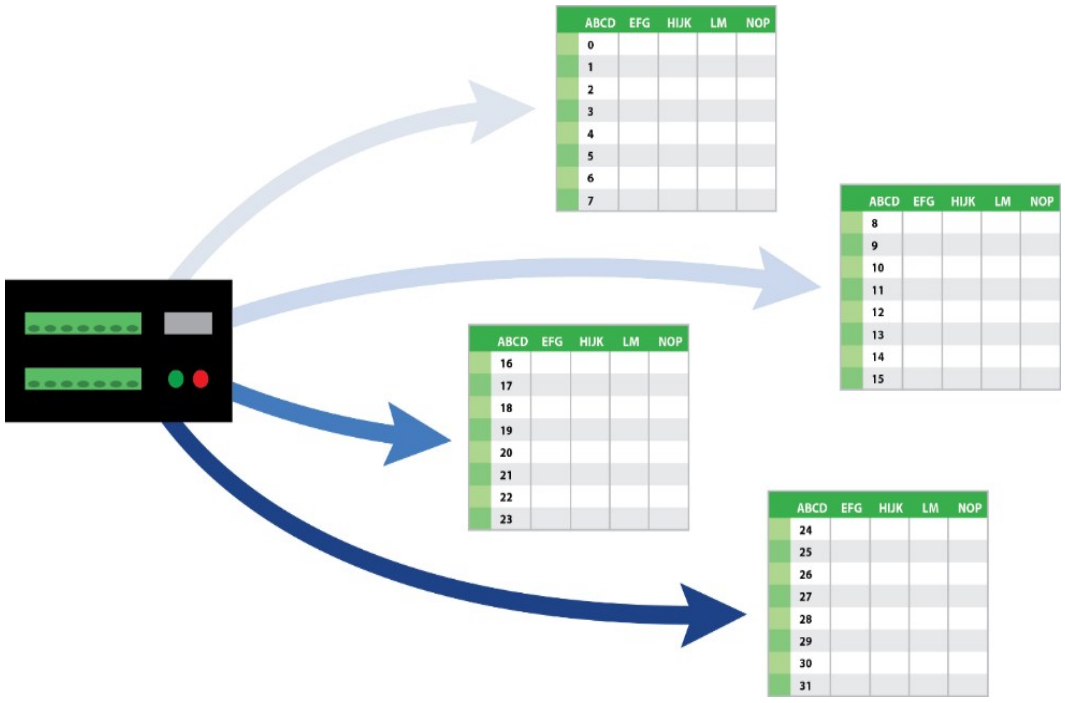
APPLICATION NOTE

Data Logger

FTP Troubleshooting

Outgoing communications



Revision: 07/2023

RELIABLE MONITORING SINCE 1974

Campbell SCIENTIFIC®

Table of Contents

1. Process and general protocol requirements **1**
1.1 FTP 1
1.2 SFTP 1
1.2.1 Authentication 2
1.2.2 SFTP key file requirements 2
1.2.3 FTPS 2
2. Verifying FTPClient() options **3**
3. Comms watch (sniff) FTP communications **6**
4. Analyzing FTP sniff-file error messages **12**
5. Verifying FTP server functionality with WinSCP **22**
6. Verifying IP interfaces are online and correct **26**
7. Checking DNS issues **27**

When troubleshooting FTP communications there are many variables. As a result, it can be difficult to know if a failure is due to incorrect configuration of data logger settings, or if there is a programming issue, or an incompatibility between the server and the data logger. The following sections are intended to help streamline the troubleshooting process.

1. Process and general protocol requirements

Work with your local IT personnel to ensure your data logger and network settings meet all specified requirements, including server settings, correct protocols and protocol versions, port numbers, and server cipher suites.

1.1 FTP

Basic FTP authentication usually requires a *username* and *password*.

There are two FTP connection modes, active and passive. In Active mode, the server actively initiates the connection and sends commands over port 21. Data is transferred over port 20.

In Passive mode, both the command connection and the data connection are established by the client on port 21. Passive mode is more common than Active mode.

See Verifying FTPClient() options (p. 3) for more information on the setting that determines whether the data logger operates in active or passive mode.

1.2 SFTP

The transaction starts with the server sending its banner. The data logger then tells the server what host key types, authentication methods, and other information it can accept. Regardless of authentication method, the data logger must then accept the host key of the server. If the server does not have a host key, it will generate one and send it to the data logger. If the host key type is supported, the data logger will accept the host key without question. After that the two will authenticate using key authentication. If the server does not offer key authentication or the data logger has missing, or incomplete keys then the data logger will attempt password authentication.

Ports: SFTP uses port 22 (SSH) for everything.

1.2.1 Authentication

Public key and password authentication are supported by SFTP.

Public key authentication: Requires a private and public key on the data logger. The public key is the same one that is on your server. Before client authentication takes place, the client (data logger) will be prompted by the SFTP server to validate the server public key when establishing a connection. The stack we are using in the data logger does not derive the public key from the private key. So, for this process to work, the data logger needs the public key. Obtain private and public keys from your IT department.

Password authentication: The server authenticates the client using a username and password from your data logger.

1.2.2 SFTP key file requirements

> NOTE:
> Microsoft Azure is not supported due to host key incompatibility.

File type: PEM formatted key files

Maximum key file size: 4 KB public, 4 KB private

Key exchange methods: diffie-hellman-group1-sha1, diffie-hellman-group14-sha1, diffie-hellman-group-exchange-sha1, diffie-hellman-group-exchange-sha256

Host key types: ssh-rsa, ssh-dss

Supported ciphers: aes256-ctr, aes192-ctr, aes128-ctr, aes256-cbc, aes192-cbc, aes128-cbc, 3des-cbc, blowfish-cbc, cast128-cbc, arcfour, arcfour128, none

Encrypted keys or keys with a passphrase are not currently supported.

See the blog article at https://www.campbellsci.com/blog/generate-sftp-keys-easily for information on generating compatible SFTP private and public keys for your data logger and server.

1.2.3 FTPS

FTPS operates similarly to basic FTP when FTP runs over an encrypted TLS connection. Your server will be configured to use one of two methods of encryption:

Explicit encryption: Port 21 is used to establish encryption.

Implicit encryption: Port 990 is used. Specify the port :990 in the address field of `FTPClient()`. Otherwise, the data logger will attempt to perform the transaction using port 21.

Either Passive or Active mode is used for transfer. In Passive mode, a random port over 45000 will be opened for the actual data session. Ensure those ports/ranges can pass through your firewall. Passive or Active mode can be selected on your data logger using `FTPClient()` `PutGetOption`. See Verifying FTPClient() options (p. 3).

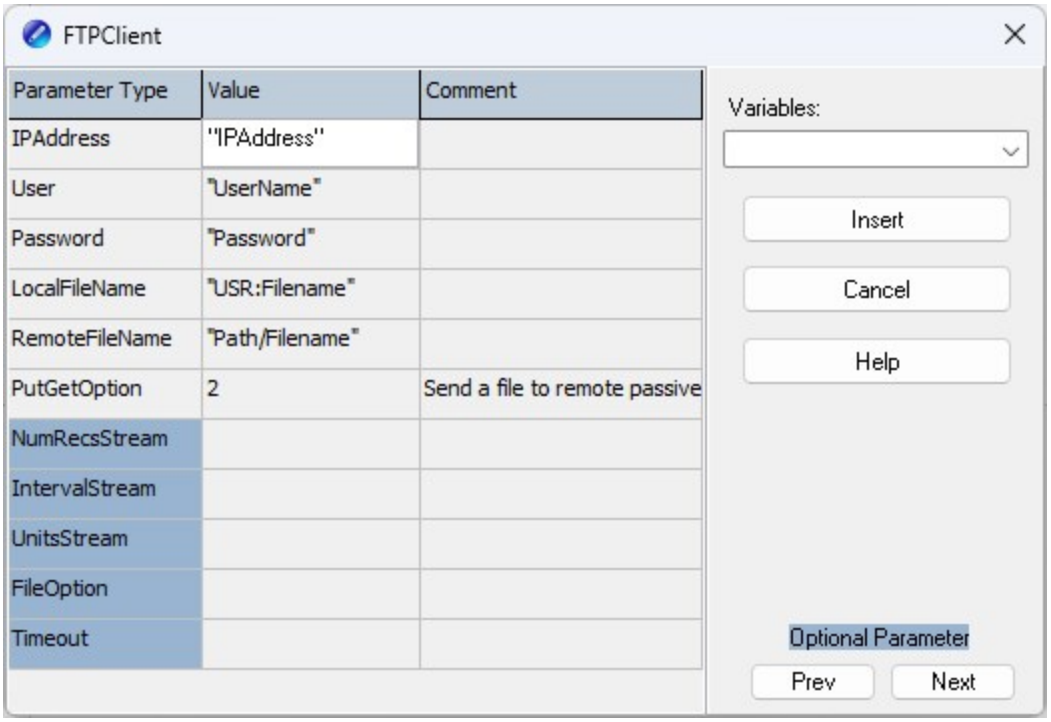
TLS versions supported: 1.2. Versions 1.0 and 1.1 have been deprecated after CR1000X OS 6.02 and CR6 OS 12.02.

The supported cipher suite list is very long. Most common suites are supported. Contact Campbell Scientific for more information.

> NOTE:
> If the server running FTPS (TLS) has a self-signed certificate or the certificate authority is unreachable, then the data logger will not accept the certificate. The data logger does not have a way of accepting a certificate it cannot verify.

2. Verifying `FTPClient()` options

Use the following guidelines to help verify `FTPClient()` has been configured correctly. The following image shows the instruction parameter entry dialog.



1. Ensure that you are using the syntax **FTPStatus**=**FTPClient()** in order to see status codes in your **Public** table. **FTPClient()** returns the following status codes: **-1** if successful, **0** if the connection fails, or **-2** if execution did not occur when the instruction was called. Receiving an error of -2 likely means that your data table does not have enough records to execute, based on the requirements of your **FTPClient()** instruction. This could occur if using the **NumRecsStream** parameter for streaming data and the number of specified records has not been reached yet. If you are receiving the failure message of **0**, see Comms watch (sniff) FTP communications (p. 6) and Analyzing FTP sniff-file error messages (p. 12) .

   See the highlighted portion of the following code snippet.

   ```
   FTPStatus=FTPClient ("IPAddress","User","Password","USR:Filename","Path/Filename",2)
   ```

2. Verify the **IPaddress** (or server address) is correct. If your server is using implicit encryption (FTPS), you may have to specify the port number by adding **:990** to the end of your server address. Otherwise, the data logger may attempt to do the communications over port 21 instead.

   See the highlighted portion of the following code snippet.

   ```
   FTPStatus=FTPClient ("IPAddress","User","Password","USR:Filename","Path/Filename",2)
   ```

3. Verify your username and password are correct.

   See the highlighted portion of the following code snippet.

   ```
   FTPStatus=FTPClient ("IPAddress","User","Password","USR:Filename","Path/Filename",2)
   ```

4. Verify your **LocalFileName** or table name is enclosed in quotes, unless it is a variable.

   See the highlighted portion of the following code snippet.

   ```
   FTPStatus=FTPClient ("IPAddress","User","Password","USR:Filename","Path/Filename",2)
   ```

5. If using a directory for the **RemoteFileName**, ensure that the folder has already been created on the server.

6. Enclose the contents of the **RemoteFileName** in quotes, unless you are using a variable.

   See the highlighted portion of the following code snippet.

   ```
   FTPStatus=FTPClient ("IPAddress","User","Password","USR:Filename","Path/Filename",2)
   ```

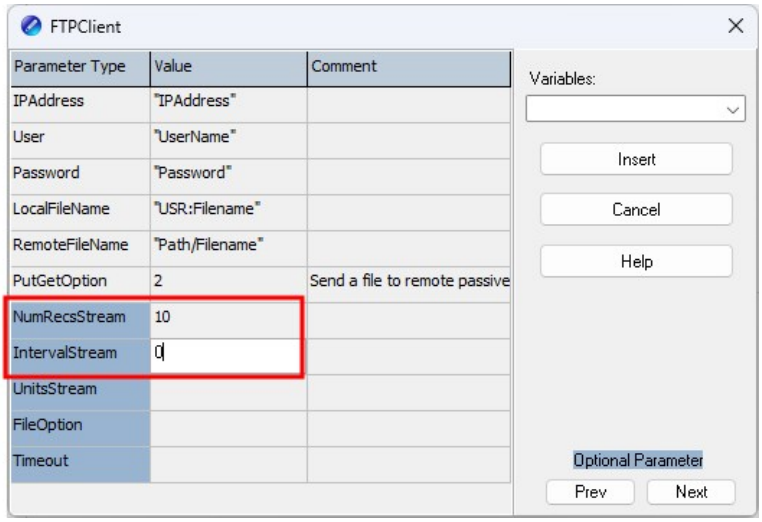
7. Check the ***CRBasic*** Help to ensure that your **PutGetOption** matches the protocol and FTP function you want to perform. On the CR1000X/CR6, options **0-9** are for FTP, options **10-19** are for FTPS, and options **20-28** are for SFTP.

   See the highlighted portion of the following code snippet.

   ```
   FTPStatus=FTPClient ("IPAddress","User","Password","USR:Filename","Path/Filename",2)
   ```

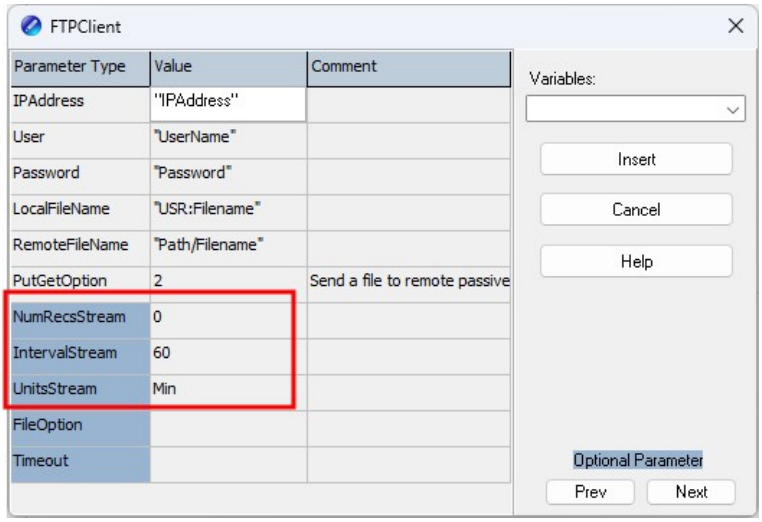
8. **NumRecsStream**, **IntervalStream**, and **UnitsStream** are optional parameters that are only used if you are streaming data from a data table.

9. If you are FTPing data from a table based on the number of unsent records in that table, ensure that you have specified the number of records in **NumRecsStream** and also set **IntervalStream** to **0**.



10. If you are FTPing data from a table based on a time interval, the **NumRecsStream** parameter now becomes a **TimeIntoInterval** parameter and must be filled in. In this case, a **NumRecsStream** parameter of **0** with an **IntervalStream** value of **60** with a **UnitsStream** value of Min indicates the FTP instruction will execute 0 minutes into a 60-

minute interval.



11. `FileOption` lets you select the format of the FTP file output on the server. The options include: `Binary`, `ASCII`, `XML`, and `JSON`.

> **NOTE:**
> Option **8** (TOA5, Header, TimeStamp, Record#) is the standard output format for .dat files that ***LoggerNet*** creates.

See the highlighted portion of the following code snippet.

```
FTPStatus=FTPClient
("IPAddress","User","Password","USR:Filename","Path/Filename",2,0,60,Min,8,8000)
```

12. The `TimeOut` parameter is in units of 0.01 seconds. The default is 7500 (75 seconds) does not need to be added to your instruction. Increasing it to greater than 7500 may help where latency or other network and server factors require the data logger to wait longer than normal for responses.

```
FTPStatus=FTPClient
("IPAddress","User","Password","USR:Filename","Path/Filename",2,0,60,Min,8,8000)
```

3. Comms watch (sniff) FTP communications

A terminal emulator can be used to watch (sniff) FTP communications.

ESC or a 40 second timeout will terminate on-going commands. The **W: Comms Watch** ("sniff") mode does not have a timeout when connected in terminal mode via PakBus. Otherwise, the timeout can be changed from the default of 40 seconds to any value ranging from 1 to 86400 seconds (86400 seconds = 1 day).

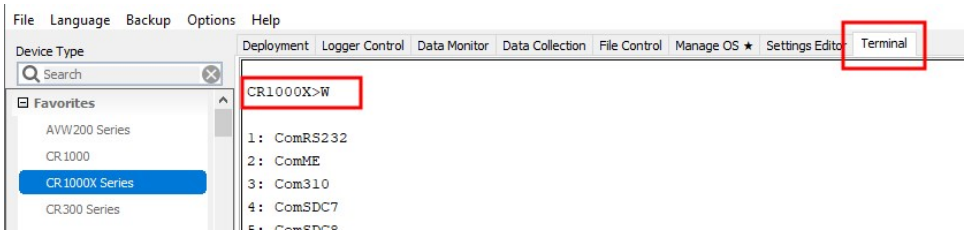
When using **W** in a terminal session, consider the following:

- Concurrent terminal sessions are not allowed and will result in dropped communications.
- Opening a new terminal session will close the current terminal session.
- The data logger will attempt to enter a terminal session when it receives non-PakBus characters on the **RS-232** port or **CS I/O** port, unless the port is first opened with the `SerialOpen()` instruction.

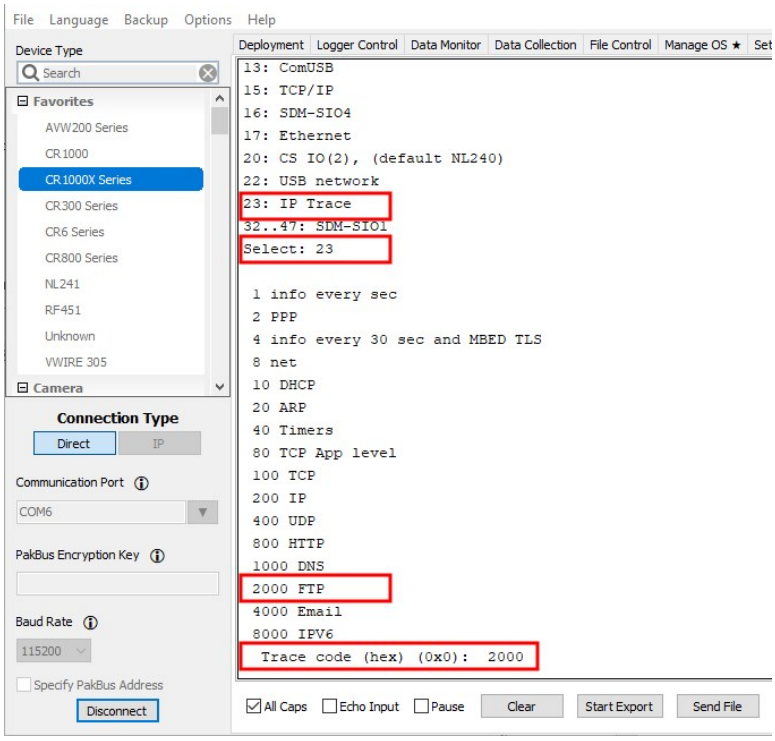
If the data logger attempts to enter a terminal session on the **RS-232** port or **CS I/O** port because of an incoming non-PakBus character, and that port was not opened using `SerialOpen()`, any currently running terminal function, including the comms watch, will immediately stop. So, in programs that frequently open and close a serial port, the probability is higher that a non-PakBus character will arrive at the closed serial port, thus closing an existing talk-through or comms watch session. If this occurs, use the **FilesManager** setting to send comms watch or sniffer to a file.

For more information on Comms Watch, see a video at: https://www.campbellsci.com/videos/sdi12-sensors-watch-or-sniffer-mode.

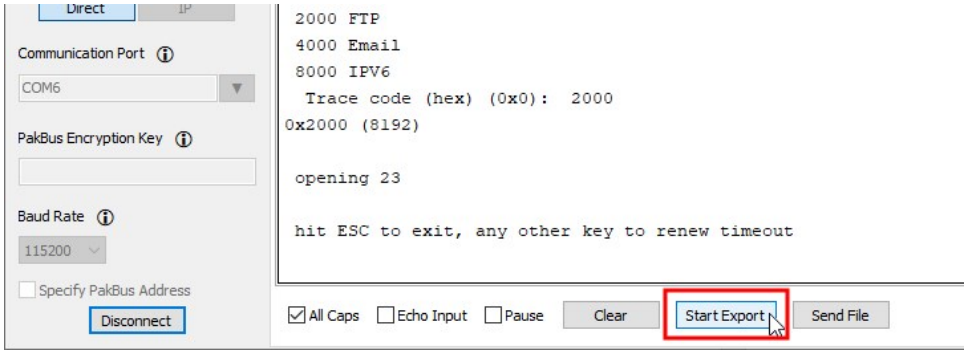
1. To enter terminal mode, connect a computer to the data logger. Open a terminal emulator program from Campbell Scientific data logger support software:
   - **Connect** window > **Datalogger** menu item> **Terminal Emulator...**
   - ***Device Configuration Utility*** **Terminal** tab
2. After entering a terminal emulator, press **Enter** a few times until the data logger prompt such as **CR1000X>** is returned. Type **W** followed by **Enter**.



3. Type the number that corresponds to IP Trace (**23** for the CR1000X) followed by **Enter**.
4. Type the number that corresponds to FTP (**2000**) followed by **Enter**.



5. You will now see a "hit ESC to exit, any other key to renew timeout" message. Press **Start Export** to create an export file of the results.

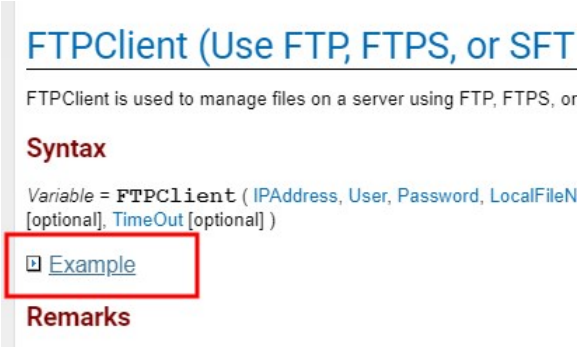


6. Type the location to save the export file. Click **Save**.
7. Wait for your FTP instruction to trigger in your data logger program or trigger it manually if you have a mechanism for doing it.

> **TIP:**
> For best results, let the FTP instruction complete or fail twice.

If the `FTPClient()` instruction only triggers once per day, then consider adding a mechanism for triggering it manually. ***CRBasic*** program example #2 includes a Flag for

manually triggering `FTPClient()`. See the ***CRBasic Editor*** help for detailed instruction information and program examples: https://help.campbellsci.com/crbasic/cr1000x/.



8. Click **End Export**. If the failure is due to a problem with FTP, then this export should contain the details of the failure.

9. The following example results show different causes of errors.

   Basic FTP missing directory error:

```
17:02:53.395  PASV
17:02:53.562  buffer_ssl_receive
17:02:53.562  ftp rx: 227 Entering Passive Mode (51,140,246,245,35,56)
17:02:53.563  NLST Station1.dat
17:02:53.563
17:02:53.724  buffer_ssl_receive
17:02:53.724  ftp rx: 550 Directory not found.
17:02:54.096  buffer_ssl_receive
17:02:54.096  ftp rx: 220-FileZilla Server 0.9.60 beta
17:02:54.096  ftp rx: 220-written by Tim Kosse (tim.kosse@filezilla-project.org)
17:02:54.097  ftp rx: 220 Please visit https://filezilla-project.org/
17:02:54.097  USER Weather
17:02:54.097
17:02:54.292  buffer_ssl_receive
17:02:54.292  ftp rx: 331 Password required for weather
17:02:54.292  PASS secret
17:02:54.293
17:02:54.463  buffer_ssl_receive
17:02:54.463  ftp rx: 230 Logged on
17:02:54.463  TYPE I
17:02:54.634  buffer_ssl_receive
17:02:54.634  ftp rx: 200 Type set to I
17:02:54.634  PASV
17:02:54.801  buffer_ssl_receive
17:02:54.801  ftp rx: 227 Entering Passive Mode (51,140,246,245,35,52)
17:02:54.802  NLST Station1.dat
17:02:54.802
17:02:54.969  buffer_ssl_receive
17:02:54.969  ftp rx: 550 Directory not found.
17:04:19.970  ftp client FAILED
17:04:55.634  buffer_ssl_receive
17:04:55.634  ftp rx: 421 Connection timed out.
17:04:55.634  QUIT
17:04:55.636  Closing handle 111.  Connection aborted.
17:04:55.636  ftp client FAILED
```

SFTP incompatible host encryption keys error:

```
10:02:02.961  Recved 1 bytes banner
10:02:02.961  Recved 1 bytes banner
10:02:02.961  Recved 1 bytes banner
10:02:02.961  Recved 1 bytes banner
10:02:02.961  Recved 1 bytes banner
10:02:02.961  Recved 1 bytes banner
10:02:02.962  Recved 1 bytes banner
10:02:02.962  Received Banner: SSH-2.0-AzureSSH_1.0.0
10:02:02.962  Sent KEX: diffie-hellman-group-exchange-sha256,diffie-hellman-grou
p-exchange-sha1,diffie-hellman-group14-sha1,diffie-hellman-group1-sha1
10:02:02.962  Sent HOSTKEY: ssh-rsa
10:02:02.962  Sent CRYPT_CS: aes128-ctr,aes192-ctr,aes256-ctr,aes256-cbc,rijndae
l-cbc@lysator.liu.se,aes192-cbc,aes128-cbc,blowfish-cbc,arcfour128,arcfour,3des-
cbc
10:02:02.962  Sent CRYPT_SC: aes128-ctr,aes192-ctr,aes256-ctr,aes256-cbc,rijndae
l-cbc@lysator.liu.se,aes192-cbc,aes128-cbc,blowfish-cbc,arcfour128,arcfour,3des-
cbc
10:02:02.962  Sent MAC_CS: hmac-sha2-256,hmac-sha2-512,hmac-sha1,hmac-sha1-96,hm
ac-md5,hmac-md5-96,hmac-ripemd160,hmac-ripemd160@openssh.com
10:02:02.962  Sent MAC_SC: hmac-sha2-256,hmac-sha2-512,hmac-sha1,hmac-sha1-96,hm
ac-md5,hmac-md5-96,hmac-ripemd160,hmac-ripemd160@openssh.com
10:02:02.962  Sent COMP_CS: none
10:02:02.962  Sent COMP_SC: none
10:02:02.962  Sent LANG_CS: Sent LANG_SC: => libssh2_transport_write plain (697
bytes)
10:02:02.963  Sent 712/712 bytes at 063D0170
10:02:02.963  => libssh2_transport_write send() (712 bytes)
10:02:02.963  Looking for packet of type: 20
10:02:03.386  Recved 372/16384 bytes to 060CC156+0
10:02:03.386  => libssh2_transport_read() raw (372 bytes)
10:02:03.386  Recved 252/16384 bytes to 060CC156+0
10:02:03.386  => libssh2_transport_read() raw (252 bytes)
10:02:03.386  => libssh2_transport_read() plain (610 bytes)
10:02:03.386  Packet type 20 received, length=610
10:02:03.386  Looking for packet of type: 20
10:02:03.387  -5 - Unable to exchange encryption keys
10:02:03.387  Failure establishing SSH session: -5
10:02:03.387  Freeing session resource
10:02:03.387  Extra packets left 0
10:02:03.387  ftp client FAILED_
```

FTPS username and password failure:

```
18:03:01.779  ftp rx: 220 (vsFTPd 3.0.5)
18:03:01.779  AUTH TLS
18:03:01.779  ftp rx: 530 Please login with USER and PASS.
18:03:01.780  QUIT
18:03:01.780  ftp rx: 221 Goodbye.
18:03:01.781  ftp client FAILED
```

FTPS data logger is trying to stream a DataTable to the server but the Streaming Options have not been correctly selected in **FTPClient()**:

```
11:15:11.433  ftp rx: 220 (vsFTPd 3.0.5)
11:15:11.433  USER user
11:15:11.433
11:15:11.434  ftp rx: 331 Please specify the password.
11:15:11.434  PASS FP5KQ3yH5
11:15:11.434
11:15:11.484  ftp rx: 230 Login successful.
11:15:11.484  TYPE I
11:15:11.486  ftp rx: 200 Switching to Binary mode.
11:15:11.486  PASV
11:15:11.487  ftp rx: 227 Entering Passive Mode (10,30,220,112,39,112).
11:15:11.487  cannot open file Test
11:15:11.488  QUIT
11:15:11.490  ftp rx: 221 Goodbye.
11:15:11.490  Write on handle 101 aborted, connection closing
11:15:11.493  ftp client FAILED
```

10. If the sniff file does not have usable information and you are confident that **FTPClient()** triggered and ran during the time you were capturing the results, you may have an IP configuration, DNS, or data logger IP interface routing issue. See Verifying IP interfaces are online and correct (p. 26) and Checking DNS issues (p. 27).

4. Analyzing FTP sniff-file error messages

Open the text file you saved and look for the **SUCCESS** or **FAILED** message. The error code or point of failure is shown above **FAILURE** message. For example, in Basic FTP missing directory

error: (p. 9) the following error code is shown on the lines preceding **FAILED**:

```
ftp rx: 550 Directory not found
```

In SFTP incompatible host encryption keys error: (p. 11) you see the following error code on the lines before the **FAILED** message:

```
-5 – Unable to exchange encryption keys
```

Other SFTP error codes could include messages similar to the following:

```
-18 - Authentication failed (username/password)
Authentication by password failed -18.
```

Use the following tables to help interpret error messages.

Table 4-1: FTP Error codes

Code	Description
331	User name okay, need password.
332	Need account for login.
350	Requested file action pending further information.
421	Service not available, closing control connection. This may be a reply to any command if the service knows it must shut down.
425	Cannot open data connection. Try changing from PASV to PORT mode.
426	Connection closed; transfer aborted.
450	Requested file action not taken. File unavailable (e.g., file busy).
451	Requested action aborted: local error in processing.
452	Requested action not taken. Insufficient storage space in system.
501	Syntax error in parameters or arguments.
502	Command not implemented. (The server does not support the FTP command you are using)
503	Bad sequence of commands.
504	Command not implemented for that parameter.
530	Not logged in. Your password is being rejected, contact the server administrator.
532	Need account for storing files.

Table 4-1: FTP Error codes	
Code	Description
550	Requested action not taken. File unavailable (e.g., file or directory not found, or no access). Contact the server administrator.
552	Requested file action aborted. Exceeded storage allocation (for current directory or data set). Contact the server administrator.
553	Requested action not taken. File name not allowed. Try changing the file name, or getting rid of spaces in the file name.

Table 4-2: SFTP Error codes	
Code	Description
-2	ERROR_BANNER_RECV
-3	ERROR_BANNER_SEND
-4	ERROR_INVALID_MAC
-5	ERROR_KEX_FAILURE – likely a host key incompatibility.
-6	ERROR_ALLOC
-7	ERROR_SOCKET_SEND
-8	ERROR_KEY_EXCHANGE_FAILURE
-9	ERROR_TIMEOUT
-10	ERROR_HOSTKEY_INIT
-11	ERROR_HOSTKEY_SIGN
-12	ERROR_DECRYPT
-13	ERROR_SOCKET_DISCONNECT
-14	ERROR_PROTO
-15	ERROR_PASSWORD_EXPIRED
-16	ERROR_FILE
-17	ERROR_METHOD_NONE
-18	ERROR_AUTHENTICATION_FAILED
	ERROR_PUBLICKEY_UNRECOGNIZED
	ERROR_AUTHENTICATION_FAILED

Table 4-2: SFTP Error codes	
Code	Description
	ERROR_AUTHENTICATION_FAILED (username/password) – data logger does not have valid keys for key authentication (which may be ok if you want to use username/password) and logger is not using correct username and password for username/password method.
-19	ERROR_PUBLICKEY_UNVERIFIED – this error will not be able to show up on our loggers.
-20	ERROR_CHANNEL_OUTOFORDER
-21	ERROR_CHANNEL_FAILURE
-22	ERROR_CHANNEL_REQUEST_DENIED
-23	ERROR_CHANNEL_UNKNOWN
-24	ERROR_CHANNEL_WINDOW_EXCEEDED
-25	ERROR_CHANNEL_PACKET_EXCEEDED
-26	ERROR_CHANNEL_CLOSED
-27	ERROR_CHANNEL_EOF_SENT
-28	ERROR_SCP_PROTOCOL
-29	ERROR_ZLIB
-30	ERROR_SOCKET_TIMEOUT
-31	ERROR_SFTP_PROTOCOL
-32	ERROR_REQUEST_DENIED
-33	ERROR_METHOD_NOT_SUPPORTED
-34	ERROR_INVAL
-35	ERROR_INVALID_POLL_TYPE
-36	ERROR_PUBLICKEY_PROTOCOL
-37	ERROR_EAGAIN
-38	ERROR_BUFFER_TOO_SMALL
-39	ERROR_BAD_USE
-40	ERROR_COMPRESS
-41	ERROR_OUT_OF_BOUNDARY

Table 4-2: SFTP Error codes	
Code	Description
-42	ERROR_AGENT_PROTOCOL
-43	ERROR_SOCKET_RECV
-44	ERROR_ENCRYPT
-45	ERROR_BAD_SOCKET
-46	ERROR_KNOWN_HOSTS

Table 4-3: FTPS (TLS) errors	
Code	Description
0x1080	PEM - No PEM header or footer found
0x1100	PEM - PEM string is not as expected
0x1180	PEM - Failed to allocate memory
0x1200	PEM - RSA IV is not in hex-format
0x1280	PEM - Unsupported key encryption algorithm
0x1300	PEM - Private key password cannot be empty
0x1380	PEM - Given private key password does not allow for correct decryption
0x1400	PEM - Unavailable feature, such as hashing/encryption combination
0x1480	PEM - Bad input parameters to function
0x1E00	PKCS12 - Given private key password does not allow for correct decryption
0x1E80	PKCS12 - PBE ASN.1 data not as expected
0x1F00	PKCS12 - Feature not available, such as unsupported encryption scheme
0x1F80	PKCS12 - Bad input parameters to function
0x2080	X509 - Unavailable feature, such as RSA hashing/encryption combination
0x2100	X509 - Requested OID is unknown
0x2180	X509 - The CRT/CRL/CSR format is invalid, different type expected
0x2200	X509 - The CRT/CRL/CSR version element is invalid
0x2280	X509 - The serial tag or value is invalid
0x2300	X509 - The algorithm tag or value is invalid

Table 4-3: FTPS (TLS) errors	
Code	Description
0x2380	X509 - The name tag or value is invalid
0x2400	X509 - The date tag or value is invalid
0x2480	X509 - The signature tag or value invalid
0x2500	X509 - The extension tag or value is invalid
0x2580	X509 - CRT/CRL/CSR has an unsupported version number
0x2600	X509 - Signature algorithm (oid) is unsupported
0x2680	X509 - Signature algorithms do not match. (see \\c ::mbedtls_x509_crt sig_oid)
0x2700	X509 - Certificate verification failed, for example CRL, CA or signature check failed
0x2780	X509 - Format not recognized as DER or PEM
0x2800	X509 - Input invalid
0x2880	X509 - Allocation of memory failed
0x2900	X509 - Read/write of file failed
0x2980	X509 - Destination buffer is too small
0x2e00	PKCS5 - Given private key password does not allow for correct decryption
0x2e80	PKCS5 - Requested encryption or digest alg not available
0x2f00	PKCS5 - Unexpected ASN.1 data
0x2f80	PKCS5 - Bad input parameters to function
0x3000	X509 - A fatal error occurred, for example, the chain is too long or the verify callback failed
0x3080	DHM - Bad input parameters
0x3100	DHM - Reading of the DHM parameters failed
0x3180	DHM - Making of the DHM parameters failed
0x3200	DHM - Reading of the public values failed
0x3280	DHM - Making of the public value failed
0x3300	DHM - Calculation of the DHM secret failed
0x3380	DHM - The ASN.1 data is not formatted correctly

Table 4-3: FTPS (TLS) errors	
Code	Description
0x3400	DHM - Allocation of memory failed
0x3480	DHM - Read or write of file failed
0x3500	DHM - DHM hardware accelerator failed
0x3580	DHM - Setting the modulus and generator failed
0x3880	PK - PK hardware accelerator failed
0x3900	PK - The buffer contains a valid signature followed by more data
0x3980	PK - Unavailable feature, for example, RSA disabled for RSA key
0x3A00	PK - Elliptic curve is unsupported (only NIST curves are supported)
0x3A80	PK - The algorithm tag or value is invalid
0x3B00	PK - The pubkey tag or value is invalid (only RSA and EC are supported)
0x3B80	PK - Given private key password does not allow for correct decryption
0x3C00	PK - Private key password cannot be empty
0x3C80	PK - Key algorithm is unsupported (only RSA and EC are supported)
0x3D00	PK - Invalid key tag or value
0x3D80	PK - Unsupported key version
0x3E00	PK - Read/write of file failed
0x3E80	PK - Bad input parameters to function
0x3F00	PK - Type mismatch, for example, attempt to encrypt with an ECDSA key
0x3F80	PK - Memory allocation failed
0x4080	RSA - Bad input parameters to function
0x4100	RSA - Input data contains invalid padding and is rejected
0x4180	RSA - Something failed during generation of a key
0x4200	RSA - Key failed to pass the validity check of the library
0x4280	RSA - The public key operation failed
0x4300	RSA - The private key operation failed
0x4380	RSA - The PKCS#1 verification failed
0x4400	RSA - The output buffer for decryption is not large enough

Table 4-3: FTPS (TLS) errors	
Code	Description
0x4480	RSA - The random generator failed to generate non-zeros
0x4500	RSA - The implementation does not offer the requested operation, for example, because of security violations or lack of functionality
0x4580	RSA - RSA hardware accelerator failed
0x4B00	ECP - Operation in progress, call again with the same parameters to continue
0x4B80	ECP - The ECP hardware accelerator failed
0x4C00	ECP - The buffer contains a valid signature followed by more data
0x4C80	ECP - Invalid private or public key
0x4D00	ECP - Generation of random value, such as ephemeral key, failed
0x4D80	ECP - Memory allocation failed
0x4E00	ECP - The signature is not valid
0x4E80	ECP - The requested feature is not available, for example, the requested curve is not supported
0x4F00	ECP - The buffer is too small to write to
0x4F80	ECP - Bad input parameters to function
0x5080	MD - The selected feature is not available
0x5100	MD - Bad input parameters to function
0x5180	MD - Failed to allocate memory
0x5200	MD - Opening or reading of file failed
0x5280	MD - MD hardware accelerator failed
0x6080	CIPHER - The selected feature is not available
0x6100	CIPHER - Bad input parameters
0x6180	CIPHER - Failed to allocate memory
0x6200	CIPHER - Input data contains invalid padding and is rejected
0x6280	CIPHER - Decryption of block requires a full block
0x6300	CIPHER - Authentication failed (for AEAD modes)
0x6380	CIPHER - The context is invalid. For example, because it was freed

Table 4-3: FTPS (TLS) errors	
Code	Description
0x6400	CIPHER - Cipher hardware accelerator failed
0x6480	SSL - Internal-only message signaling that a message arrived early
0x6500	SSL - The asynchronous operation is not completed yet
0x6580	SSL - Internal-only message signaling that further message-processing should be done
0x6600	SSL – Could not set the hash for verifying CertificateVerify
0x6680	SSL - The alert message received indicates a non-fatal error
0x6700	SSL - Record header looks valid but is not expected
0x6780	SSL - The client initiated a reconnect from the same port
0x6800	SSL - The operation timed out
0x6880	SSL - Connection requires a write call
0x6900	SSL - No data of requested type currently available on underlying transport
0x6980	SSL - None of the common ciphersuites is usable (for example, no suitable certificate, see debug messages)
0x6A00	SSL - A buffer is too small to receive or write a message
0x6A80	SSL - DTLS client must retry for hello verification
0x6B00	SSL - Unexpected message at ServerHello in renegotiation
0x6B80	SSL - A counter would wrap (for example, too many messages exchanged)
0x6C00	SSL - Internal error (for example, unexpected failure in lower-level module)
0x6C80	SSL - Unknown identity received (for example, PSK identity)
0x6D00	SSL - Public key type mismatch (for example, asked for RSA key exchange and presented EC key)
0x6D80	SSL - Session ticket has expired
0x6E00	SSL - Processing of the NewSessionTicket handshake message failed
0x6E80	SSL - Handshake protocol not within min/max boundaries
0x6F00	SSL - Processing of the compression / decompression failed
0x6F80	SSL - Hardware acceleration function skipped / left alone data

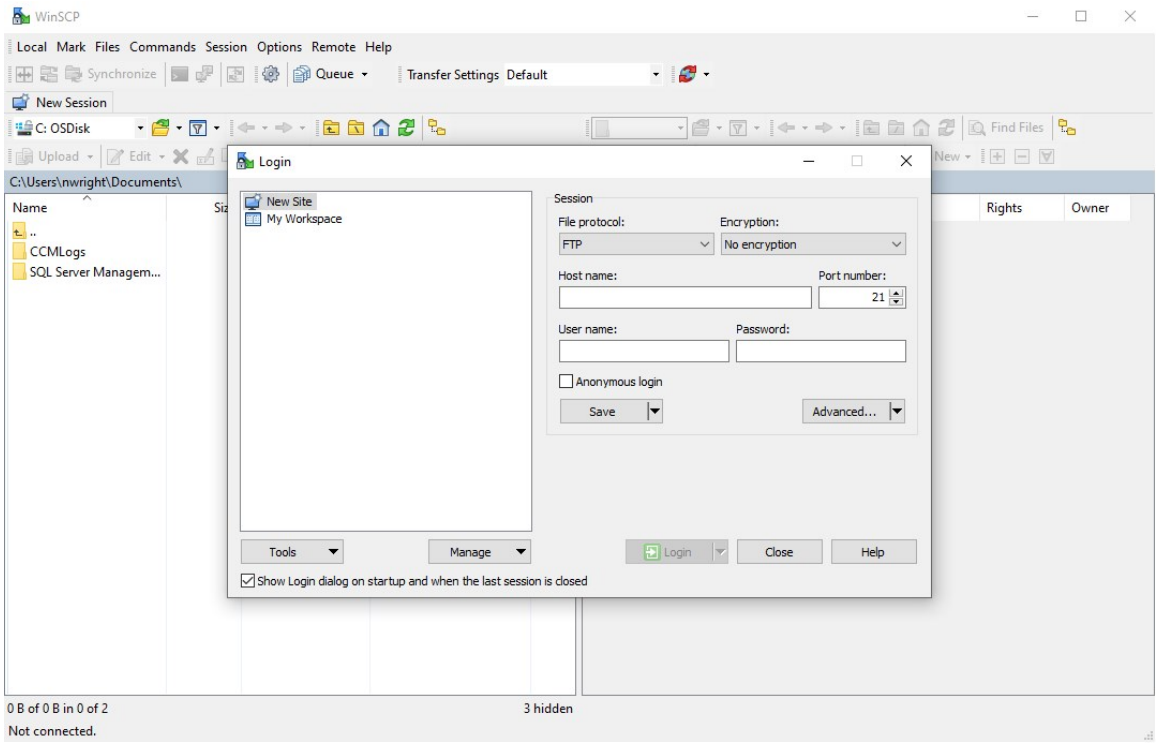
Table 4-3: FTPS (TLS) errors	
Code	Description
0x7000	SSL - A cryptographic operation is in progress. Try again later
0x7080	SSL - The requested feature is not available
0x7100	SSL - Bad input parameters to function
0x7180	SSL - Verification of the message MAC failed
0x7200	SSL - An invalid SSL record was received
0x7280	SSL - The connection indicated an EOF
0x7300	SSL - An unknown cipher was received
0x7380	SSL - The server has no ciphersuites in common with the client
0x7400	SSL - No RNG was provided to the SSL module
0x7480	SSL - No client certification received from the client, but required by the authentication mode
0x7500	SSL - Our own certificate(s) is/are too large to send in an SSL message
0x7580	SSL - The own certificate is not set, but needed by the server
0x7600	SSL - The own private key or pre-shared key is not set, but needed
0x7680	SSL - No CA Chain is set, but required to operate
0x7700	SSL - An unexpected message was received from our peer
0x7780	SSL - A fatal alert message was received from our peer
0x7800	SSL - Verification of our peer failed
0x7880	SSL - The peer notified us that the connection is going to be closed
0x7900	SSL - Processing of the ClientHello handshake message failed
0x7980	SSL - Processing of the ServerHello handshake message failed
0x7A00	SSL - Processing of the Certificate handshake message failed
0x7A80	SSL - Processing of the CertificateRequest handshake message failed
0x7B00	SSL - Processing of the ServerKeyExchange handshake message failed
0x7B80	SSL - Processing of the ServerHelloDone handshake message failed
0x7C00	SSL - Processing of the ClientKeyExchange handshake message failed

Table 4-3: FTPS (TLS) errors	
Code	Description
0x7C80	SSL - Processing of the ClientKeyExchange handshake message failed in DHM / ECDH Read Public
0x7D00	SSL - Processing of the ClientKeyExchange handshake message failed in DHM / ECDH Calculate Secret
0x7D80	SSL - Processing of the CertificateVerify handshake message failed
0x7E00	SSL - Processing of the ChangeCipherSpec handshake message failed
0x7E80	SSL - Processing of the Finished handshake message failed
0x7F00	SSL - Memory allocation failed
0x7F80	SSL - Hardware acceleration function returned with error

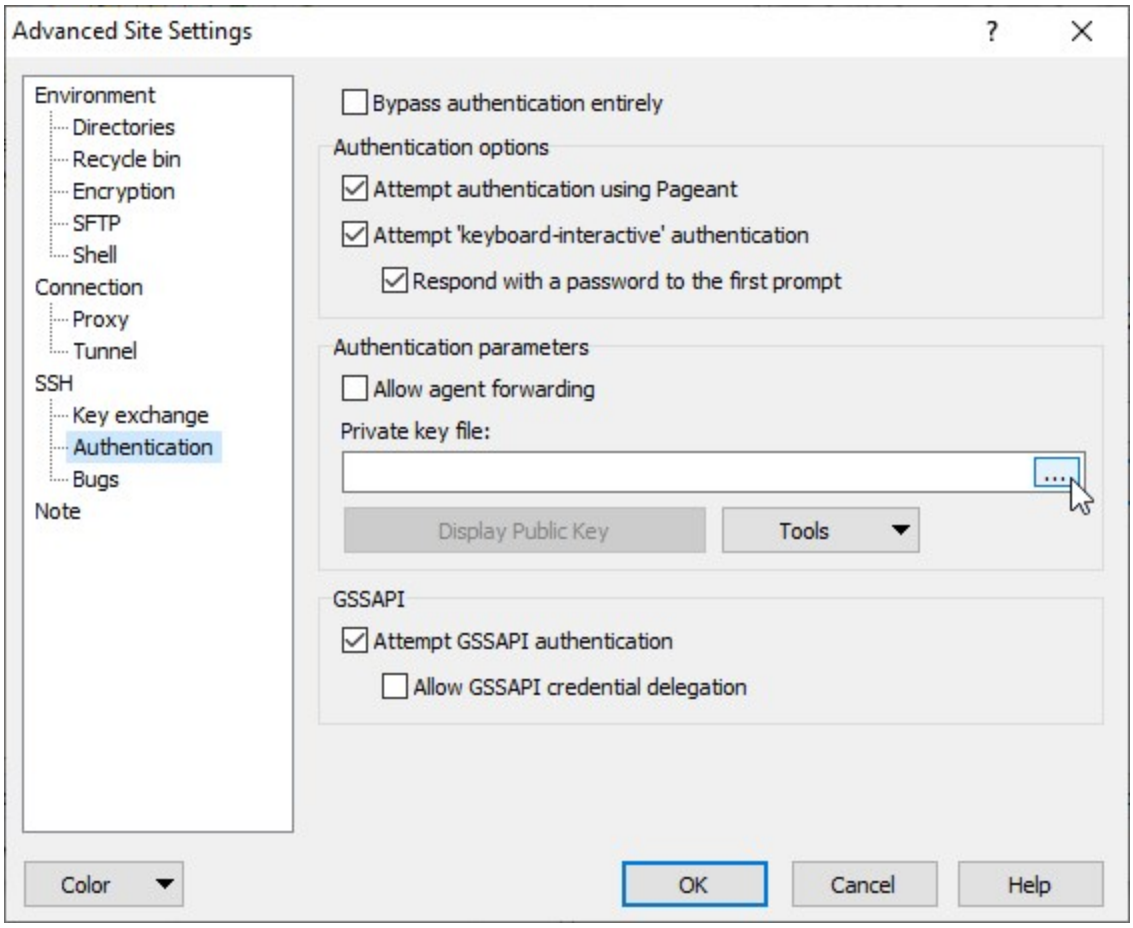
5. Verifying FTP server functionality with *WinSCP*

It is helpful to verify that your FTP server is setup correctly, especially when you are not the individual who maintains it. If your data logger is not communicating with the server, then testing from your computer with a 3rd party application may be useful. There are many FTP client solutions available. This example uses ***WinSCP***. ***WinSCP*** is a free download from: https://winscp.net/eng/download.php.

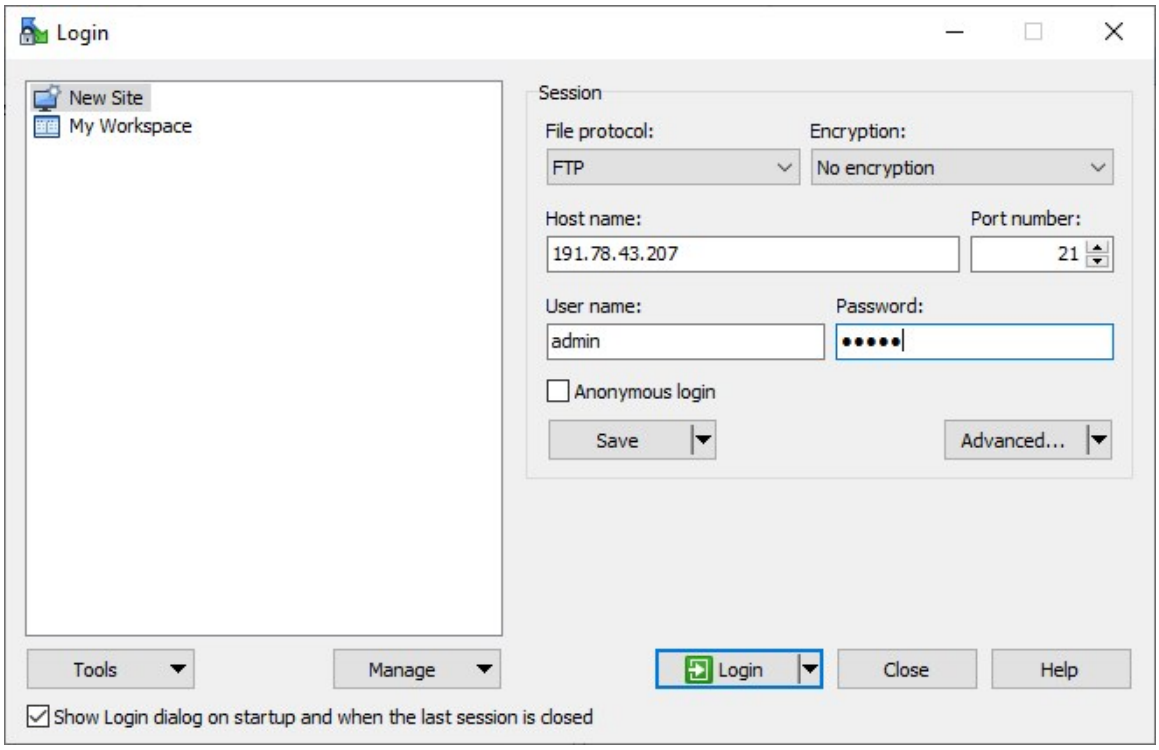
1. Install the application.
2. Start ***WinSCP***.
3. Enter your server **File protocol**, **Port number**, login credentials and other information. If your server is FTPS, select an option from the **Encryption** drop down that is either TLS/SSL Implicit Encryption or TLS/SSL Explicit Encryption.



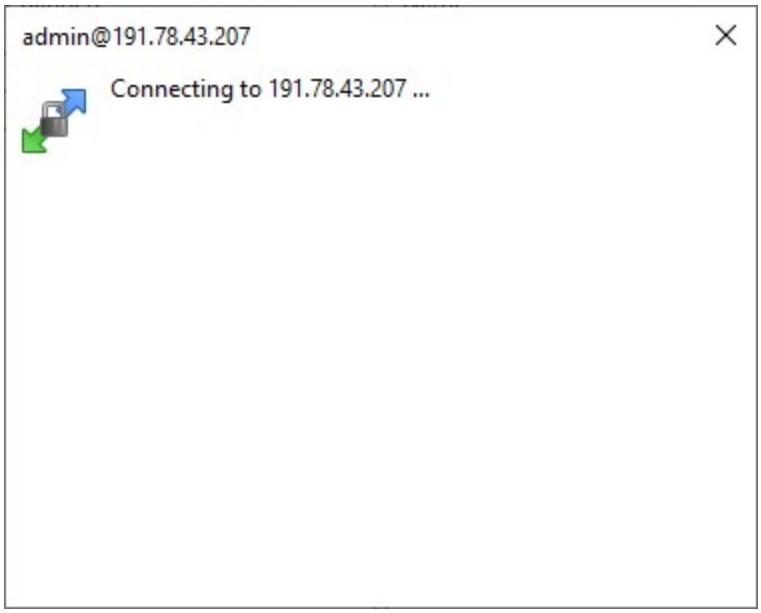
4. If your server is SFTP and requires that you use keys, they can be applied to the ***WinSCP Client*** by clicking **Advanced** > **Advanced option**. This will open the Advanced Site Settings screen. On the left menu click **Authentication** and apply your **Private Key file**.



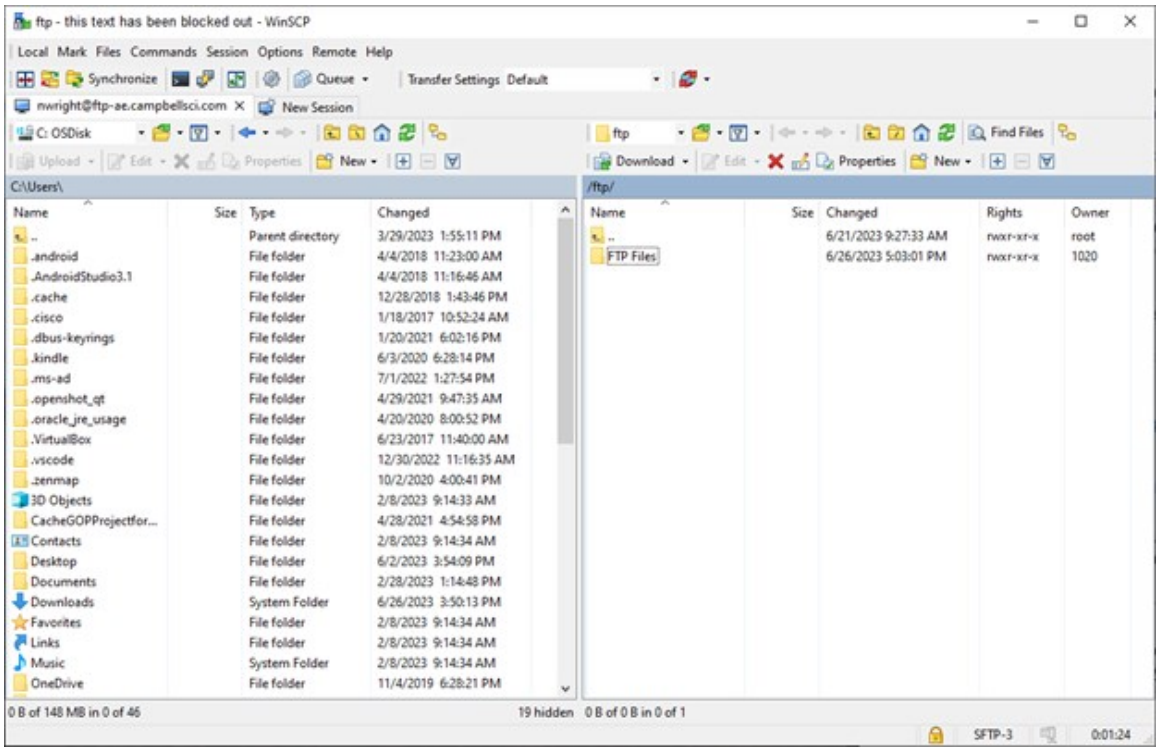
5. Click **OK**.
6. To test the connection once all credentials, settings, and keys have been entered, click the **Login** button at the bottom.



7. As your computer makes the connection to the server you will see a connection status window similar to the following.



8. A successful connection will display the file transfer screen with your local files displayed on the left and the FTP server path on the right.

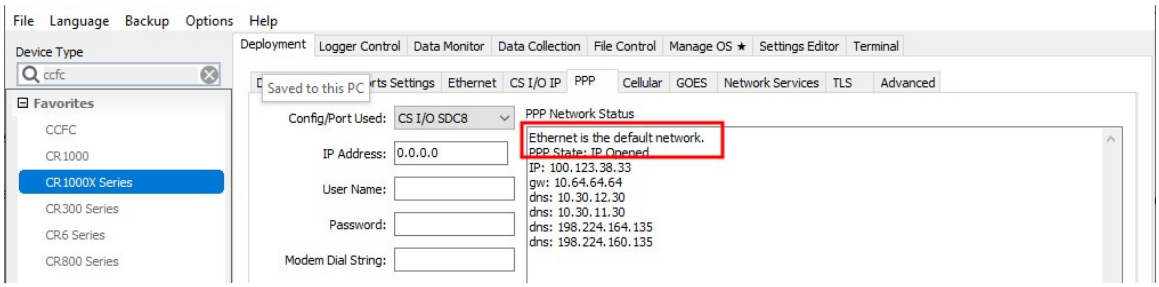


9. Once connected perform the following tasks:
    - Attempt to transfer a file by dragging it from the left into the right half of the window.
    - Make note of any error messages.
    - If performing a test with an SFTP server, make note of any warning messages which may include the algorithm, or server hashes.
10. Address connection issues and error messages with your IT department. Use the collected information to compare to your data logger connection attempt.

6. Verifying IP interfaces are online and correct

This section pertains to data loggers with IP interfaces that are performing communications. IP communications will generally go out the default network interface unless you tell the data logger to do something different. This can create a problem if the traffic (such as FTP) must go out a particular interface. For example, the data logger needs to communicate over a private network via Ethernet, but it needs to send FTP traffic out the PPP interface which has a cellular modem.

To check if you have a routing issue, verify the default interface of your data logger. Use ***Device Configuration Utility*** and any Network Interface tab such as **Ethernet**, **PPP**, **CS I/O IP**. In the following example, we're using the **PPP** tab. The **Network Status** shows that Ethernet is the default network interface.



In this case, the Ethernet connection has no Internet access. It is meant only for communicating with a PLC via Modbus. All FTP requests need to go out to the Internet, but the data logger is sending all traffic out the default Ethernet interface. Add something similar to this to your ***CRBasic*** program:

```
IPRoute ("",1,1)
IPRoute ("192.168.13.95",0,1)
```

The first `IPRoute()` sends all traffic out the PPP interface. The second instruction is used to make an exception to the first rule and always send traffic destined for `the 192.168.13.95` IP address out the Ethernet interface. The first parameter of `IPRoute()` is the IP address of the traffic. Using a pair of double quotes next to each other means all traffic. The second parameter specifies the `IPInterface` where traffic will be sent out from.

The ***CRBasic*** help describes and lets you select viable parameter options.

The two instructions need to be inserted in your program in a place where they will initialize after the data logger network interfaces have come online. Running them earlier can result in them not taking effect. Good places to insert them include a slow sequence scan, the main scan, or a subroutine.

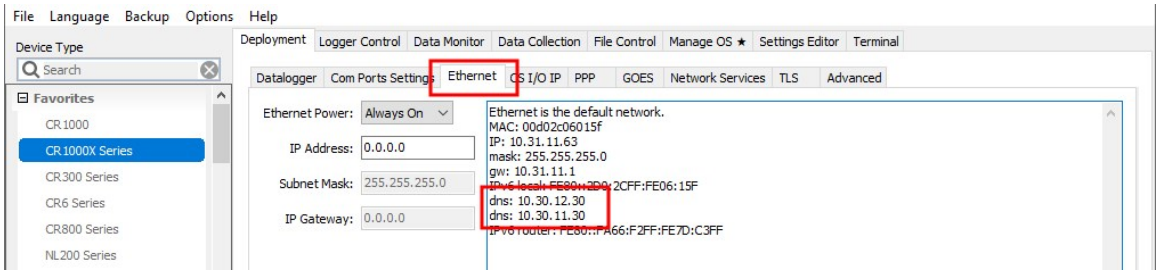
7. Checking DNS issues

If your FTP server is at an address that needs to be resolved by DNS (domain name server), verify that your data logger has DNS servers configured either statically or automatically that can resolve the FTP server name. One way to tell if an address needs to be resolved by DNS is if the address looks like an Internet URL instead of an IP address:

ftp.myserver.com needs DNS resolution.

166.10.33.207 does not need DNS resolution.

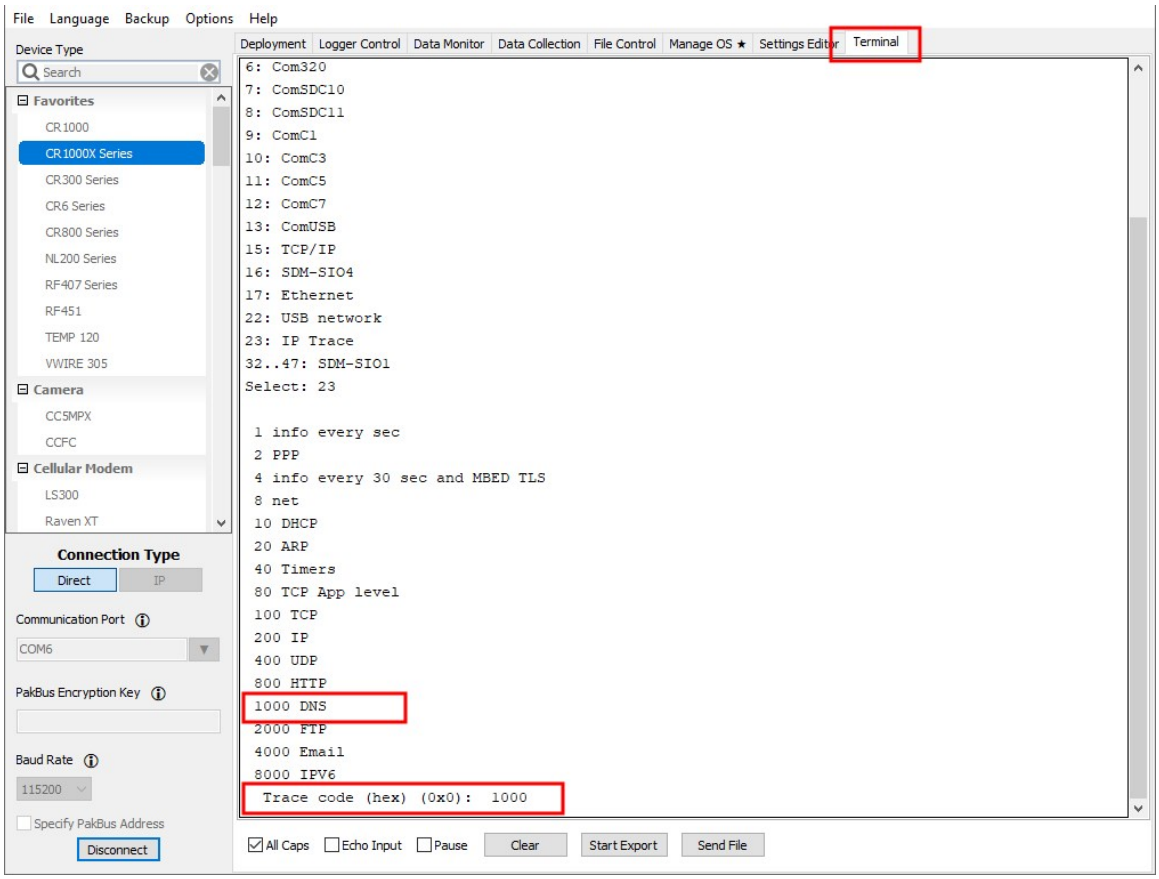
Use ***Device Configuration Utility*** to check if your data logger is using DNS servers with its connection. Select the tab for the interface that will be making the network connection to your FTP, SFTP, or FTPS server. The following uses the Ethernet interface:



This Ethernet interface has two DNS servers. If your address requires DNS resolution and you do not see any DNS servers here, you can manually assign DNS servers in ***Device Configuration Utility*** > **Settings Editor** > **Advanced** tab in both **DNS Server Address 1** and **DNS Server Address 2** fields. Once applied your data logger will direct DNS requests to the server(s) specified.

If you are unsure if your data logger is resolving the FTP server name correctly, using your DNS servers, you can monitor the DNS requests your data logger makes from ***Device Configuration Utility*** **Terminal** mode.

Follow the instruction in Comms watch (sniff) FTP communications (p. 6). In step 4, specify the number corresponding to DNS (**1000**) instead of FTP. Press **Enter**.



Export and save the file as described in Comms watch (sniff) FTP communications (p. 6).

Let the results scroll by for a few minutes, and then click **End Export**. Open the file in any text editor to read the messages.

The following examples will help with interpreting the DNS results messages.

DNS check entry with a time to live

Expect to see many of these with the time to live (ttl) counting down:

```
10:26:59.358 dns_check_entry: "google.com": ttl 60
```

DNS send and request messages

You will see less of these than the DNS check entry. Notice the server 0 on the end. This indicates the request was sent to the first DNS server in the data logger. Server 1 refers to the second DNS server configured in the data logger.

```
11:21:00.003 sending DNS request ID 65130 for name "google.com" to server 0
```

Successful DNS responds with found

This means that your data logger was able to resolve the name to an IP address. In this case, the name was “google.com”, and the IP address it resolved to was 142.250.176.14. In instances where

your data logger is attempting to resolve multiple addresses, you will want to look for the name of the FTP server you are using to ensure that it is correctly resolving.

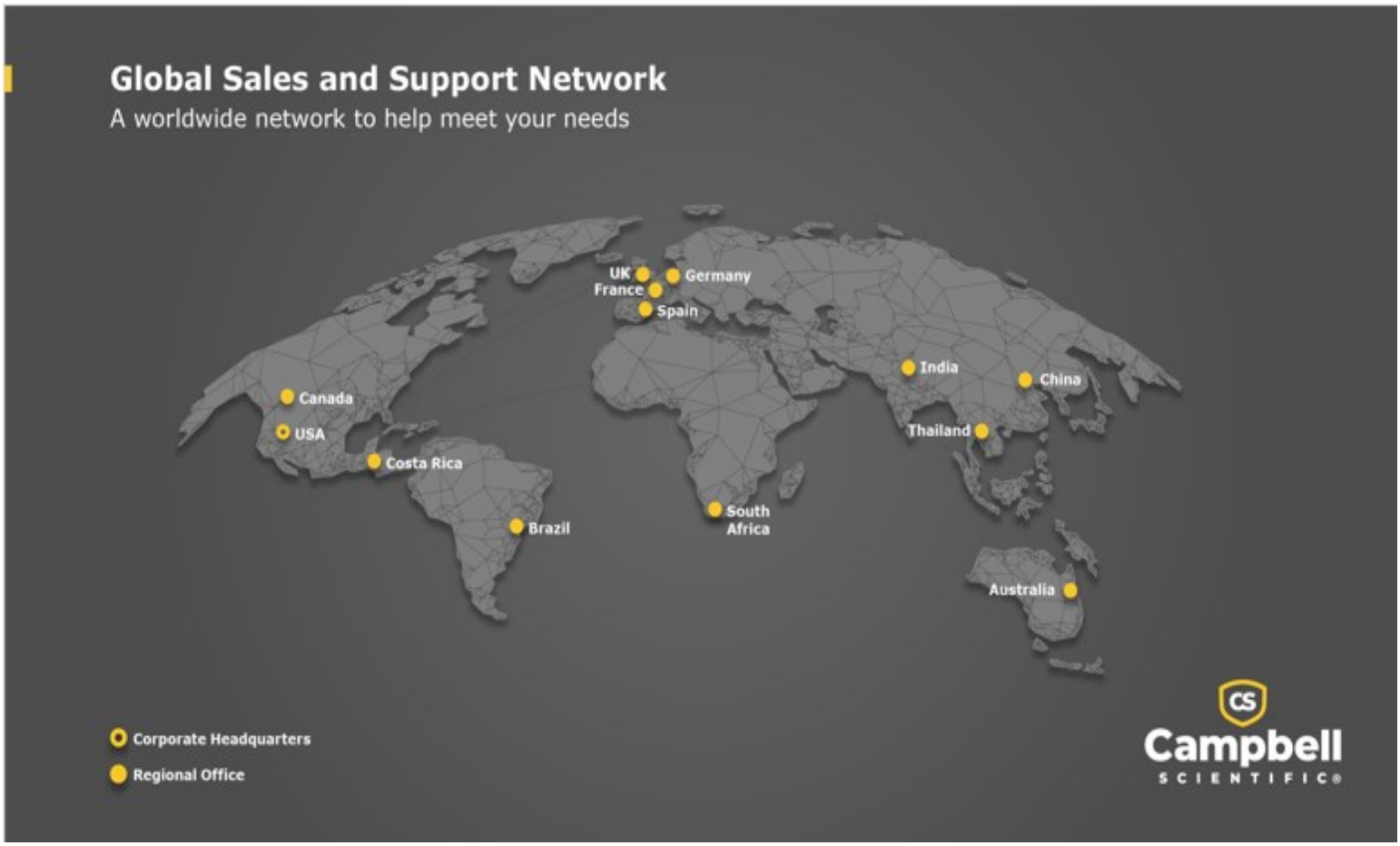
```
10:27:00.003 dns_lookup: "google.com": found = 142.250.176.14
```

Failure to resolve the DNS

```
11:21:00.028 dns_recv: "afakeURL.com": error 3 in flags
```

Do the following to resolve DNS errors:

1. Verify that the FTP address is correct.
2. Verify that the data logger is configured with DNS servers.
3. If applicable, verify that the data logger is connected to the same private network as the server address you are attempting to connect to.
4. If applicable, verify that the data logger is connected to the Internet.
5. Verify that if the data logger is connected to multiple network interfaces that the traffic is being sent out the correct interface. See Verifying IP interfaces are online and correct (p. 26).
6. Obtain alternative DNS servers from your local IT department and switch the data logger to the alternative DNS server(s).
7. Work with your local IT department to obtain the IP address of the server that goes with the server name and use the IP address of the server in your *CRBasic* program instead of the name.



Campbell Scientific Regional Offices

Australia

Location: Garbutt, QLD Australia
Phone: 61.7.4401.7700
Email: info@campbellsci.com.au
Website: www.campbellsci.com.au

Brazil

Location: São Paulo, SP Brazil
Phone: 11.3732.3399
Email: vendas@campbellsci.com.br
Website: www.campbellsci.com.br

Canada

Location: Edmonton, AB Canada
Phone: 780.454.2505
Email: dataloggers@campbellsci.ca
Website: www.campbellsci.ca

China

Location: Beijing, P. R. China
Phone: 86.10.6561.0080
Email: info@campbellsci.com.cn
Website: www.campbellsci.com.cn

Costa Rica

Location: San Pedro, Costa Rica
Phone: 506.2280.1564
Email: info@campbellsci.cc
Website: www.campbellsci.cc

France

Location: Montrouge, France
Phone: 0033.0.1.56.45.15.20
Email: info@campbellsci.fr
Website: www.campbellsci.fr

Germany

Location: Bremen, Germany
Phone: 49.0.421.460974.0
Email: info@campbellsci.de
Website: www.campbellsci.de

India

Location: New Delhi, DL India
Phone: 91.11.46500481.482
Email: info@campbellsci.in
Website: www.campbellsci.in

South Africa

Location: Stellenbosch, South Africa
Phone: 27.21.8809960
Email: sales@campbellsci.co.za
Website: www.campbellsci.co.za

Spain

Location: Barcelona, Spain
Phone: 34.93.2323938
Email: info@campbellsci.es
Website: www.campbellsci.es

Thailand

Location: Bangkok, Thailand
Phone: 66.2.719.3399
Email: info@campbellsci.asia
Website: www.campbellsci.asia

UK

Location: Shepshed, Loughborough, UK
Phone: 44.0.1509.601141
Email: sales@campbellsci.co.uk
Website: www.campbellsci.co.uk

USA

Location: Logan, UT USA
Phone: 435.227.9120
Email: info@campbellsci.com
Website: www.campbellsci.com