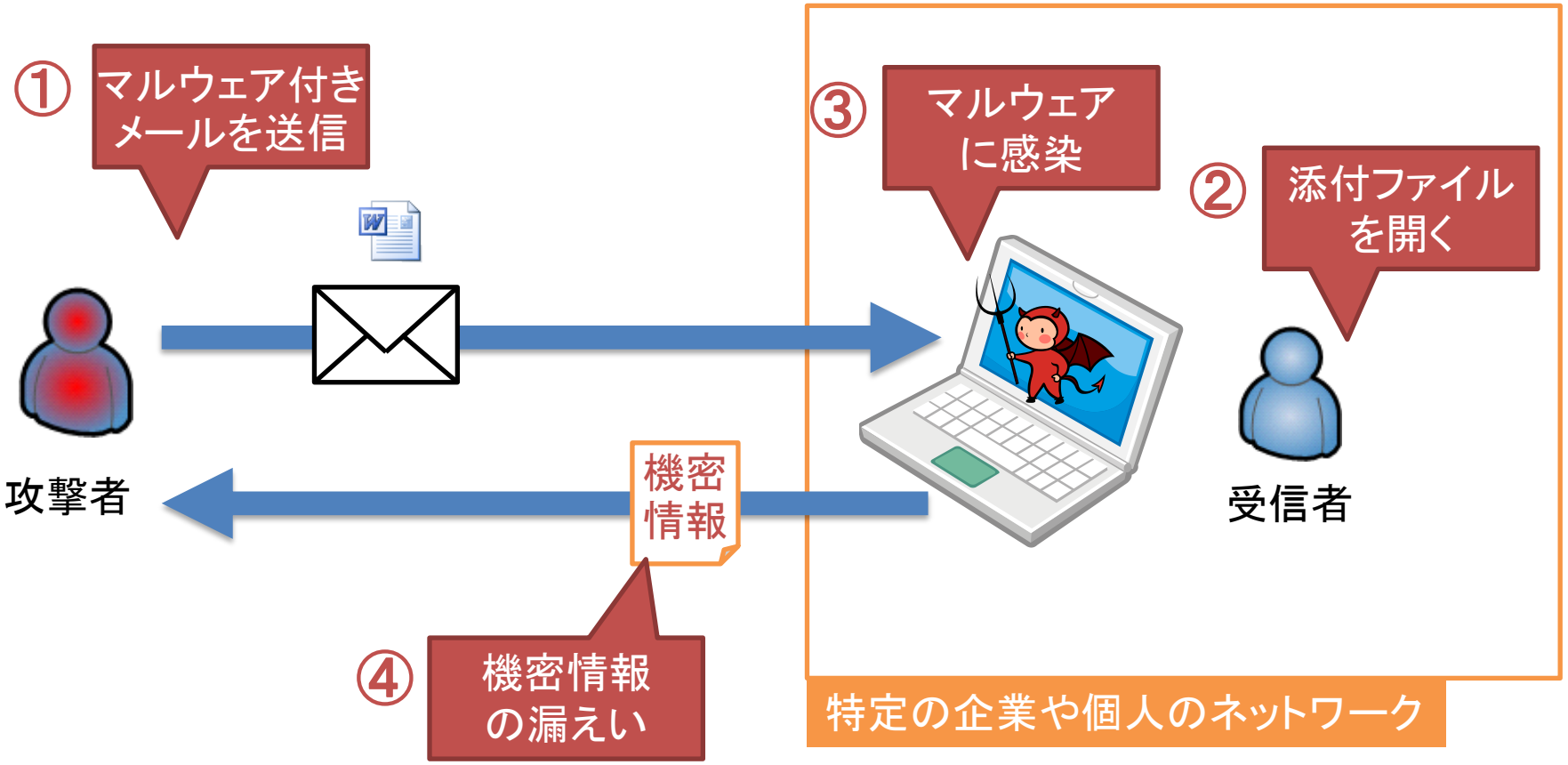
PDFの構造検査による悪性PDFの検知

大坪　雄平
三村　　守
田中　英彦

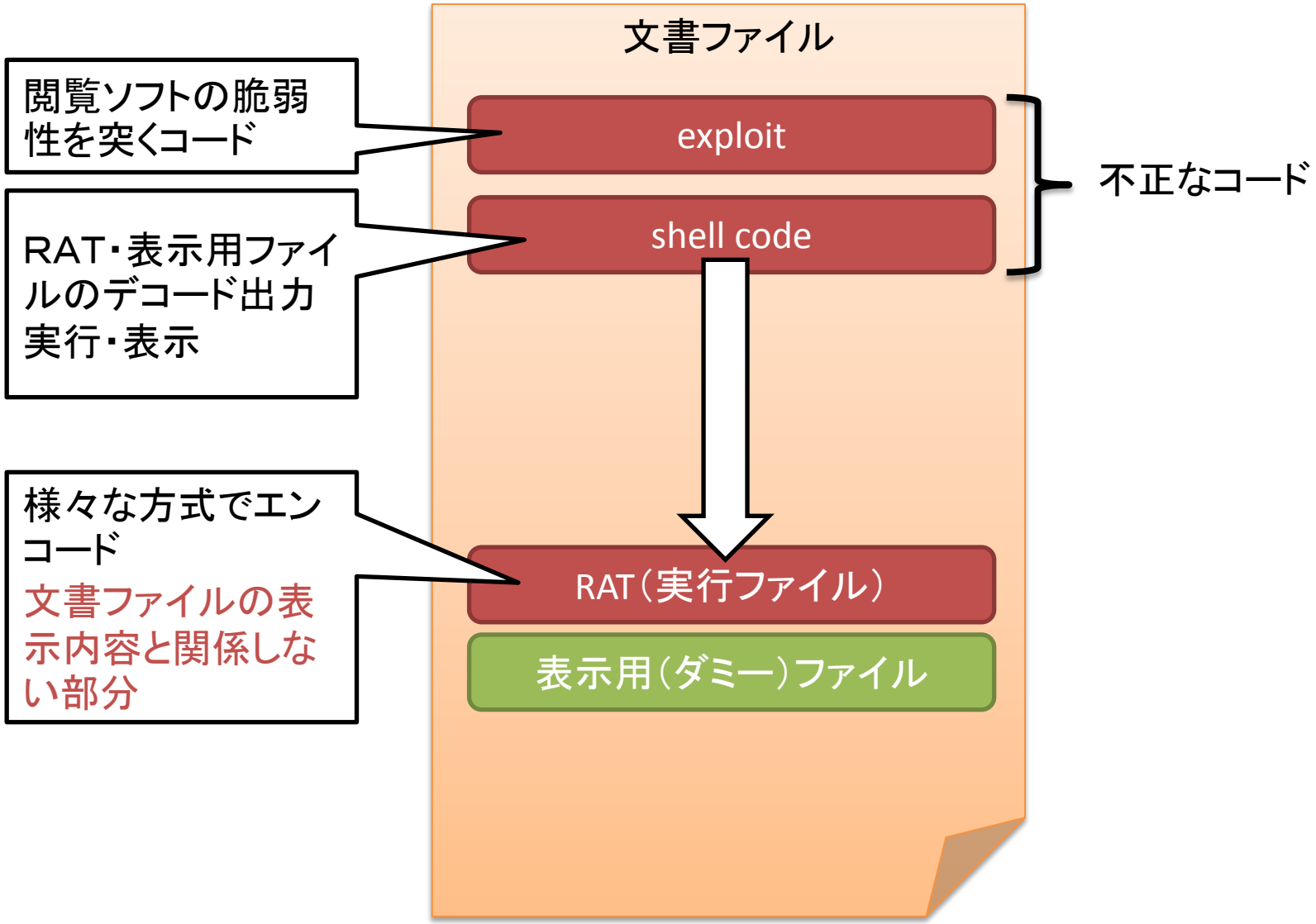
目次

標的型攻撃の脅威



実行ファイルが文書ファイルに埋め込まれた場合
受信者がマルウェアに気づくことは難しい

実行ファイルが埋め込まれた悪性文書ファイルの構造



関連研究

静的解析

特徴	課題
文書ファイルに不正なコードや実行ファイルが埋め込まれているか否かを判定	不正なコードや実行ファイルは様々な方式でエンコードされている

動的解析

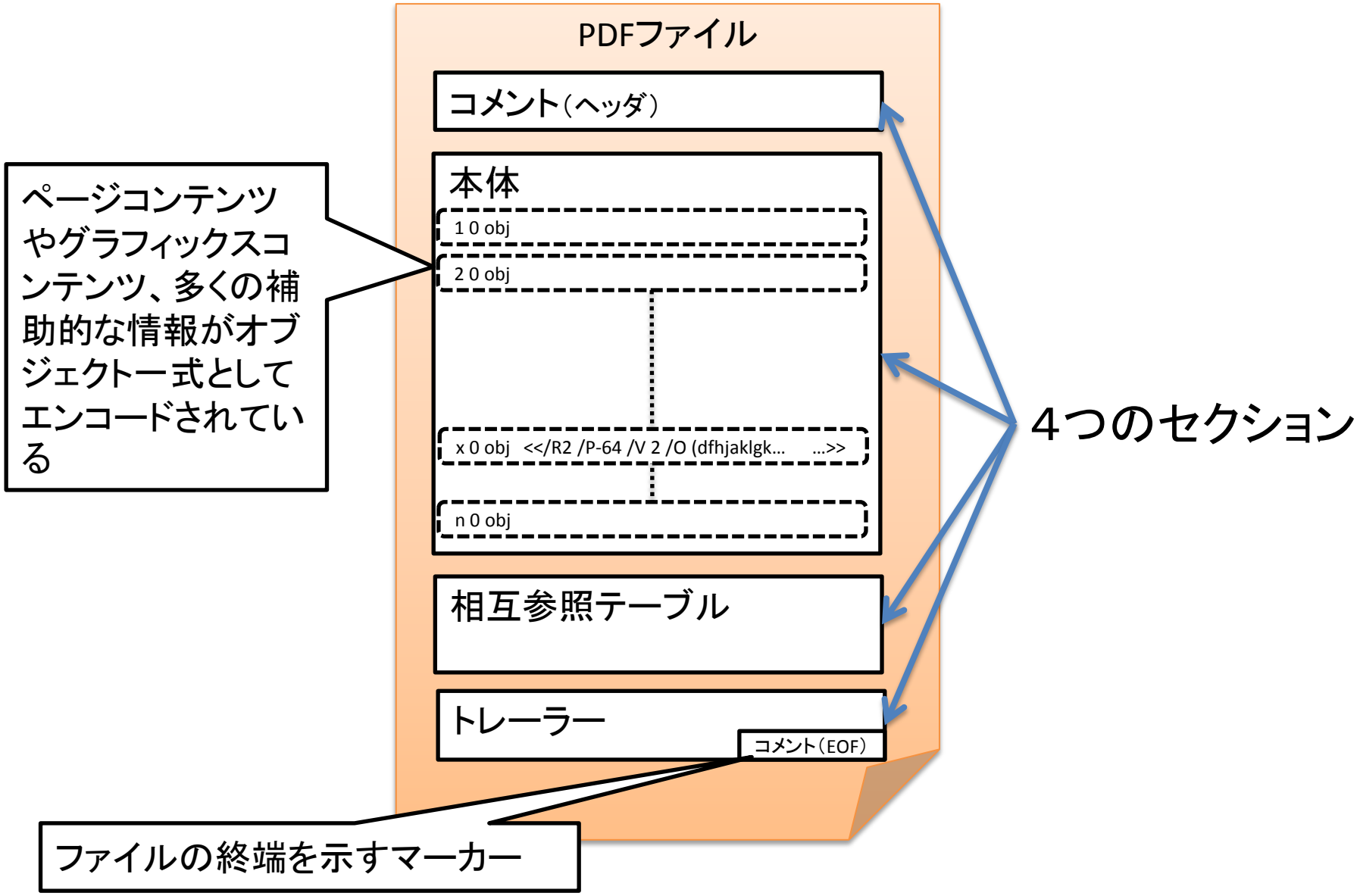
特徴	課題
文書ファイルを実際に閲覧して挙動を解析。	悪性文書ファイルが動作する環境を整備する必要がある

本研究

文書ファイルの仕様はエンコードに依存しないことに着目し悪性文書ファイルの特徴を検知する手法を提案

PDFの基本構造：ファイル構造

PDFは大量のオブジェクト（整数、文字列、バイナリデータ等）の集合体



PDFの基本構造：オブジェクト型

（PDF32000-1:2008 7.3）

基本的なオブジェクト

①整数や実数
②文字列
③名前
④ブーリアン値
⑤nullオブジェクト

複合オブジェクト

⑥配列
⑦辞書

その他

⑧Stream（バイナリデータを格納するためのもの）
⑨間接参照

PDFの基本構造：Streamのエンコード

PDF標準の機能としてStreamにフィルタをかけることができ、そのデコード処理を行う。
主なフィルタは下表のとおり。なお、フィルタは複数がけが可能（PDF32000-1:2008 7.4）

名称	概要
/ASCIIHexDecode	2桁の16進数で表現された文字列から1バイトのデータを復元
/ASCII85Decode	!からzまでの印字可能文字を使用して表現された文字列からデータを復元
/LZWDecode	TIFF画像形式で用いられているLZW圧縮されたデータを展開
/FlateDecode	Zlibライブラリで用いられているFlate圧縮されたデータを展開
/RunLengthDecode	バイト単位の単純なランレングス圧縮されたデータを展開
/CCITTFaxDecode	ファックス機器で使用されているエンコーディング形式のデータを展開
/JBIG2Decode	JBIG2圧縮されたデータを展開
/DCTDecode	JPEGによる不可逆圧縮されたデータを展開
/JPXDecode	JPEG2000による不可逆圧縮されたデータを展開

PDFの基本構造:ドキュメント構造

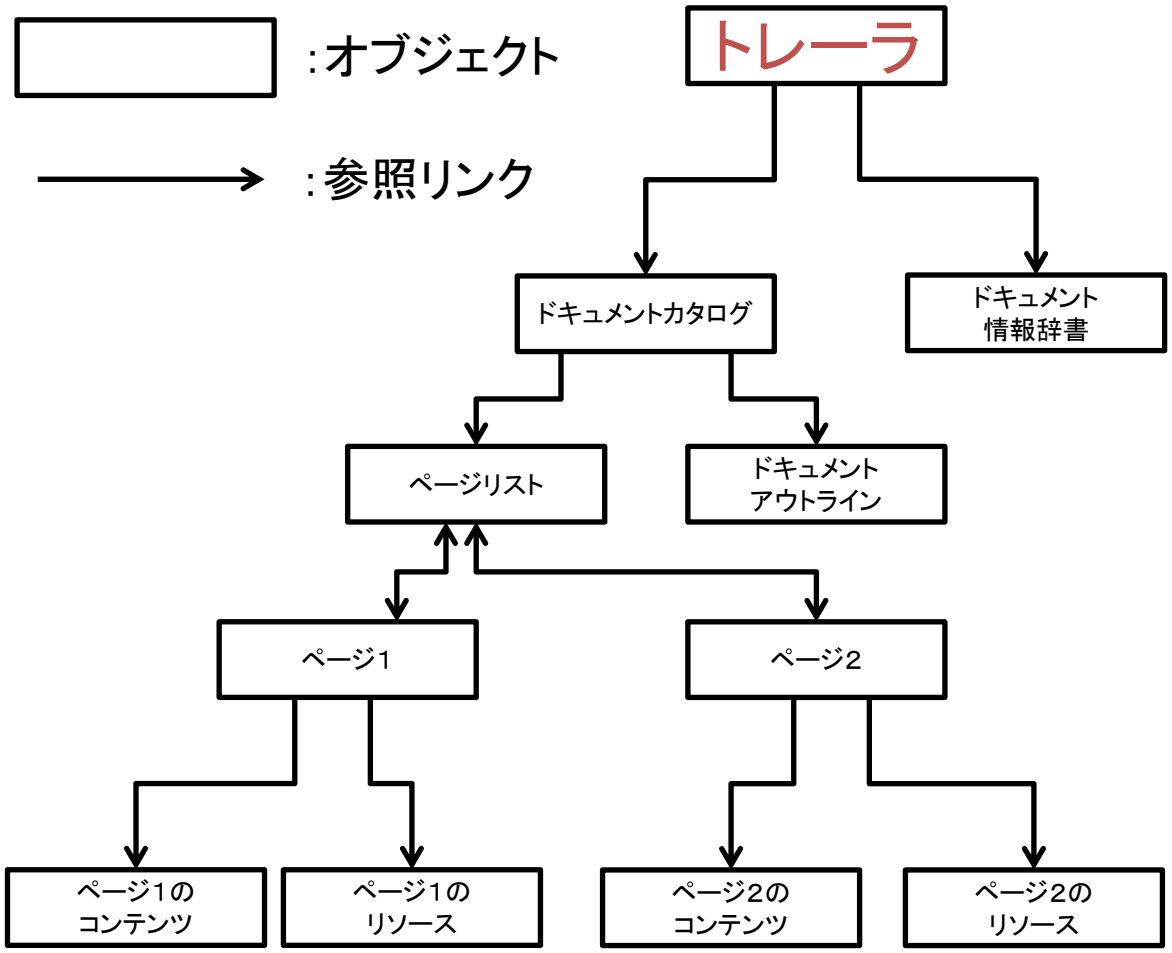


図:2ページからなる一般的なPDFファイルのドキュメント構造

トレーラ辞書オブジェクトからリンクをたどっていくと
すべてのオブジェクトを参照することができる

PDFの基本構造:ドキュメントの暗号化

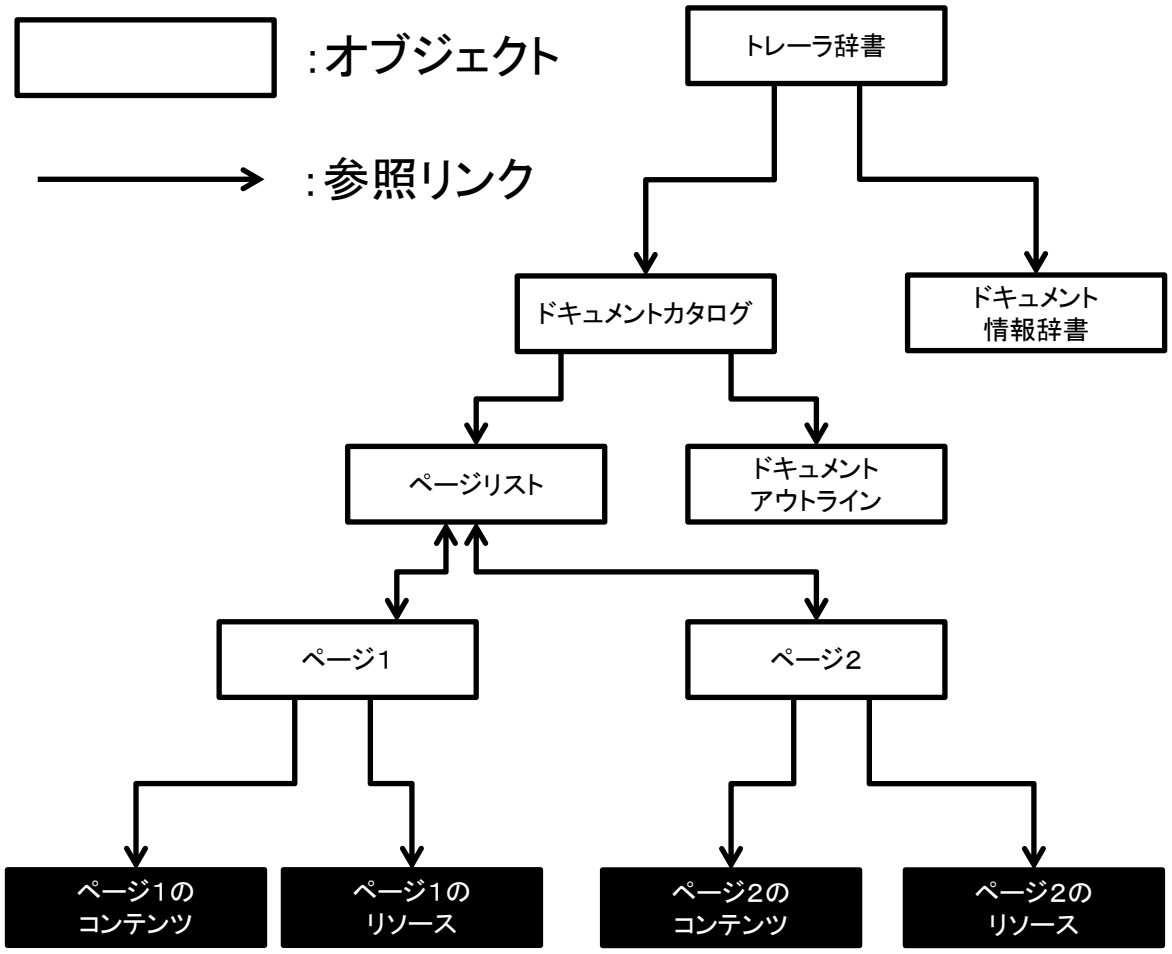
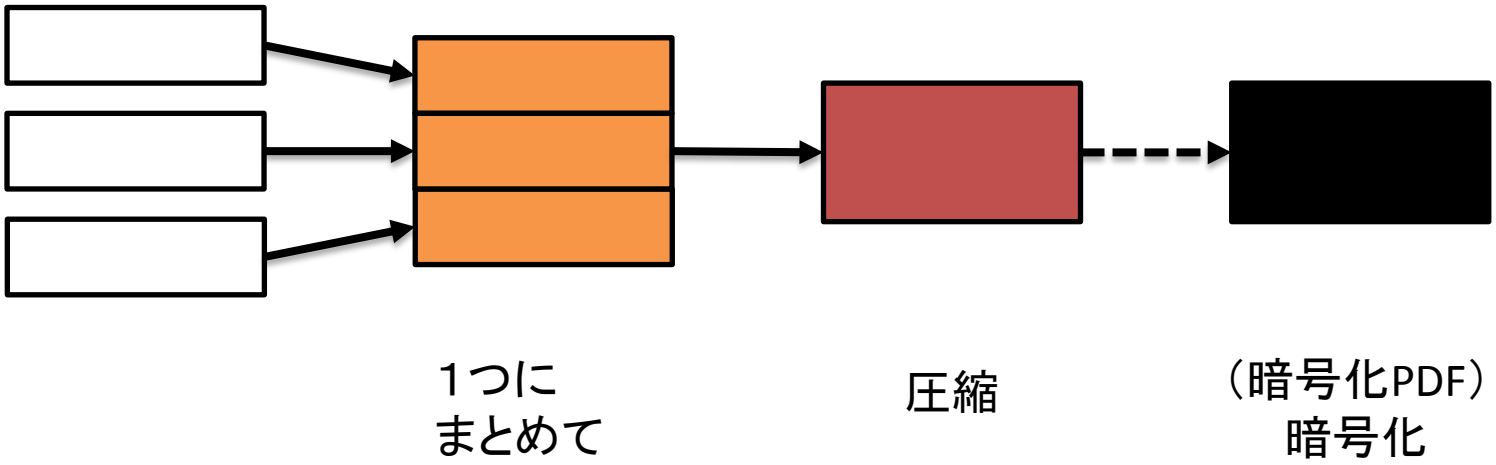


図:暗号化されたPDFファイル

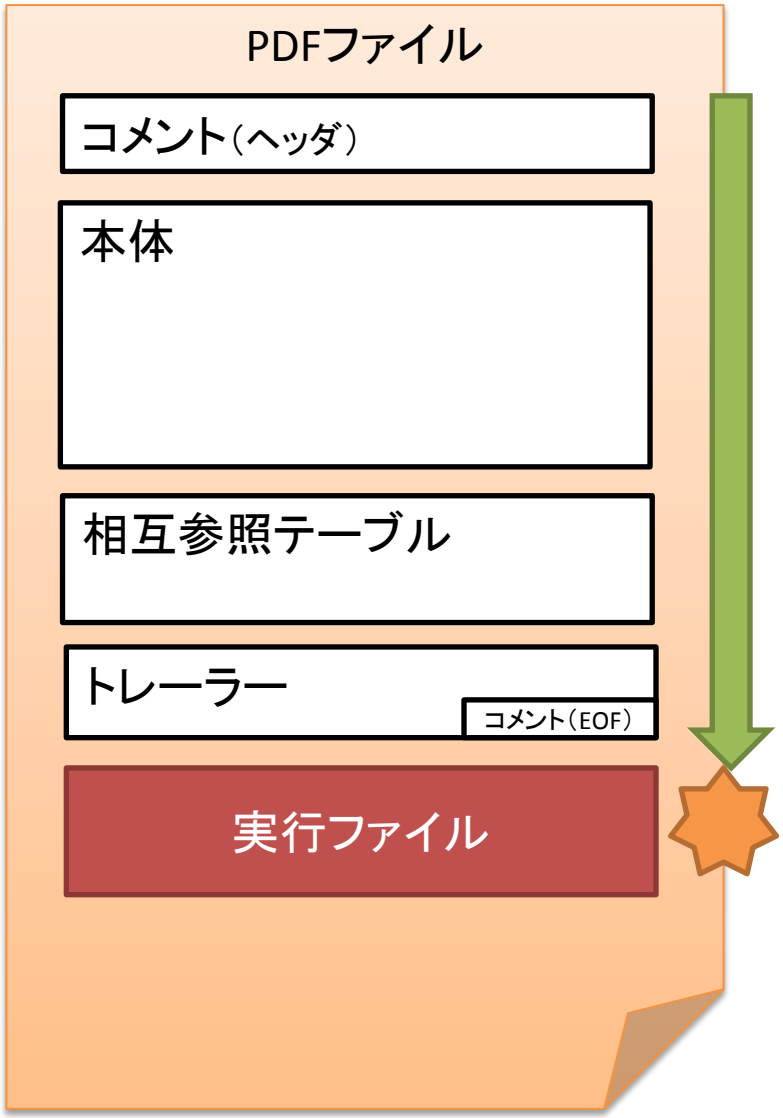
基本的に文字列及びStreamのみが暗号化
復号しなくてもドキュメント構造にアクセスできる(ObjStmに格納されているオブジェクトは除く)

PDFの基本構造：ObjStm（オブジェクトストリーム）

PDF1.5以降で、複数のオブジェクトを単一のStreamに格納しそのStream全体を圧縮することでPDFファイルをさらにコンパクトなものにするというObjStmというものが導入されている
（ PDF32000-1:2008 7.5.7）

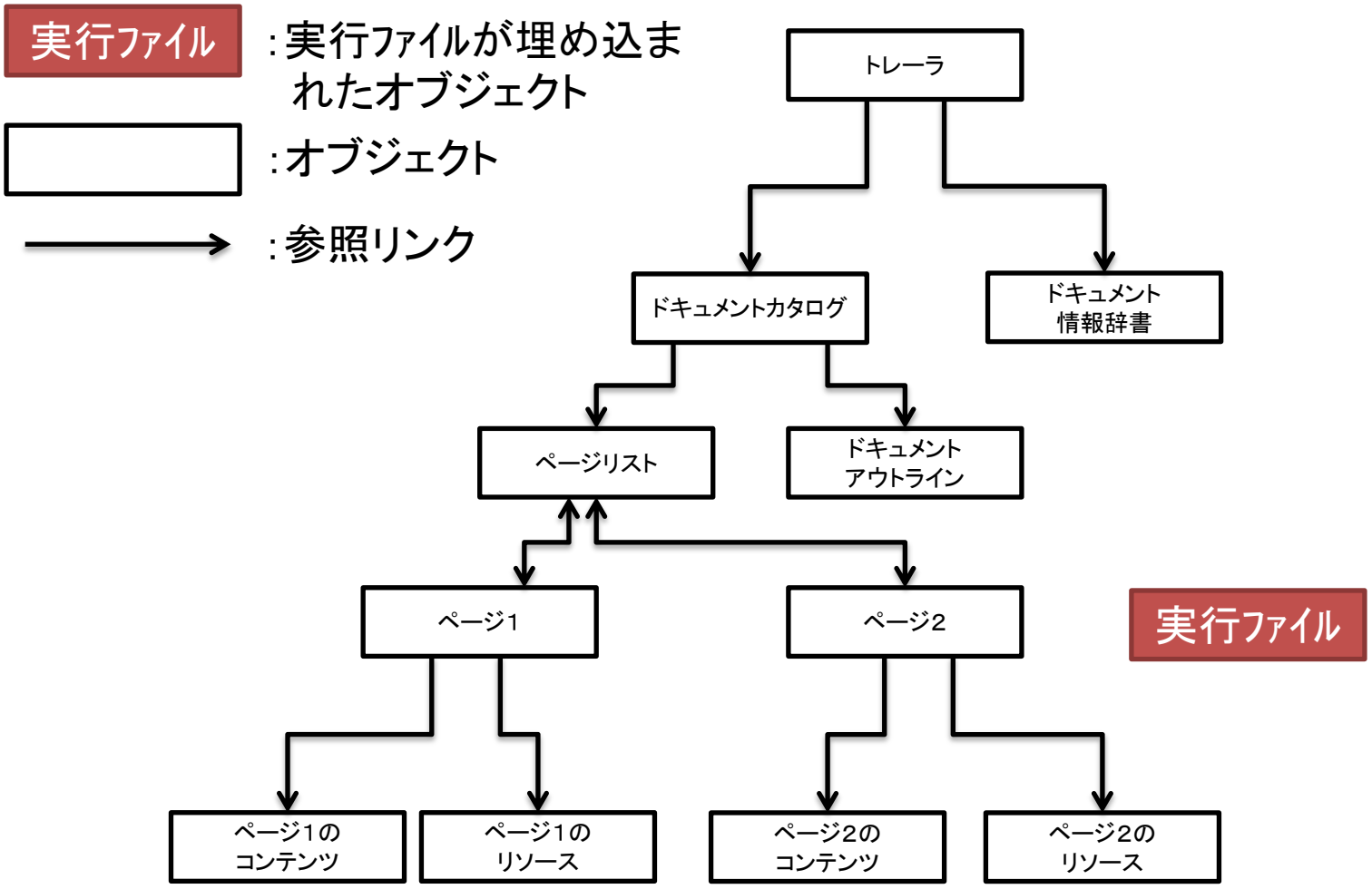


特徴1：分類できないセクション



ファイルの先頭から順番に読み込みどのセクションに該当するか分類しようとすると分類できない部分がある

特徴2：参照されないオブジェクト



図：実行ファイルが埋め込まれたPDFファイルのドキュメント構造

ドキュメント構造を無視して実行ファイルがオブジェクトに埋め込まれると
どこからも参照されていないオブジェクトとなることが多い

特徴3：偽装されたStream

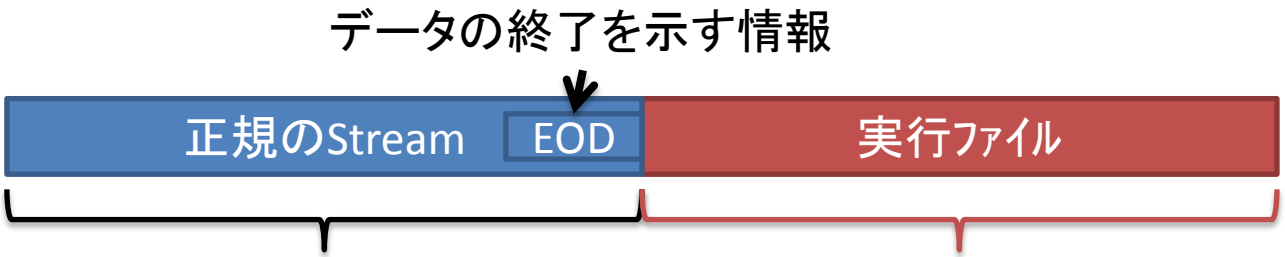
フィルタ偽装

	エントロピー
プレーンテキスト	小
FlateDecode	大
実行ファイル	大

←エントロピーの値が近いフィルタを使用しているように偽装
（デコードしようとすると失敗する）

Stream末端に追加

FlateDecode、DCTDecodeおよびJBIG2Decodeの場合



試験プログラムの実装

検知名	検知内容	対象ファイル
手法1	特徴1：分類できないセクション	・すべてのPDF
手法2	特徴2：参照されないオブジェクト	・平文PDF ・暗号化PDFのうち パスワードが空白 またはObjStmがない もの
手法3	特徴3：偽装されたStream	・平文PDF ・暗号化PDFのうち パスワードが空白 のもの

検体の概要

2009年から2012年までに複数の組織において受信した不審なメールに添付されたもの
実行ファイルが埋め込まれていることが確認されている文書ファイル

マルウェアダンプサイトcontagioでclean（マルウェアではない）とされた文書ファイル
ただし、拡張子とヘッダ情報が一致しないものを除く

悪性PDFファイル		通常のPDFファイル	
検体数	平均容量（KB）	検体数	平均容量（KB）
164	351.2	9,109	101.7

実験環境

CPU	Core i5-3450 3.1GHz
Memory	8.0GB
OS	Windows 7 SP1
Memory(VM)	2.0GB
OS(VM)	Windows XP SP3
Interpreter(VM)	Python 2.7.3

試験プログラムの検知率等

	検知数	検知率
特徴1	81 / 164	49.4%
特徴2	72 / 164	43.9%
特徴3	104 / 164	63.4%
全体	163 / 164	99.4%

平均実行時間：0.69s

	誤検知数	誤検知率
誤検知	19 / 9,109	0.2%

ウイルス対策ソフト等との検知率の比較

	検知数	検知率
試験プログラム	163 / 164	99.4%
T社AV	32 / 164	19.5%
S社AV	16 / 164	9.8%
M社AV	5 / 164	3.0%
T,S,M社AV	39 / 164	23.8%

考察

検知に失敗した原因

試験プログラムが未対応の暗号鍵生成技術（Public-Key Security Handlers）が使用されており、PDFの復号に失敗
ObjStmがあったため、特徴2および特徴3の判定ができなかった

⇩

試験プログラムが当該暗号鍵生成技術に対応すれば検知可能

誤検知の原因

原因は3点

・間接参照されない通常のオブジェクト（14個）
・ファイルの一部が破損しているもの（3個）
・ファイルの途中に不要なデータが付加されているもの（1個）

⇩

オブジェクトのサイズでフィルタリングすることで一部回避可能

試験プログラムの効果

- 実用性

検知率	99.4%	誤検知率	0.2%
平均実行時間	0.69s		

- 検知プログラムの更新頻度が低い

検査対象	更新頻度	
マルウェア	高	1日あたり20万個の新種
エンコード方式	中	マルウェア埋め込みツールの更新頻度に依存
PDFの構造	低	PDFのファイルフォーマットの更新頻度に依存

攻撃者の意志でコントロール可能

攻撃者の意志ではコントロールできない

- パスワードで暗号化されたPDFファイルへの適用

ほとんどの暗号化PDFは復号しなくてもドキュメント構造にアクセス可能
特徴1および特徴2の判定が可能

まとめ

- 実行ファイルが埋め込まれたPDFファイルの構造上の特徴を調査し検知法を考察
- 上記検知法を実装した試験プログラムを作成し、検知率を評価
- 試験プログラムの効果について考察

今後の課題

- 他のフィルタへの対応
- 悪性でないもののPDFのファイルフォーマットに準拠していないPDFファイルへの対応