Traditional Encoding

The Traditional Encoding is the encoding used in all versions of the standard prior to 2008. The format and rules for this encoding of the ANSI/NIST-ITL 1-2011 version are consistent with ANSI/NIST-ITL 1-2007.

A transaction shall consist of one or more records. For each record, several information fields appropriate to that record type shall be present. Each information field may contain one or more basic single-valued information items. Taken together these items are used to convey different aspects of the data contained in that field. An information field may also consist of one or more information items grouped together and repeated multiple times within a field. Such a group of information items is known as a subfield. An information field may therefore consist of one or more subfields of information items.

The first field in all records shall be labeled as field "1" and contain the length in bytes of the record. With the exception of the Type-1 record, the second field shall be labeled as field "2" and contain the image designation (IDC).

The order of fields for Type-4 and Type-8 records is fixed. With the exception of the first two fields, the order of the remaining fields of the Type-7 record is user-defined. All fields and data in Type-4, Type-7 and Type-8 records shall be records as binary information.

The data in the Type-1 record shall always be recorded in variable length fields using the 7-bit American National Standard Code for Information Interchange (ASCII) as described in ANSI X3.4-1986. For purposes of compatibility, the eighth (leftmost) bit shall contain a value of zero.

Textual fields in Record Types 10-99 may occur in any order after the first two fields and contain the information as described for that particular numbered field, except for field 999, which shall be the concluding field.

Information separators

In the records Type-1, Type-2, Type-9 through Type-99, mechanisms for delimiting information are implemented by use of the four ASCII information separators. The delimited information may be items within a field or subfield, fields within a logical record, or multiple occurrences of subfields. These information separators are defined in the referenced standard ANSI X3.4 whose code table is shown below. These characters are used to separate and qualify information in a logical sense. Viewed in a hierarchical relationship, the File Separator *"FS"* character is the most inclusive followed by the Group Separator *"GS"*, the Record Separator *"RS"*, and finally the Unit Separator *"US"* characters.

The four characters are only meaningful when used as separators of data items in the fields of the ASCII text records. There is no specific meaning attached to these characters occurring in binary image records and binary fields – they are just part of the exchanged data.

Information separators should be functionally viewed as an indication of the type data that follows.

The *"US"* character shall separate individual information items within a field or subfield. This is a signal that the next information item is a piece of data for that field or subfield.

Multiple subfields within a field separated by the *"RS"* character signals the start of the next group of repeated information item(s).

The *"GS"* separator character used between information fields signals the beginning of a new field preceding the field identifying number that shall appear.

Multiple records within a transaction are separated by the *"FS"* character, which signals the end of a logical record.

Use of separators within the Type-1, Type-2 and tagged-field records shall always be observed. The "US" separator shall separate multiple items within a field or subfield; the "RS" separator shall separate multiple subfields, and the "GS" separator shall separate information fields.

The following is a detailed description of the separator characters.

FN	is the number of a field (including record type) within a tagged-field record.
IF	is the information field associated with an FN.
II	is the information item belonging to an IF.
SF	is the subfield used for multiple entries of an II or an IF.

F S	File separator character – separates logical records.
G S	Group separator character – separates fields.
R S	Record separator character – separates repeated subfields.
U S	Unit separator character – separates information items.

The $\overset{G}{S}$ is used between fields – the $\overset{F}{S}$ between logical records:

FN$_j$:IF $\overset{G}{S}$ **FN$_k$: …** $\overset{G}{S}$ **FN$_1$: IF** $\overset{G}{S}$ **…** $\overset{F}{S}$

For fields with more than one information item, the $\overset{R}{S}$ is used:

FN$_j$:II$_a$ $\overset{U}{S}$ **II$_b$** $\overset{G}{S}$ **FN$_k$ …** $\overset{F}{S}$

For fields with multiple subfields, the $\overset{R}{S}$ is used:

FN$_j$: II$_a$ $\overset{U}{S}$ **II$_b$** $\overset{R}{S}$ **II$_a$** $\overset{U}{S}$ **II$_b$** $\overset{G}{S}$ **FN$_k$ …** $\overset{F}{S}$

which can be expressed as:

FN$_j$: SF$_1$ $\overset{R}{S}$ **SF$_2$** $\overset{G}{S}$ **FN$_k$ …** $\overset{F}{S}$

Normally**,** there should be no empty fields or information items and therefore only one separator character should appear between any two data items. The exception to this rule occurs for those instances where the data in fields or information items in a transaction are unavailable, missing, or optional, and the processing of the transaction is not dependent upon the presence of that particular data. In those instances, multiple and adjacent separator characters shall appear together rather than requiring the insertion of dummy data between separator characters.

Consider the definition of a field that consists of three information items. If the information for the second information item is missing, then two adjacent *"US"* information separator characters would occur between the first and third information items. If the second and third information items were both missing, then three separator characters should be used – two *"US"* characters in addition to the terminating field or subfield separator character. In general, if one or more mandatory or optional information items are unavailable for a field or subfield, then the appropriate number of separator character should be inserted.

It is possible to have side-by-side combinations of two or more of the four available separator characters. When data are missing or unavailable for information items, subfields, or fields, there must be one fewer separator characters present than the number of data items, subfields, or fields required.

Record layout

For tagged-field logical records (Type-1, Type-2, Type-9 through Type-99), each information field that is used shall be numbered in accordance with this standard. The format for each field shall consist of the logical record type number followed by a period ".", a field number followed by a colon ":", followed by the information appropriate to that field. The tagged-field number can be any one to nine-digit number occurring between the period "." And the colon ":". It shall be interpreted as an unsigned integer field number. This implies that a field number of "2.123:" is equivalent to and shall be interpreted in the same manner as a field number of "2.000000123:".

NOTE: For purposes of illustration throughout this document, a three-digit number shall be used for enumerating the fields contained in each of the tagged-field logical records described herein. Field numbers will have the form of "TT.xxx:" where the "TT" represents the one- or two-character record type followed by a period. The next three characters comprise the appropriate field number followed by a colon. Descriptive ASCII information or the image data follows the colon.

Logical Type-1, Type-2, and Type-9 records contain only ASCII textual data fields. The entire length of the record (including field numbers, colons, and separator characters) shall be recorded as the first ASCII field within each of these record types. The ASCII File Separator "FS" control character (signifying the end of the logical record or transaction) shall follow the last byte of ASCII information and shall be included in the length of the record.

The Type-4 and Type-8 records contain only binary data recorded as ordered fixed-length binary fields. The entire length of the record shall be recorded in the first four-byte binary field of each record. For these binary records, neither the record number with its period, nor the field identifier number and its following colon, shall be recorded. Furthermore, as all the field lengths of these six records are either fixed and specified, none of the four separator characters (*"US", "RS", "GS",* or *"FS"*) shall be interpreted as anything other than binary data. For these binary records, the "*FS*" character shall not be used as a record separator or transaction terminating character.

The Type-10 through Type-99 records combine ASCII fields with a single binary image field. Each ASCII field contains a numeric field identifier and its descriptive data. The last physical field in these records shall always be numbered "999" and shall contain the data placed immediately following the colon (":") of the field identifier. The record length field shall contain the length of the record. The ASCII File Separator *"FS"* control character shall follow the last byte of the compressed or uncompressed image data. The *"FS"* character shall signify the end of the logical record or transaction and shall be included as part of the record length.

The base-64 encoding scheme, found in email, shall be used for converting non-ASCII text into ASCII form. By convention, any language or character set text string following the Start-of-Text character sequence will be base-64 encoded for subsequent processing. The field number including the period and colon, for example "2.001:", in addition to the *"US"*, *"RS"*, *"GS"*, and *"FS"* information separators shall appear in the transaction as 7-bit ASCII characters without conversion to base-64 encoding.

Notes on Encoding for specific Record Types

Record Type-1

Field 1.001 shall begin with "1.001:" followed by the length of the record including every character of every field contained in the record and the information separators. The "GS" separator character shall separate the length code of Field 1.001 from the next field.

The year, month, and day values in **Field 1.005** are concatenated "YYYYMMDD". The complete date must be a legitimate date.

In **Field 1.013,** the default is
"1.103:NORAM{US}{GS}"

For **Field 1.015**, the Greenwich mean time is represented as "YYYYMMDDhhmmssZ", a 15-character string that is the concatenation of the date with the GMT and concludes with a "Z". Here YYYY is the year, MM the month, DD the day, hh the hour, mm the minute, ss the second and Z is a set character concluding the string.

All of the fields in the Type-1 transaction record must be recorded using the 7-bit ASCII code, which is the default character set code within a transaction. In order to affect data and transaction interchanges between non-English speaking or based agencies, a technique is available to encode information using character sets other than 7-bit ASCII. Fields from theType-1 logical record and ASCII "LEN" and "IDC" text fields must still be encoded using 7-bit ASCII. But all other designated text fields can be encoded using alternate character sets. The general mechanism for accomplishing this provides for backward compatibility with existing readers, supports multiple character sets in a single text string, and handles internationally accepted character sets and text order conventions such as ISO character sets, UTF-8, and Unicode.

To switch character sets within a transaction, the Type-1 record shall contain a field listing the Directory of Character Sets (DCS) used in the transaction. The DCS is an ordered list of triples, each consisting of 3 information items containing an identifying code, the name of an international character set, and its version. The code for a specific character set and other special codes shall be embedded in the transaction to signal the conversion to a different international character set. The ASCII Start-of-Text "STX" character (0x02) followed by the equal sign "=" is used to signal the change to an alternate character set defined by the specific DCS code that follows. The entire Start-of-Text sequence is terminated by a single instance of the ASCII End-of-Text "ETX" character (0x03). This alternate character set will remain active until a closing "ETX" character is encountered or the next ASCII information separator character is encountered.

All text between the STX sequence and the closing ETX character shall be encoded in base-64 notation. This is true even when the 7-bit ASCII character set is specified.

Usage of UTF-8 is allowed as an alternative to the technique that requires the usage of the ASCII “STX” and “ETX” characters to signify the beginning or end of international characters. UTF-8 is only allowed to be used for user-defined fields of all the tagged-field records. Even though there is no overlap within the character sets used with UTF-8, it should be registered in the Type-1 record within DCS **Field 1.15** (Directory of Character Sets).

Immediately following the last information item in the Type-1 record, an “FS” separator character shall be used to separate it from the next logical record. This “FS” character shall replace the “GS” character that is normally used between information fields. (This is the case with all Type records)

Layout of Record Types 4, 7 and 8

For the Type-4 and Type-8 records, the content and order of the recorded fields are specified by this standard. With the exception of the first two fields, the remaining fields of the Type-7 logical image record are all user-defined. All fields and data in these record types shall be recorded as binary information.

Layout of Record Type-9 minutiae data record

For **Field 9.006,** if the exact finger, palm or plantar position cannot be determined, multiple positions may be entered, separated by the “RS” character. When it is not possible to uniquely identify the fingerprint class, reference fingerprint classes may be used and shall be separated by the “RS” character.

In **Field 9.008,** the “RS’ separator shall separate multiple occurrences of core positions.
In **Field 9.009,** the “RS’ separator shall separate multiple occurrences of delta positions.

All information items in **Field 9.012** shall be separated from the subsequent items by the “US” character. (Note that x, y and theta is considered to be a single item, not three items). If the quality measure is not available but the type and/or ridge count data is present, then a “US” separation character shall be included. If the minutia type information is not available for this minutia but ridge count data is present, then a “US” information separator must be included. The fifth information item is optional ridge count data. It shall be formatted as a series of information items, each consisting of a minutia number and a ridge count. This information shall be conveyed by listing the identity (index number) of the distant minutia followed by a comma, and the ridge count to that minutia. The “US” character shall be used to separate these information items. These information items shall be repeated as many times as required for each minutia (subfield). A Record separator character “RS” shall be used at the end of the information items to introduce the first information item concerning data for the next minutia.

Layout of Record Type-10 facial or SMT record

For facial feature points (**Field 10.029),** the first information item (always “1”) is followed by the “US” separator character. The second information item is feature point code, followed by “US”. The third is the X coordinate, followed by “US”. The fourth item is the Y coordinate. Each feature block must be separated by the “RS” separator character.

An example transaction that represents two feature points of eye centers is “10.029:1{US}12.2{US}120{US}130{US}1{US}12.1{US}240{US}129{GS}”

Layout of RecordType-13 variable-resolution latent record

For **Field 13.014,** multiple portions of the EJI can be listed and separated by the “RS” separator character.

For **Field 13.015,** the six information items within the field are separated by five “US” separators; individual full finger or segment definitions may be repeated as subfields separated by the “RS” separator.

Field 13.024 may contain one or more subfields, each consisting of four information items separated by the “US” character. The subfield may be repeated for each latent image and quality algorithm used, separated by the “RS” character.

Base-64 encoding scheme

The base-64 Content-Transfer-Encoding is designed to represent arbitrary sequences of octets in a form that need not be humanly readable. The encoding and decoding algorithms are simple, but the encoded data are consistently only about 33 percent larger than the uuencoded data. This encoding is virtually identical to the one used in Privacy Enhanced Mail (PEM) applications, as defined in RFC 1421. The base-64 encoding is adapted from RFC 1421, with one change: base-64 eliminates the “*” mechanism for embedded clear text.

A 65-character subset of US-ASCII is used, enabling 6 bits to be represented per printable character. (The extra 65th character, “=”, is used to signify a special processing function.)

NOTE: This subset has the important property that it is represented identically in all versions of ISO 646, including US ASCII and all characters in the subset are also represented identically in all versions of EBCDIC. Other popular encodings, such as the encoding used by the uuencode utility and the base-85 encoding specified as part of Level 2 PostScript, do not share these properties, and thus do not fulfill the portability requirements a binary transport encoding for mail must meet.

The encoding process represents 24-bit groups of input bits as output strings of 4 encoded characters. Proceeding from left to right, concatenating 3 8-bit input groups forms a 24-bit input group. These 24 bits are then treated as 4 concatenated 6-bit groups, each of which is translated into a single digit in the base-64 alphabet. When encoding a bit stream via the base-64 encoding, the bit stream must be presumed to be ordered with the most significant bit first. That is, the first bit in the stream will be the high-order bit in the first byte, and the eighth bit will be the low-order bit in the first byte, and so on.

Each 6-bit group is used as an index into an array of 64 printable characters. The character referenced by the index is placed in the output string. These characters, identified in Table C1, below, are selected so as to be universally representable, and the set excludes characters with particular significance to SMTP (e.g., ".", CR, LF) and to the encapsulation boundaries defined in this document (e.g., "-").

The output stream (encoded bytes) must be represented in lines of no more than 76 characters each. All line breaks or other characters not found in Table C1 must be ignored by decoding software. In base-64 data, characters other than those in Table C1, line breaks, and other white space probably indicate a transmission error, about which a warning message or even a message rejection might be appropriate under some circumstances.

Table 1 Base-64 alphabet

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	I	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		
14	O	31	f	48	w	(pad)	=
15	P	32	g	49	x		
16	Q	33	h	50	y		

Special processing is performed if fewer than 24 bits are available at the end of the data being encoded. A full encoding quantum is always completed at the end of a body. When fewer than 24 input bits are available in an input group, zero bits are added (on the right) to form an integral number of 6-bit groups. Padding at the end of the data is performed using the '=' character. Since all base-64 input is an integral number of octets, only the following cases can arise: (1) the final quantum of encoding input is an integral multiple of 24 bits; here, the final unit of encoded output will be an integral multiple of 4 characters with no "=" padding, (2) the final quantum of encoding input is exactly 8 bits; here, the final unit of encoded output will be two characters followed by two "=" padding characters, or (3) the final quantum of encoding input is exactly 16 bits; here, the final unit of encoded output will be three characters followed by one "=" padding character.

Because it is used only for padding at the end of the data, the occurrence of any '=' characters may be taken as evidence that the end of the data has been reached (without truncation in transit). No such assurance is possible, however, when the number of octets transmitted was a multiple of three.

Any characters outside of the base-64 alphabet are to be ignored in base-64-encoded data. The same applies to any illegal sequence of characters in the base-64 encoding, such as "=====" .Care must be taken to use the proper octets for line breaks if base-64 encoding is applied directly to text material that has not been converted to canonical form. In particular, text line breaks must be converted into CRLF sequences prior to base-64 encoding. The important thing to note is that this may be done directly by the encoder rather than in a prior cannibalization step in some implementations.

NOTE: There is no need to worry about quoting apparent encapsulation boundaries within base-64-encoded parts of multipart because no hyphen characters are used in the base-64 encoding.

Table 2: 7-bit American Standard Code for Information Interchange (ASCII)

B_7 = M S B →						0	0	0	0	1	1	1	1
b_6 →						0	0	1	1	0	0	1	1
b_5 →						0	1	0	1	0	1	0	1
Bits	b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	COLUMN → ROW ↓	0	1	2	3	4	5	6	7
	0	0	0	0	0	NUL	DLE	SP	0	@	P	N	p
	0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
	0	0	1	0	2	STX	DC2	“	2	B	R	b	r
	0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
	0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
	0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
	0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
	0	1	1	1	7	BEL	ETB	N	7	G	W	g	w
	1	0	0	0	8	BS	CAN	(	8	H	X	h	x
	1	0	0	1	9	HT	EM	)	9	I	Y	i	y
	1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
	1	0	1	1	11	VT	ESC	+	;	K	[	K	{
	1	1	0	0	12	FF	FS	,	<	L	\	*	|
	1	1	0	1	13	CR	GS	-	=	M	]	m	}
	1	1	1	0	14	SO	RS	.	>	N	^	n	˜
	1	1	1	1	15	SI	US	/	?	O	_	o	DEL