);

            // Ensure that data is encrypted
```

```
            // in the communication with the FTPS server
            sslFTP.setDataEncryptionOn(true);

            // Transfer a file to the FTPS server in ASCII mode
            sslFTP.ascii();
            sslFTP.store(new File(fileName), false);

            // Logout
            sslFTP.logout();

        } catch (Exception e) {
            System.err.println("An error occured: " + e.getMessage());
            e.printStackTrace();
            System.exit(-1);
        }
    }

    // Callback method required to implement interface SSLSessionManager

    public boolean continueWithCertificateHostMismatch(SSLCertificate cert,
                                                       String actualHost,
                                                       String certHost) {
        System.out.println("Certificate host mismatch.");
        return false;
    }

    // Callback method required to implement interface SSLSessionManager

    public boolean continueWithExpiredCertificate(SSLCertificate cert) {
        System.out.println("Certificate expired.");
        return false;
    }

    // Callback method required to implement interface SSLSessionManager

    public boolean continueWithInvalidCertificate(SSLCertificate cert) {
        System.out.println("Certificate invalid.");
        return false;
    }

    // Callback method required to implement interface SSLSessionManager

    public boolean continueWithoutServerCertificate() {
        System.out.println("Certificate not sent from server.");
        return false;
    }

    // Callback method required to implement interface SSLSessionManager

    public short newCertificateEncountered(SSLCertificate cert) {
        System.out.println("New certificate found:");
        System.out.println("Common Name........: " + cert.getCN());
        System.out.println("Start Date.........: " + cert.getStartDate());
        System.out.println("End Date...........: " + cert.getEndDate());
```

```
        System.out.println("Fingerprint........: " + cert.getFingerprint());
        System.out.println("Serial Number......: " + cert.getSerialNumber());
        System.out.println("Organization.......: " + cert.getOrg());
        System.out.println("Organizational Unit: " + cert.getOU());
        System.out.println("Locality...........: " + cert.getLocality());
        System.out.println("State/Province.....: " + cert.getState());
        System.out.println("Country............: " + cert.getCountry());
        System.out.println("Email..............: " + cert.getEmail());
        System.out.println("Issuer's Common Name........: " +
                           cert.getIssuerCN());
        System.out.println("Issuer's Organization.......: " +
                           cert.getIssuerOrg());
        System.out.println("Issuer's Organizational Unit: " +
                           cert.getIssuerOU());
        System.out.println("Issuer's Locality...........: " +
                           cert.getIssuerLocality());
        System.out.println("Issuer's State/Province.....: " +
                           cert.getIssuerState());
        System.out.println("Issuer's Country............: " +
                           cert.getIssuerCountry());
        System.out.println("Issuer's Email..............: " +
                           cert.getIssuerEmail());
        return SSLSessionManager.ALLOW_CERTIFICATE;
    }

    // Callback method required to implement interface SSLSessionManager

    public short replaceCertificate(SSLCertificate oldCert,
                                    SSLCertificate newCert) {
        System.out.println("Replace certificate.");
        return SSLSessionManager.ALLOW_CERTIFICATE;
    }

    // Callback method required to implement interface SSLSessionManager

    public void randomSeedIsGenerating() {
        System.out.print("The random seed is generating... ");
    }

    // Callback method required to implement interface SSLSessionManager

    public void randomSeedGenerated() {
        System.out.println("The random seed is generated... ");
    }

    // Callback method required to implement interface SSLSessionManager

    public void setCurrentCertificate(SSLCertificate currentCert) {
        this.currentCert = currentCert;
    }
}
```

5. SPECIFICATIONS FOR UPLOAD FILES

This section provides or references the specifications of the files that may be uploaded through the FTPS system.

5.1. Data Files

5.1.1. Payroll Contribution Data File Contents

For specifications of the contents of the payroll contribution data files, please see document http://www.csb.gc.ca/wp-content/uploads/2009/02/s3conv-technicalspecifications.pdf

. The format of the data contents has not changed with the migration to FTPS.

The FTPS server performs the following validations on the data file before accepting it:

- The user id that uploaded the file must be defined in the FTPS server as a user of type "employer"; otherwise, the file is not processed.

- The transmitter's Organization Id that appears in the transmission header record (record type 10) and in the transmission trailer record (record type 90) must correspond to the FTPS user id that is uploading the file. For example, if the user id is ftp12345 then the transmitter's Organization Id must be 12345 in both the transmission header and the transmission trailer; otherwise, the file is not processed.

- The transmitter's Organization Id that appears in the transmission header record (record type 10) and in the transmission trailer record (record type 90) must correspond to the Organization Id that appears in the file name (see section 5.1.3 'Data File Naming'); otherwise, the file is not processed.

- The transmitter's Organization Id must be permitted to submit records on behalf of each Organization Id that appears within a batch header record (record type 20), a batch detail record (record types 30, 40, or 50), and a batch trailer record (record type 80); otherwise, the file is not processed.

5.1.2. CSB Purchase Data File Contents

For specifications of the contents of the purchase data files, please see the "Retail Debt Management System (RDMS) Purchase File Specifications Logical Record Standards and the RDMS Purchase File Specifications Data Element Dictionary" located on the CSB Website at the following address: http://csb.gc.ca/fis/selling-and-processing-s42/?language=en. The format of the data contents has not changed with the migration to FTPS.

The FTPS server performs the following validation on the data file before accepting it:

- The user id that uploaded the file must be defined in the FTPS server as a user of type "purchase agent"; otherwise, the file is not processed.

- The Organization Id that appears in the header record (record type A) and in the trailer record (record type Z) must be the same as the Organization Id that appears in the file name (see section 5.1.3 'Data File Naming'); otherwise, the file is not processed.

5.1.3. Data File Naming

A data file name (for payroll contribution data files and purchase data files) must be as follows (the data file naming convention has not changed with the migration to FTPS):

`xxxxxnnn.##T` *(not case sensitive) for testing*

`xxxxxnnn.##P` *(not case sensitive) for production*

Where:

xxxxx	The Organization Id – this is Organization Id that appears in the data file header.
nnn	A sequence number of your choice. **Payroll contribution data files**: The sequence number must use alphanumeric characters only (A to Z, a to z, and 0 to 9) and can be of any length greater or equal to 1. This number can be used to meet the naming requirements of your organization, such as different payrolls, different paydays, etc. Note that if several payroll contribution files are sent on the same day, it is required that this number be unique for each file within that day. **Purchase data files:** The sequence number must use alphanumeric characters only (A to Z, a to z, and 0 to 9) and must be 3 characters. Sequence Numbers should be unique within a campaign period. It is advisable to increment with each transmission.
"##T" / "##P"	Use as is (not case sensitive). A file having an extension of "##P" will be processed normally. A file having an extension of "##T" will be processed as a test file.

Examples of valid **payroll contribution** data file names:

`123451.##T`

`1234599999.##P`

```
12345001.##t

12345999.##p
```

Examples of valid **purchase** data file names:

```
12345001.##T

12345999.##P

12345678.##T

00001999.##P

12345001.##t

12345999.##p
```

Examples of invalid file names:

`12345.##p`	(Sequence number is missing)
`12345999.P`	("##" is missing from the extension)
`12345001.###T`	(Too many "#" in the extension)
`12345  1.##T`	(Spaces not allowed)
`12345999.#P`	(Not enough "#" in the extension)
`12345001.TXT`	("TXT" is not an acceptable extension here)
`12345001##T`	(Period is missing)
`99912345.##P`	(Organization Id must be before sequence number)

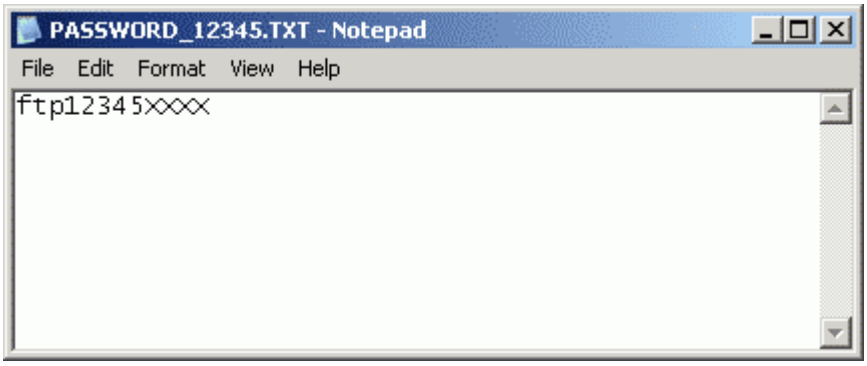
Notes:

- A data file must strictly follow this naming convention to be considered valid and to be processed.
- The original file stored in your environment is not required to have the same name as the file uploaded to the FTPS server; however, using the same name or a related name (e.g., "2009-JAN-05-12345001.##T" for upload file "12345001.##T") may allow an easier correlation of your files to the files uploaded to the FTPS server.
- We strongly recommend that you keep a copy of the file in your environment until you receive confirmation of transmission receipt and batch processing from Bank of Canada.

5.2. Password Files

5.2.1. Password File Contents

As stated earlier, we recommend that you update your password on a regular basis. The FTPS system allows you to modify your password by uploading a password file. The password file is expected to contain a single word (case sensitive). The word included in the file will be the new password. Example of password file (note that this is not a good example of password to use):



The FTPS server performs the following validations on the password before accepting it:

- The password must be ASCII character encoded text.
- The password must be at least 12 characters long but no more than 40 characters long.
- The password must contain at least one lower-case alphabetic character, one upper-case alphabetic character, and two numeric characters.
- The password must contain only a combination of the following characters: ‘a’ to ‘z’, ‘A’ to ‘Z’, and ‘0’ to ‘9’.
- The first eight characters of the password must contain at least one numeric character and two alphabetic characters.
- The password cannot be a circular shift of the user id (note that such a password would be invalid anyway because it would be only nine characters long and therefore too short).
- The new password must differ from the previous password and cannot be a reverse or circular shift of the previous password. For this comparison, uppercase letters and lowercase letters are considered to be equal.
- The new password must have at least three characters that are different from the old password. For this comparison, uppercase letters and lowercase letters are considered to be equal.

Examples of valid passwords:

```
ABCxyz123456

1933to1995ElizabethVictoriaMontgomery
```

Examples of invalid passwords:

`ABCxyz123`	(less than 12 characters long)
`1933to1995ElizabethVictoriaMontgomeryWasBewitched`	(too long)
`ElizabethVictoriaMontgomery`	(no numeric characters)
`1933to1995elizabethmontgomery`	(no upper-case characters)
`1933TO1995EMONTGOMERY`	(no lower-case characters)
`ElizabethMontgomery1933to1995`	(no numeric characters in first 8 characters)
`33-95ElizabethMontgomery`	(special character '-' is not accepted)
`33 95 Elizabeth Montgomery`	(space is not accepted)

5.2.2. Password File Naming

A password file name must be "password_xxxxx.txt" (not case sensitive) where xxxxx equals the Organization Id of the FTPS User id that is uploading the file. If a password file uses any other name, it will not be processed as a password file.

Examples of valid file names:

```
Password_12345.txt

PASSWORD_12345.TXT

Password_12345.txt

Password_12345.TXT

PassWord_12345.TXT
```

`PaSsWoRd.TxT`Examples of invalid file names:

`PASSWORD_12345`	(extension is missing)
`PASSWORD_12345.DOC`	(extension must be "TXT")
`Pass word_12345.txt`	(space is not accepted)

`Password_12345-txt`	(dash is not accepted in place of period)
`OpenSesame_12345.txt`	("OpenSesame" must be changed to "password")
Password.txt	(org id is missing)

Notes:

- A password file must strictly follow this naming convention to be considered valid and to be processed.

- The password change takes effect within a few minutes after a password file is uploaded. You will receive an email after the password change is attempted by the FTPS server – the email will state whether the change was successful or not.