

Finding Software License Violations Through Binary Code Clone Detection

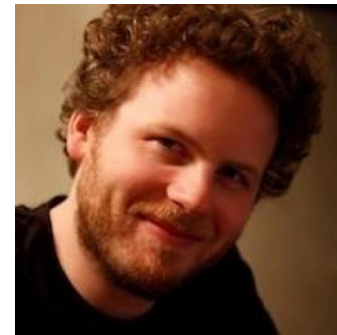


Armijn Hemel

Karl Trygve Kalleberg

Rob Vermaas

Eleco Dolstra



:[:gpl-violations.org](http://www.gpl-violations.org)

:[:kolibriFX](http://www.kolibriFX.com)

:[:Delft University](http://www.delft.nl)

:[:Delft University](http://www.delft.nl)

事前知識

- 組み込みソフトウェアにはよくOSSが利用されている
 - 例:Linux(GPLライセンス)が搭載されている機器



液晶TV



無線ルータ



オーディオプレイヤー

- ソースコードを公開しなければならない

しかし近年、組み込みソフトウェアの
ライセンス違反が増えている

Braviaソースコード公開画面

SONY

Support / Contact Us Global

Source Code Distribution Service

Model/Module

Find

Japan	
Category	Model/Module
Digital TV	KDL-16M1/KDL-20M1 / KDL-20J1/KDL-26J1/KDL-32J1 / KDL-32F1/KDL-40F1/KDL-46F1 / KDL-40V1/KDL-46V1/KDL-52V1 / KDL-32JE1 / KDL-40W1/KDL-46W1
Digital TV	KDL-40ZX1
Digital TV	KDL-19J5,KDL-22J5,KDL-26J5,KDL-32J5,KDL-40V5,KDL-46V5
Digital TV	KDL-32F5/KDL-40F5/KDL-46F5/KDL-40W5/KDL-46W5/KDL-52W5/KDL-46ZX5/KDL-52ZX5
Digital TV	KDL-46HX700/KDL-40HX700/KDL-52EX700/KDL-46EX700/KDL-40EX700/KDL-32EX700/KDL-40EX500/KDL-32EX300/KDL-26EX300/KDL-22EX300/KDL-40NX800/KDL-46LX900/KDL-40LX900/KDL-52HX900/KDL-46HX900/KDL-46HX800/KDL-40HX800/KDL-46NX800/KDL-60LX900/KDL-52LX900/KDL-40EX710/KDL-32EX710
Digital TV	KDL-22BX30H/KDL-26BX30H/KDL-32BX30H
Digital TV	KDL-26EX30R/KDL-32EX30R/KDL-40HX80R/KDL-46HX80R/KDL-55HX80R
Digital TV	KDL-22CX400/KDL-22EX420/KDL-26EX420/KDL-32EX420/KDL-32CX400/KDL-32EX720/KDL-32EX72S/KDL-40EX720/KDL-40EX72S/KDL-46EX720/KDL-46EX72S/KDL-55EX720/KDL-60EX720/KDL-40HX720/KDL-46HX720/KDL-46HX820/KDL-46HX920/KDL-55HX820/KDL-55HX920/KDL-40NX720/KDL-46NX720/KDL-65HX920
Digital TV	KDL-22EX42H/KDL-32EX42H/KDL-40EX52H
Digital TV	KDL-32HX65R/KDL-40HX65R/KDL-46HX65R
Digital TV	KDL-55HX750/KDL-32EX550/KDL-26EX540/KDL-22EX540/KDL-55HX850/KDL-46HX850/KDL-46HX750/KDL-40HX850/KDL-40HX750/KDL-32HX750/KDL-40EX750/KDL-55HX950/KDL-65HX950
Digital TV	KDL-42W802A/KDL-32W600A/KDL-42W650A/KDL-47W802A/KDL-55W802A/KDL-40W900A/KDL-46W900A/KDL-55W900A/KD-65X9200A/KD-55X9200A/KDL-24W600A

<http://oss.sony.net/Products/Linux/TV/category01.html>

組み込みソフトウェアの ライセンス違反

- BusyBox訴訟
 - BusyBox(1996~):GPLv2
 - Linuxのcp,grep,pingなどの万能コマンドを単一の実行ファイルで実行できるため, **容量を大幅に削減できる**
 - Linuxが用いられている組み込みソフトウェアはほとんどBusyBoxをファームウェアとしている
- 2007/9/20 SFLCがMonsoon Multimedia Inc.に訴訟を起こし, 勝訴
- その後, 大企業14社を提訴

バイナリを用いた ライセンス特定が必要な理由

- 組み込みソフトウェア開発者のOSSライセンス違反に対する意識の低さ
 - 組み込まれているので、言わなければ分からないだろう
- オフショア開発などアウトソースによる開発
 - 海外開発者による不適切なOSS利用
 - バイナリだけ受け取る場合もある
 - ソースコードのコメントアウト部分のライセンスが嘘の場合が存在する

そもそも自社コードにOSSが利用されているかどうか不明

バイナリコードを用いた ライセンス特定の問題

- BlackDuck社のProtex
 - 100万以上の固有プロジェクトに関する様々な情報(ソースコード, バイナリ, ライセンス含む)を蓄積した世界最大級のデータベースを利用
- バイナリからソースコードを起こしデータベースと突き合わせる
- 料金:検査対象ファイル50M以下 345万～
- 値段高すぎて使えない!!

目的

- 目的:ライセンス違反を支援するバイナリコード分析ツール(BAT)の作成
- ゴール:バイナリに含まれているOSSの検出
- 3つのアプローチで比較
 - 文字列リテラルを利用したアプローチ
 - 圧縮率を利用したアプローチ
 - バイナリの差分を利用したアプローチ

文字列リテラルを利用したアプローチ

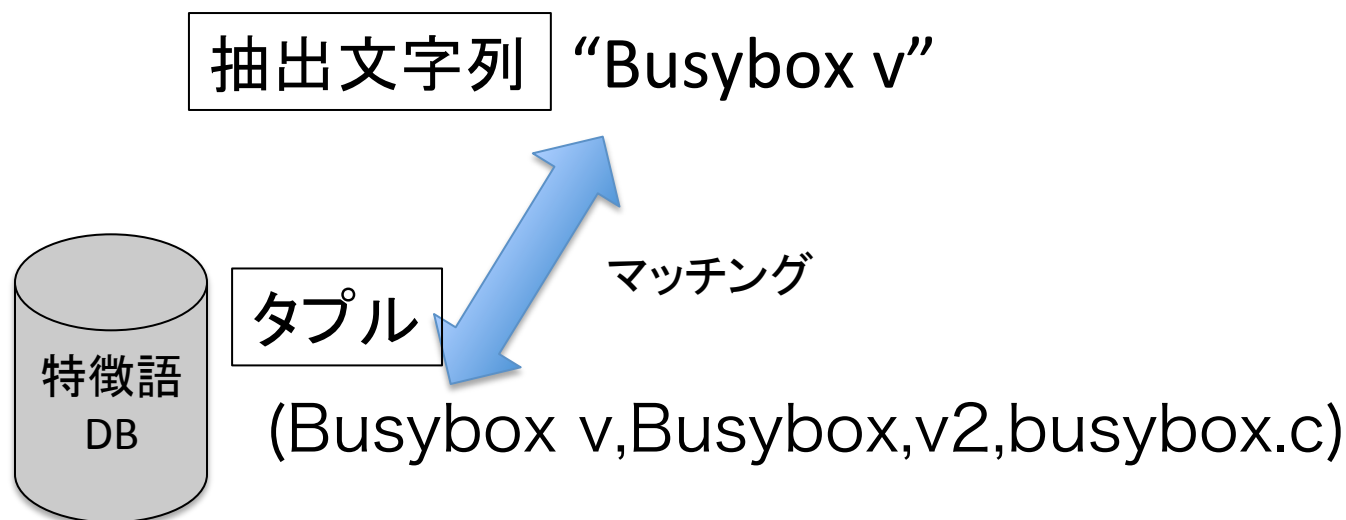
1. バイナリコードを取得
2. OSSに関する特徴語をバイナリから検出する
例:char s[] = “BusyBox v”←これは復元可能
3. 検出結果をRankingMethodにかけて各パッケージごとにOSSの潜在性について分析する

OSSに関する特徴語

- Fedora v5,9,11,14から文字列リテラルを抽出
 - 以下のタプル表現で特徴語DBに保存
(string,packageName,version,sourceFile)
- C,C++のソースコード対象とした23,896のソフトウェア
- 42,238,120の文字列を取得

バイナリからの特徴語検出方法

- GNU string toolsを利用
 - アスキーコードを文字列に変換
 - 非表示文字(0x00-0x1f)の後に続く アスキーコードをスキャンする
- 抽出できた文字列と特徴語DBとマッチさせる



RankingMethod

- OSSのコードクローンと結論づけるためのスコアを算出する

$$\text{スコア計算式} \quad \frac{\text{length}(s)}{\alpha |\text{pkgs}(s)| - 1}$$

閾値:スコアが100以上のものをクローンとする

- 同じパッケージの特徴語がたくさん検出されるほど、高いスコアになる
 - $\text{length}(s)$ =トータルでマッチした文字列の長さ
 - $\text{pkg}(s)$ は s に含まれているpackage P の数

スコア算出例

- SVNのバイナリからOSSを検出

Score	Package	# Unique
114573.95	subversion	2901
61089.06	libxml2	44
45354.69	openssl	2066
43808.14	db (sqlite)	381
6890.76	glibc	147
2013.99	<i>cadaver</i> (neon)	0
392.00	wireshark	6
342.01	<i>php</i> (sqlite)	2
340.94	<i>vtk</i> (sqlite)	0
256.74	zlib	0
253.75	ruby	13
179.23	<i>thunderbird</i> (expat)	0
174.27	Subcommander	2
117.01	apr	0

Table 1: Potential clones in svn ranked by evidence, as detected by scanning for strings

圧縮率を利用したアプローチ

1. 事前にDebianのリポジトリからバイナリ y を抽出
2. 実験用のバイナリファイル x とバイナリ y を C で圧縮する
 - もし次のような関係が成り立てば x は y のクローンであるといえる $|C(xy)| < |C(x)| + |C(y)|$

スコア計算式

$$\text{reuse}_c(x, y) = \frac{|C(x)| + |C(y)| - |C(xy)| - \beta}{|C(y)|}$$

閾値: スコアが0.1以上であればクローンとする

バイナリの差分を利用したアプローチ

1. 事前にDebianのリポジトリからバイナリ y を抽出しておく
2. 検査対象バイナリ x がバイナリ y のクローンを含んでいるかどうかを見る

スコア計算式

$$\text{reuse}_d(x, y) = 1 - \frac{|D(x, y)|}{|D(\epsilon, y)|}$$

閾値:スコアが0.01以下であればクローンとする

評価

- RQ1:文字列リテラルを利用したアプローチはバイナリコードクローン特定に効果的があるか？
- RQ2:圧縮率を利用したアプローチはバイナリコードクローン特定に効果的があるか？
- RQ3:バイナリの差分を利用したアプローチはバイナリコードクローン特定に効果的があるか？

文字列リテラルを利用した アプローチの評価

Binary	<i>tp</i>	<i>fp</i>	<i>fn</i>	Recall	Precision
busybox	6	2	0	1.00	0.75
ffmpeg	6	3	6	0.50	0.67
vim	3	0	0	1.00	1.00
qgit	13	9	8	0.62	0.59
yasm	4	0	0	1.00	1.00
speexenc	2	0	1	0.67	1.00
pdftops	5	0	0	1.00	1.00
gd2topng	4	0	2	0.67	1.00
w3m	4	0	1	0.80	1.00
openssh	4	4	0	1.00	0.50
Average				0.83	0.85

圧縮率を利用した アプローチの評価

Binary	tp	fp	fn	Recall	Precision
busybox	5	0	0	1.00	1.00
ffmpeg	9	0	7	0.56	1.00
vim	4	0	2	0.67	1.00
qgit	23	9	6	0.79	0.72
yasm	2	0	2	0.50	1.00
speexenc	5	0	1	0.83	1.00
pdftops	4	0	2	0.67	1.00
gd2topng	6	0	3	0.67	1.00
w3m	6	9	3	0.67	0.40
openssh	6	0	1	0.86	1.00
Average				0.72	0.91

バイナリの差分を利用した アプローチの評価

Binary	<i>tp</i>	<i>fp</i>	<i>fn</i>	Recall	Precision
busybox	4	0	0	1.00	1.00
ffmpeg	8	0	7	0.53	1.00
vim	3	0	2	0.60	1.00
qgit	15	9	13	0.54	0.62
yasm	1	0	2	0.33	1.00
speexenc	4	0	1	0.80	1.00
pdftops	3	0	1	0.75	1.00
gd2topng	5	0	3	0.62	1.00
w3m	3	10	4	0.43	0.23
openssh	4	0	1	0.80	1.00
Average				0.64	0.89

サードパーティ製バイナリでの評価

文字列リテラルを利用したアプローチを試した結果

Binary	Type	Size (MiB)	<i>tp</i>	<i>fp</i>	Preci -sion
Vodafone Webby	Firmware	29	42	46	0.48
Asus WL500G	Firmware	2	26	12	0.68
Spotify	Core dump	344	27	61	0.31

Precisionが低い理由

- 複数のパッケージに存在する単語が多数存在

スコアが上位のOSSパッケージでは高いPrecision値
Vodafoneの上位18位のOSSは全て検出できている

まとめ

- 以下を利用したアプローチはライセンス違反検出支援に効果があることを示した
 - バイナリの文字列リテラルの抽出 prec:85%
 - バイナリの圧縮率 prec:91%
 - バイナリの差分 prec:89%
- サードパーティ製のバイナリにも, そこそこ有用であることを示した