ANNEX V COMPRESSED FILE TRANSFER PROTOCOL (Optional)

Compressed File Transfer Protocol (CFTP) provides a protocol for transferring basic SMTP messages over STANAG 5066. It replaces the earlier HMTP protocol, which did not provide compression.

The Compressed File Transfer Protocol (CFTP) **may** be used to meet requirements in the NATO C3 Technical Architecture and the NATO Common Standards Profile (NCSP) for use of the SMTP Protocol. CFTP is used to reliably send compressed SMTP e-mail over a STANAG 5066 HF subnetwork from one message server to another.

CFTP is a vendor-defined standard derived from client definitions made in earlier, non-mandatory, versions of STANAG 5066 Annex F.

V.1. Changes in This Edition

The text of this annex is taken from Annex F Section F.14 of Edition 3.

There are no changes to this text.

V.2. General Requirements

Implementations of STANAG 5066 **may** provide a CFTP client. If provided, a CFTP client **shall**(1) conform to the requirements specified herein.

The CFTP protocol **shall not** be used for Formal or High Grade Military Messaging (i.a. military orders). For Formal or High Grade Military Messaging, ACP 142 as specified in Annex Q **shall**(2) be used. The CFTP protocol **may** be used for informal interpersonal e-mail only.

CFTP **shall**(3) operate within the node model shown, providing transfer services from one SMTP e-mail server to another via the CFTP client.

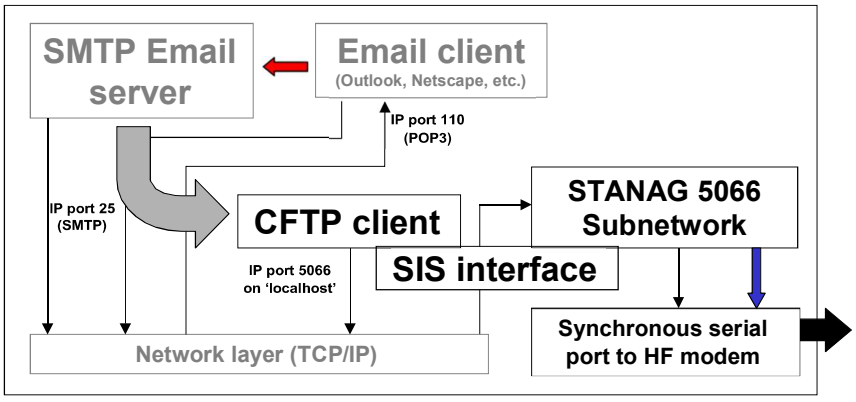

Figure V-1: CFTP Client and Node model

In general, the interaction of the CFTP client with the mail-server is beyond the scope of this STANAG. In operation, when an email message is received at a 5066 node, it is placed in an incoming mail folder (mail spool directory). The CFTP client, also called the Delivery Agent (DA), removes mail from this incoming folder and processes the mail for delivery over HF via 5066. The CFTP DA compresses the message and information about the message, e.g. size, id, recipients, etc. into a file. This compressed file is then transferred to the destination 5066 node(s) using the original form of the Basic File Transfer Protocol (BFTP), referred to here as BFTPv1. The original BFTPv1 format is incorporated directly in the CFTP specification below to free it from dependencies on (and incompatibilities with) BFTP specification found in STANAG 5066 Ed3.

V.3. CFTP Subnetwork Service Requirements

CFTP clients **shall** bind to the HF Subnetwork at SAP ID 12.

V.4. CFTP Connection-Oriented Protocol

The CFTP application **shall** use the earlier form of the RCOP Protocol Data Unit ("RCOPv1") defined in the Figure below. [NB: As a historical note, this corresponds to the definitions of the RCOP protocol found in the original information-only Annex F Edition 1. It does not include the Application Identifier within the PDU Header that has been added in STANAG 5066 Edition 2.]

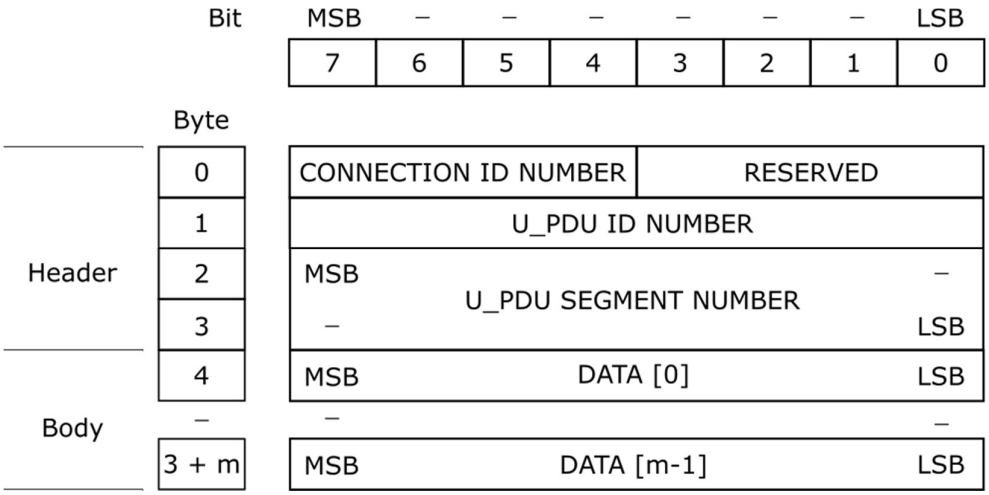

Figure V-2:. CFTP Protocol Data Units (identical in format to (Original) RCOP Protocol Data Units (RCOPv1) from STANAG 5066 Annex F Edition 1)

The following are required for RCOPv1 PDUs:

1. The connection ID number **shall** be a value from 0-15. Connection ID number 0 shall be reserved for non-multiplexed connections.
2. The reserved bits **shall** be set to 0.

3. The U_PDU ID numbers **shall** be assigned consecutively to U_PDU (i.e., files).
4. The U_PDU segment number **shall** be assigned consecutively to segments within a single U_PDU. The first segment transmitted **shall** be assigned segment number 0. If a U_PDU is not segmented, the single segment transmitted **shall** be assigned number 0.

V.4.1. Compressed File-Delivery and Delivery-Confirmation

Compressed files **shall**(1) be transferred from one CFTP client to another using the Edition 1 Basic File Transfer Protocol ('BFTPv1') as defined in the subsections below. Client Delivery confirmation **shall**(2) be provided using the CFTP Message Acknowledgement, defined in the subsequent section (see Section V.4.3), as the body-part of the PDU.

In principle, up to 256 files could be transferred concurrently using the unique RCOPv1 U_PDU ID number for each transfer, using the U_PDU_ID as the identifier to match acknowledgements to the file acknowledged. As there is no negotiation protocol currently defined to determine if a given receiving node supports this capability, a sending node **must** have prior knowledge that a given receiving node supports concurrent multiple- file delivery.

Consequently, the Client Delivery confirmation protocol is nominally stop-and-wait — a new file **should not**(1) be sent with a given U_PDU ID until a message acknowledgement has been received. However, this recommendation **may**(1) be relaxed to allow concurrent multiple-file delivery when the sending node has prior knowledge that the receiving node supports the capability. New implementations of CFTP **should** support concurrent multiple file delivery.

V.4.1.1.BFTPv1 Specification [NB: corresponding to the original Edition 1 BFTP specification]

The format for the basic-file-transfer-protocol data unit Version 1 (BFTPv1) **shall**(1) be in accordance with the following Figure, which defines a header part and a file-data part for the BFTP_PDUv1.

BFTP_HDR	Byte[0]	- - - FILE_DATA[] - - -	Byte[p-1]

Figure V-3:: Basic FTP Version 1 Protocol Data Unit (BFTPv1 PDU)

The detailed structure of the BFTPv1_PDU **shall**(2) be in accordance with the following Figure, and provide the following information fields:

1. BFTPv1_PDU Header Part:
    - SYNCHRONIZATION - two bytes corresponding to the control bytes DLE (Data Link Escape) and STX (Start of Text).

- SIZE_OF_FILENAME - one octet in size.
- FILE_NAME - a variable length field, equal in size to the value specified by the SIZE_OF_FILENAME field.
- SIZE_OF_FILE - a four-octet field.

2. BFTPv1_PDU Body Part:
   - FILE_DATA[] - a variable length field, equal in size to the value specified by the SIZE_OF_FILE field.

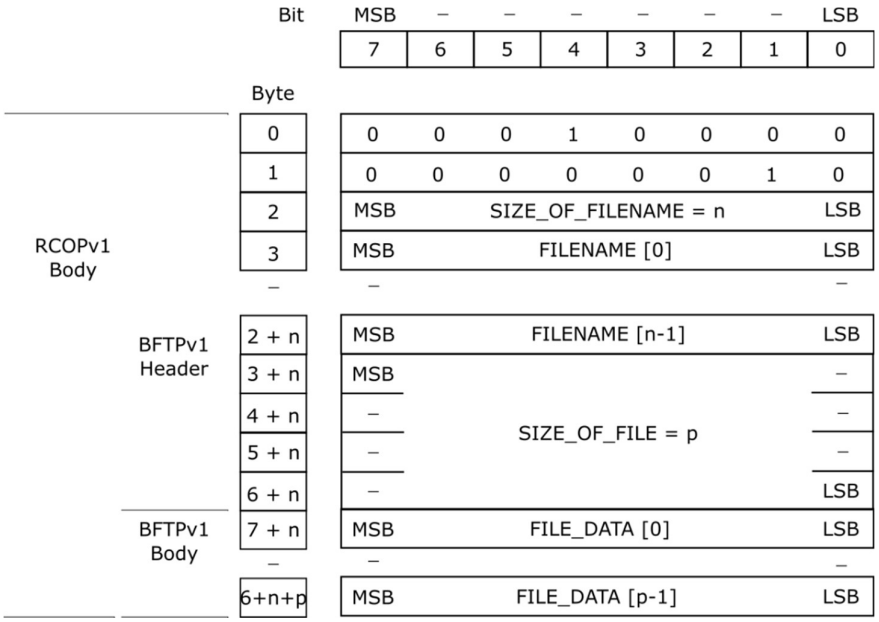

Figure V-4: BFTPv1 Protocol Data Unit Structure

The SIZE_OF_FILENAME field **shall**(1) be a 1-octet fixed-length field whose value (n) **shall**(2) equal the number of octets used to encode the FILENAME field.

The FILENAME field is **shall**(1) be a variable-length field, the size of which **shall**(2) be specified by the value (n) of the field SIZE_OF_FILENAME. This field represents the name of the file sent using the Basic File Transfer Protocol. The first byte of the filename **shall**(3) be placed in the first byte of this field, with the remaining bytes placed in order. The semantics of file names and naming conventions are beyond the scope of this STANAG (e.g., there is no requirement that the filename be a null-terminated character string.)

The SIZE_OF_FILE field **shall**(1) be a 4-octet fixed-length field whose value **shall**(2) specify the size (p) in octets of the file to be sent. The first octet of the SIZE_OF_FILE field **shall**(3) be the highest order byte and the last byte the lowest order byte of the field's binary value.

V.4.2. **BFTPv1 Segmentation and Reassembly Requirements**

If the BFTPv1_PDU exceeds the maximum size of the data field permitted in the RCOPv1 PDU (i.e, if the CTFP_PDU is larger than the MTU_size less 4 octets (i.e., MTU-4)), the CFTP client **shall** segment the BFTPv1 PDU, placing successive segments in RCOPv1 PDUs (original Edition 1 format) with consecutive U_PDU sequence numbers.

When received, the CFTP client **shall**(2) reassemble the BFTPv1 PDU if it determines that the BFTPv1 PDU has been segmented. Subject to local-host file naming conventions, the CFTP client **shall**(2) store the received file with the name transmitted in the header with the file. [*NB: there is no guarantee therefore that the file will be stored on the destination host with the same name that it was sent.]*

V.4.3. **Message Acknowledgement**

Client Delivery confirmation **shall**(1) be provided using the Message Acknowledgement defined below, sent as the body-part of a CFTP/RCOPv1 PDU.

	Bit	MSB 7	6	5	4	3	2	1	LSB 0
	Byte								
BFTPv1 ACK	0	0	0	0	1	0	0	0	0
(RCOPv1 body part)	1	0	0	0	0	1	0	1	1

Figure V-5: BFTPv1 Message Acknowledgement Structure

On receiving the last byte of the message, the receiving client **shall**(2) send the Message Acknowledgement (0x10 0x0B) — with the same RCOPv1 U_PDU_ID NUMBER and CONNECTION ID NUMBER as the message being acknowledged — to confirm that the entire message has been received. (N.B. This is equivalent to the "ZEOF" message of the Z-modem protocol.)

V.5. CFTP Compression/Decompression

The compressed file **shall** be created and decompressed in accordance with RFCs 1950, 1951 and 1952 (i.e., the gzip utility defined in RFC1952).

V.6. CFTP Compressed File Data Format

The compressed file data **shall** be formatted as a series of fields. The fields are separated by the linefeed <LF> character 0x0A. The fields and the order in which they are compressed are described in the following table.

Table V-1 CFTP Mail File Structure

Order of Compression	Field Name	Description
1	MessageID	The MessageID field is represented by an arbitrary string that serves as the ID for the message. The MessageID is unique to an e-mail message. It is not the same as the ID in the email message that follows "Message-ID:" in the header. When the compressed file is decompressed, the MessageID is used as the root filename for the decompressed components. The MessageID must be less than 256 characters and is composed of upper/lowercase alphanumeric
2	RecipientList	The RecipientList is a string containing e-mail addresses extracted from the e-mail message, each address delimited by the "," character (0x2c). The first address in the recipients list is the "Return-Path". There can be cases where there is no return path, e.g. the mail is being bounced by a Mail Transfer Agent. In these cases, the first address will be an empty string (i.e., either a single space [0x20] character or no characters at all) and it will be followed by a comma
3	MessageSize	The MessageSize is encoded as a decimal number in string format. It represents the size (in bytes) of the Message field that follows the MessageSize field.
4	Message	Actual message as received by an SMTP receiver ***i.e. including the terminating sequence <CRLF>.<CRLF> and any additional characters that may be required for transparency as defined by***

Note 1. All characters are 8 bits.
Note 2. The terminating sequence **<CRLF>.<CRLF>** is that shown in Example 1 of RFC 821 and equates to the 5 ASCII Characters with codes, in hexadecimal, of 0x0D, 0x0A, 0x2E, 0x0D, 0x0A.

V.7. Detailed Description of CFTP

1) An e-mail client is used to send an e-mail to an SMTP server.
2) The CFTP application extracts the e-mail message from the directory in which it was placed by the SMTP server. An example e-mail message in the correct format is shown in Figure F-23 below.
3) The CFTP mail-file shall be built as follows:

<MessageID><LF> // The MessageID field shall(1) be represented by a character string. The MessageID **shall**(2) be unique to an e-mail message. It is not the same as the ID in the email message that follows "Message-ID:" in the header. When the compressed file is decompressed, the MessageID **shall** be used as the root filename for the decompressed components. The MessageID **must** be less than 256 characters and **may** be composed of upper/lowercase alphanumeric

<RecipientsList><LF>	//	The RecipientList **shall**(1) be a character string containing e-mail addresses extracted from the e-mail message, each address(0x2c). The first address in the recipients list **shall**(2) be the "Return-Path". There **may** be cases where there is no return path, e.g. the mail is being bounced by a Mail Transfer Agent. In these cases, the first address **shall**(3) be an empty string (i.e., either a single space [0x20] character or no characters at all) followed by a comma (i.e, a "," character with octet value = 0x2c). The recipients list **must** be less than 10240 characters separated by "," character
<MessageSize><LF>	//	The MessageSize shall be encoded as a decimal number in string format terminated by the linefeed character. It represents the size (in bytes) of the Message field that follows.
<Message>	//	The Message field **shall** contain the e-mail message body part(s) extracted from the SMTP envelope.

4) The CFTP message (including header) shall be compressed in accordance with RFCs 1950, 1951 and 1952 using an application such as gzip.
5) The compressed CFTP message shall be encapsulated within a BFTPv1 PDU (i.e., it has a BFTPv1 header prepended to it, and the CFTP message shall be byte aligned within the FILE_DATA[] field of the BFTPv1 PDU.
6) The BFTPv1 message (i.e., BFTPv1 PDU) shall be segmented if necessary.
7) Each BFTPv1 PDU segment shall have an RCOPv1 header added (in accordance with Annex F.14.3.).
8) Each RCOPv1 packet shall be packaged into an S_UNIDATA_REQUEST and transferred using a Soft Link Data Exchange.
9) On reception the BFTPv1 message shall be reassembled, if required, and decompressed using a method compliant with RFC 1952 and the CFTP message reconstructed.
10) The received email messages shall be forwarded to an SMTP server using a standard SMTP dialogue based on information extracted from the CFTP header and inserting the "message" field into the payload of the SMTP message generated.

```
Received: from northampton (unverified [127.0.0.1]) by northampton.pdw<CRLF>
 (Rockliffe SMTPRA 4.2.4) with SMTP id
 <B0000000133@northampton.pdw> for
<root@essex.pdw>;<CRLF>
 Wed, 9 May 2001 12:09:03 +0100<CRLF>
Message-ID:
<001f01c0d878$74da90d0$0d02a8c0@pdw><CRLF>
From: "Northampton"
<administrator@northampton.two><CRLF> To:
<root@essex.pdw><CRLF>
Subject: Test <CRLF>
Date: Wed, 9 May 2001 12:09:03
+0100<CRLF> MIME-Version:
1.0<CRLF>
Content-Type:
        text/plain;<CRLF>
        charset="iso-
        8859-1"<CRLF>
Content-Transfer-Encoding:
7bit<CRLF> X-Priority:
3<CRLF>
X-MSMail-Priority: Normal<CRLF>
X-Mailer: Microsoft Outlook Express 5.50.4522.1200<CRLF>
X-MimeOLE: Produced By Microsoft MimeOLE V5.50.4522.1200<CRLF>
<CRLF>
This is the body of the test email<CRLF>
<CRLF>
.<CRLF>
```

Figure V-6: Example email in the Correct Format

The red and blue text above is the message body with text in blue being the terminating sequence:

<CRLF>.<CRLF> i.e 0x0D, 0x0A, 0x2E, 0x0D, 0x0A.