平文	
char(1)	DataSubVersion（旧データでは、ver.2.00～は "5", ver.2.70～は "6"）
char(1)	reserved=NULL(0)
char(1)	MissTypeLimit
char(1)	fBroken
string[16]	ファイル・シグニチャ ="_AttacheCaseData" || "_Atc_Broken_Data"
const int(4)	DATA_FILE_VERSION = 105
const int(4)	TYPE_ALGORISM = 1:Rijndael
const int(4)	AtcHeaderSize : 暗号化ヘッダのデータサイズ
byte[32] IV	暗号化されたヘッダのInitialization Vector (IV)
暗号化	
const string	"Passcode:AttacheCase¥n"
const string	"LastDateTime:" + DateTimeToStr(Now()) + "¥n"

string[ ]	{
	Fn_{n}:FilePath ¥t FileSize ¥t FileAttr ¥t 更新日 ¥t 更新時 ¥t 作成日 ¥t 作成時 ¥n
	…
	…
	}

string[ ]	{
	U_{n}:FilePath ¥t FileSize ¥t FileAttr ¥t 更新日 ¥t 更新時 ¥t 作成日 ¥t 作成時 ¥n
	…
	…
	}

byte[32] IV	暗号化本体のIV
DATA_MAIN	CBCモードで暗号化される本体データ

32 bytes

平文		
short 2bytes	暗号化したアプリケーションのバージョン（3xxxx）	2
char[1]	ミスタイプ回数制限（デフォルト：3）	1
char[1]	ファイルを破壊するか否か（デフォルト：false=0）	1
char[16] 16 bytes	ファイル・シグニチャ ="_AttacheCaseData" || "_Atc_Broken_Data"	16
int 4bytes	暗号化データバージョン（=130） 105 < x < 200	4 合計： 24 bytes
int 4 bytes	暗号化ヘッダーデータサイズ	4 合計： 28 bytes
byte[8] 8 bytes	ランダムソルト→**Rfc2898DeriveBytes** クラスから「**key**」と「**IV**」を導出する	8 合計： 36 bytes
暗号化		
	トークン（="atc3¥n"）	
string[ ]	[格納されていているファイル情報のリスト] NUM:FilePath ¥t FileSize ¥t FileAttr ¥t 更新日 ¥t 更新時 ¥t 作成日 ¥t 作成時 ¥t SHA-1 ¥n [格納されていているファイル情報のリスト(ver.3.2.3.0～)] NUM:FilePath ¥t FileSize ¥t FileAttr ¥t 更新日 ¥t 更新時 ¥t 作成日 ¥t 作成時 ¥t チェックサム(SHA-256) ¥t UTC更新日時 ¥t UTC作成日時 ¥n … … …	
byte[]	CBCモードで暗号化される本体データ … … …	

平文		
short 2bytes	暗号化したアプリケーションのバージョン（4xxxx）	2
char[1]	ミスタイプ回数制限（デフォルト：3）	1
char[1]	ファイルを破壊するか否か（デフォルト：false=0）	1
char[16] 16 bytes	ファイル・シグニチャ =″_AttacheCaseData″ || ″_Atc_Broken_Data″ || ″_AttacheCase_Rsa″ = ブロック暗号 or 破壊されたデータ or 公開鍵暗号	16
int 4bytes	暗号化データバージョン（=140） 105 < x < 200 ※130 : ver.3; 140 : ver.4	4 総計：24 bytes
int 4 bytes	暗号化前のヘッダーデータサイズ	4
byte[16] 16 bytes	GUID	16
byte[8] 8 bytes	ランダムソルト →**Rfc2898DeriveBytes** クラスから「**key**」と「**IV**」を導出する　両暗号に使用	8 総計：52 bytes
byte[256] 256 bytes	公開鍵暗号RSA（″_AttacheCase_Rsa″）のときは、ここに暗号化されたパスワードを格納する	256 総計：(308 bytes)
暗号化		
byte[4]	トークン（=″atc4″）	4
byte[ ] 可変長	【ヘッダーデータ（暗号化）】 **AesManaged（キー:256bit, ブロック:128bit）CBC、PKCS7** [シリアルデータとして保存する] FileNameSize(2byte) | FilePath(可変) | FileSize(Int64) | FileAttr(int) | UTC更新日時(Int64) | UTC生成日時(Int64) | FileSize==0 ? ″″ : MD5(16byte)※ | … … … … ※データ改竄チェックではなく、データ破損チェック程度なら低負荷のMD5で十分と判断。	暗号化前の ヘッダーサイズを
byte[] 可変長	パディング（0～15byte）	
byte[] 可変長	【本体データ（暗号化）】 **AesManaged（キー:256bit, ブロック:128bit）CBC、PKCS7** **＋** **DeflateStream(.NET Framework 4.5以降はzlib)** … … ※今まではRijndaelManagedのキー256bit, ブロック256bitを使ってきたが、 他へ移植するときの取り回しの良さから、事実上の標準であるAES256を選択。	
byte[] 可変長	パディング（0～15byte）	