

Security Vulnerabilities in Categories of Clones and Non-Cloned Code: An Empirical Study

Author:
Md Rakibul Islam
Minhaz F. Zibran
Aayush Nagpal

ESEM '17 Proceedings of the
11th ACM/IEEE International
Symposium on Empirical
Software Engineering and
MeasurementPages 20-29

目次

- 背景, 目的
- 分析手法
- 分析結果
- まとめ, 所感

背景, 目的

背景

- 脆弱性を含むコードをコピーすると脆弱性のある部分が広がり、システム全体の安全性が低下する
- クローンの内部に含まれるセキュリティの脆弱性の研究がなされていない
- クローンに含まれる脆弱性の特徴を知り、どのような対処が必要が知りたい

目的, アプローチ

- 目的

- ▶ コードクローンのに含まれるセキュリティに関する脆弱性を理解する

- アプローチ

- ▶ 複数のタイプのクローンとクローンでないコードとの間で脆弱性の比較を行う

分析手法

リサーチクエスト

- RQ1: 脆弱性を含む数が多いのはクローンか非クローンか
- RQ2: 特定のタイプのクローンは多くの脆弱性を含むか
- RQ3: クローンの方がよりリスクが大きい脆弱性を含むのか
- RQ4: 特定のタイプのクローンはよりリスクが大きい脆弱性を含むか
- RQ5: クローンにより頻繁に現れる脆弱性を区別することが出来るか

データセット

- すべてC言語で書かれたOSS

- ネットワーク, コミュニケーション, エディタなど様々なドメインのプロジェクトが集められている

TABLE II
SUBJECT SYSTEMS AND THEIR LOC IN CLONED AND NON-CLONED CODE

Subject System	# of LOC written in C only		
	Non-clone	Clone	Total
Asn1c	40,487	5,488	45,975
Atlas	369,944	116,586	486,530
Clamav	337,019	40,371	377,390
Claws	239,784	32,868	272,652
Conky	27,759	14,494	42,253
Courier	107,223	13,105	120,328
Emacs	314,521	17,092	331,613
Ettercap	36,268	5,182	41,450
Ffdshow	832,342	46,367	878,709
Freeddiag	15,225	1,380	16,605
Freedroid	60,828	3,957	64,785
Gedit	42,112	4,331	46,443
Glimmer	29,536	4,923	34,459
Gnuplot	86,851	6,712	93,563
Gretl	302,777	36,499	339,276
Grisbi	99,205	18,479	117,684
Ipsec-tools	60,728	10,222	70,950
Modsecurity	26,332	7,232	33,564
Nedit	85,001	9,506	94,507
Net-snmp	231,422	42,725	274,147
Ocf-linux	45,761	8,743	54,504
Openkim	47,298	18,480	65,778
Opensc	119,646	11,823	131,469
Putty	78,918	11,322	90,240
Razorback	36,403	8,823	45,226
Sdce	3,477,010	4,122	3,481,132
Tboot	21,942	6,115	28,057
Tcl8	326,217	37,693	363,910
Tcpreplay	43,191	4,740	47,931
Trousers	56,673	17,599	74,272
Vi	21,924	734	22,658
Vim	314,480	5,752	320,232
XSupplicant	78,441	24,028	102,469
Zabbix	107,475	18,156	125,631

分析手法の概略図

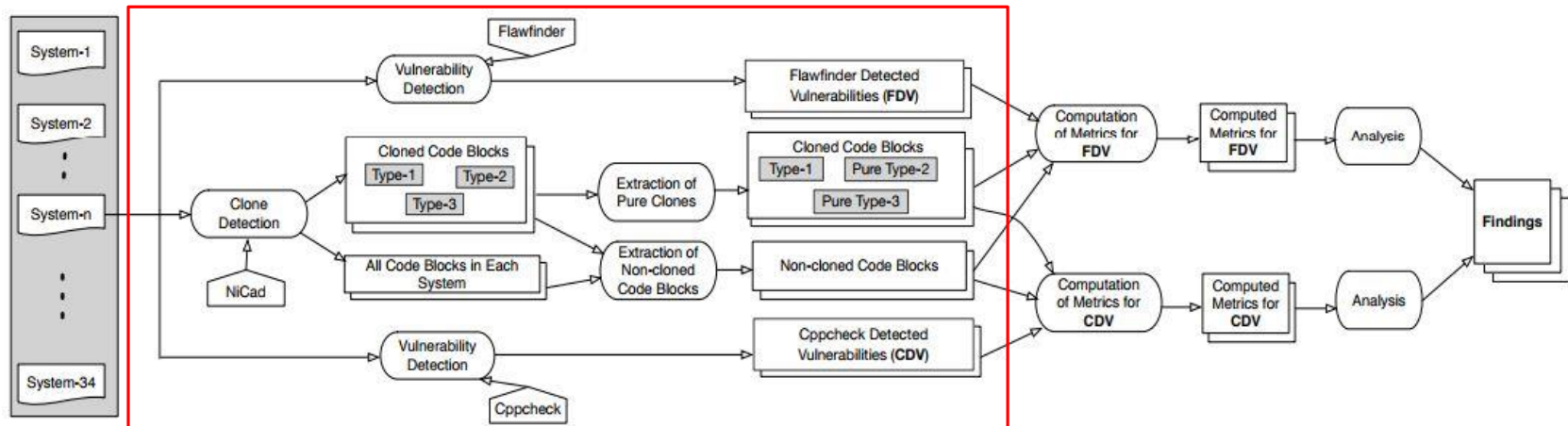


Fig. 1. Procedural Steps of the Empirical Study

クローンと脆弱性の検出

- クローンの検出は3つのタイプのクローンを検出できるNiCad [39]を用いる
- 脆弱性の検出は以下の2つを用いる
 - Flawfinder[3]
 - 再現率が高い
 - 脆弱性のリスクの重大さを5段階に分けることが出来る
 - 他の研究でよく用いられる
 - CppCheck[2]
 - False positiveが出来るだけ小さくなるように作られている
 - 多くの種類の脆弱性を検出することが出来る
- それぞれの検出結果をFDV(Flawfinder Detected Vulnerabilities), CDV(Cppcheck Detected Vulnerabilities)と略す

[39] C. Roy and J. Cordy. NICAD: Accurate detection of near-miss intentional clones using flexible pretty-printing and code normalization. In ICPC, pages 172–181, 2008.

[3] Flawfinder - C/C++ Source Code Analyzer. <http://www.dwheeler.com/flawfinder/>, July 2017.

[2] Cppcheck - A tool for static C/C++ code analysis. <http://cppcheck.sourceforge.net>, July 2017.

脆弱性の検出結果

TABLE V
DETECTED SECURITY VULNERABILITIES AND THEIR SEVERITY IN
CLONED AND NON-CLONED CODE

Subject System	# of FDV		# of CDV		Severity score	
	NC	C	NC	C	NC	C
Asn1c	119	10	183	18	309	70
Atlas	1,258	1,607	2,307	3,240	4,429	4,246
Clamav	1,151	166	6,379	1,901	2,689	345
Claws	429	35	1,496	138	1,148	108
Conky	310	69	10	41	850	200
Courier	2,569	270	1,152	67	7,479	900
Emacs	1,071	99	666	108	2,990	281
Ettcap	419	78	235	23	982	177
Ffdshow	1,306	158	10,042	424	1,457	330
Freeddiag	416	54	104	11	1,521	202
Freedroid	429	26	180	0	1,181	88
Gedit	21	9	276	7	51	18
Glimmer	151	19	326	214	450	64
Gnuplot	717	67	1,322	87	2,172	208
Gretl	2,850	359	2,992	249	8,847	1,116
Grisbi	54	40	293	27	177	82
Ipssec-tools	447	100	790	134	960	210
Modsecurity	163	22	193	40	308	44
Nedit	592	62	624	56	1,884	216
Net-snmp	1,715	352	2,002	177	4,071	763
Ocf-linux	153	17	379	61	245	134
Opendkim	201	40	296	98	383	146
Opensc	1,150	141	735	30	2,347	404
Putty	523	108	593	59	1,217	392
Razorback	104	31	359	39	261	94
Sdcc	26	9	291	38	182	32
Tboot	105	59	80	225	234	140
Tcl8	1,111	218	3,592	463	2,552	628
Tcpplay	421	87	232	10	1,252	228
Trousers	263	110	460	34	560	224
Vi	144	4	403	16	417	52
Vim	805	22	2,336	43	2,264	70
XSupplicant	838	246	2,593	505	1,822	548
Zabbix	512	82	566	24	992	278

Here, NC = non-cloned code and C = cloned code

用いる統計手法

- Mann-Whitney-Wilcoxon検定

- クローンと非クローンとで有意差があるかどうかを調べるために用いる

- Kruskal-Wallis検定

- 3つあるクローンタイプの間に有意差があるかどうかを調べるために用いる

- 有意水準 $\alpha = 0.05$

分析手法の概略図

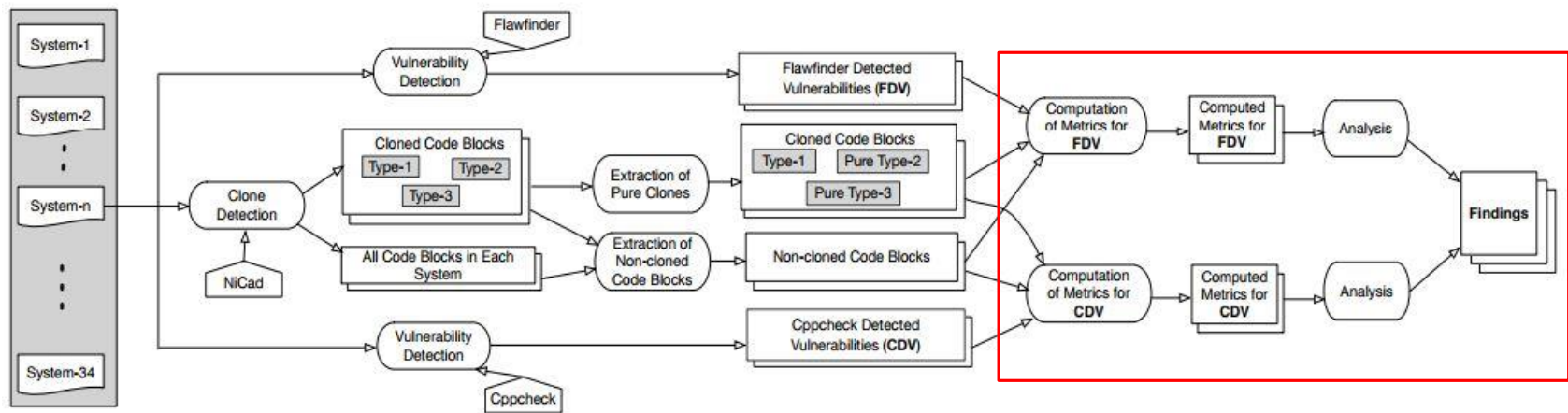


Fig. 1. Procedural Steps of the Empirical Study

検出結果からメトリクスを算出

- 1000行ごとの脆弱性の密度を計算
 - $\frac{\text{脆弱性の数}}{\text{行数}} \times 1000$
 - RQ1, 2で使用
- 1000行ごとの脆弱性のリスクの高さを計算
 - $\frac{\text{脆弱性のリスクのランクの総和}}{\text{行数}} \times 1000$
 - RQ3, 4で使用
- 1000行ごとのそれぞれの脆弱性のカテゴリの密度を計算
 - $\frac{\text{あるカテゴリの脆弱性の数}}{\text{行数}} \times 1000$
 - RQ5で使用

分析結果

リサーチクエスト

- RQ1: 脆弱性を含む数が多いのはクローンか非クローンか
- RQ2: 特定のタイプのクローンは多くの脆弱性を含むか
- RQ3: クローンの方がよりリスクが大きい脆弱性を含むのか
- RQ4: 特定のタイプのクローンはよりリスクが大きい脆弱性を含むか
- RQ5: クローンにより頻繁に現れる脆弱性を区別することが出来るか

RQ1の分析結果

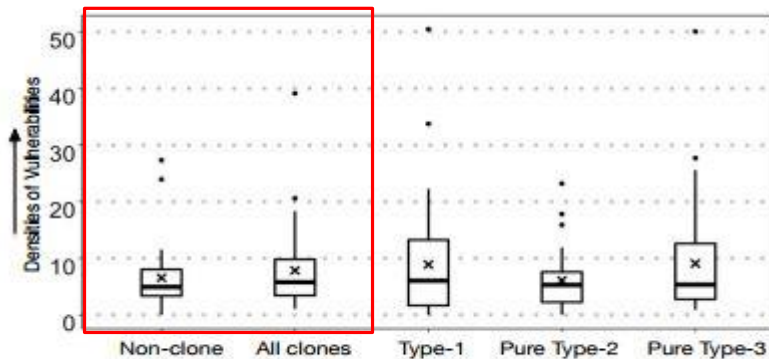


Fig. 5. Densities of FDV w.r.t. LOC

$$p = 0.20 > \alpha$$

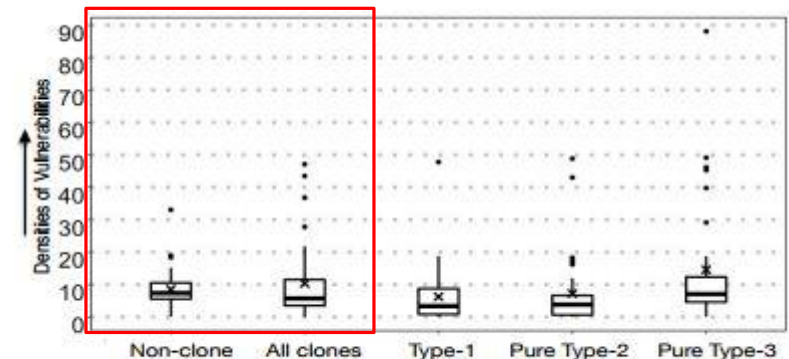


Fig. 7. Densities of CDV w.r.t. LOC

$$p = 0.42 > \alpha$$

- どちらも有意差がなく、クローンと非クローンとの間に脆弱性の多さの差がないことが分かる

リサーチクエスチョン

- RQ1: 脆弱性を含む数が多いのはクローンか非クローンか
- RQ2: 特定のタイプのクローンは多くの脆弱性を含むか
- RQ3: クローンの方がよりリスクが大きい脆弱性を含むのか
- RQ4: 特定のタイプのクローンはよりリスクが大きい脆弱性を含むか
- RQ5: クローンにより頻繁に現れる脆弱性を区別することが出来るか

RQ2の分析結果

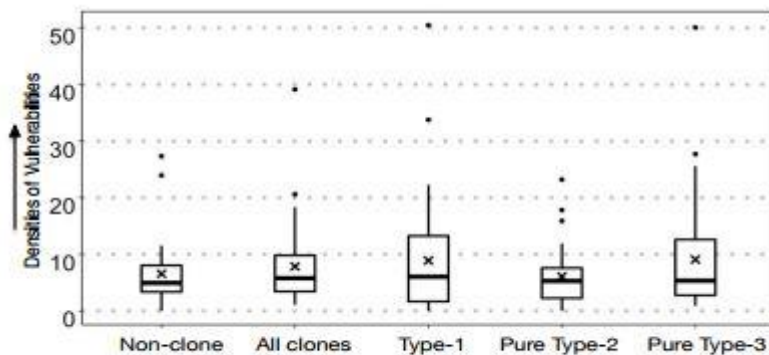


Fig. 5. Densities of FDV w.r.t. LOC

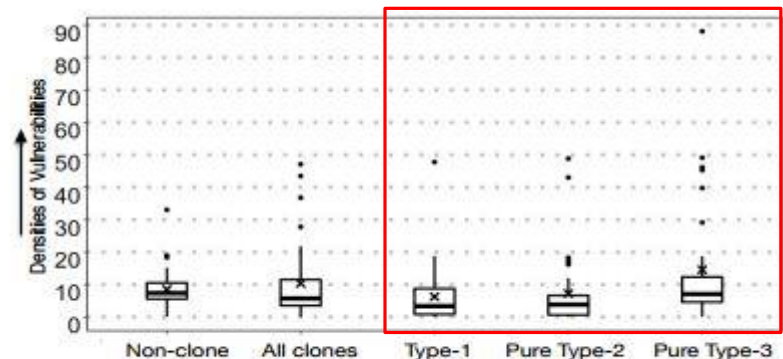


Fig. 7. Densities of CDV w.r.t. LOC

$$p = 0.008 < \alpha$$

- CDVの密度についてはクローン間で差があり、特にタイプ3のクローンが脆弱性のある部分が多いことがわかる

リサーチクエスト

- RQ1: 脆弱性を含む数が多いのはクローンか非クローンか
- RQ2: 特定のタイプのクローンは多くの脆弱性を含むか
- RQ3: クローンの方がよりリスクが大きい脆弱性を含むのか
- RQ4: 特定のタイプのクローンはよりリスクが大きい脆弱性を含むか
- RQ5: クローンにより頻繁に現れる脆弱性を区別することが出来るか

RQ3の分析結果

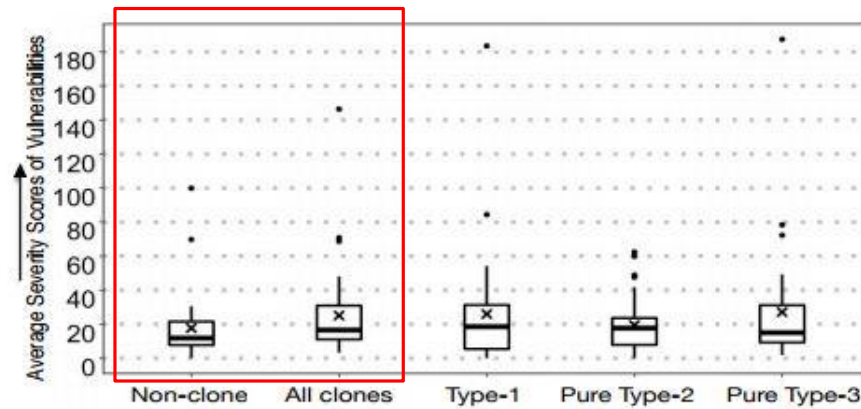


Fig. 9. Risk Severity Scores per KLOC in Cloned and Non-cloned Code

$$p = 0.0494 < \alpha$$

- クローンと非クローンの間の脆弱性のリスクの重大さに有意差があることがわかり、クローンのほうが重大なリスクの脆弱性を含みやすいことがわかる

リサーチクエスト

- RQ1: 脆弱性を含む数が多いのはクローンか非クローンか
- RQ2: 特定のタイプのクローンは多くの脆弱性を含むか
- RQ3: クローンの方がよりリスクが大きい脆弱性を含むのか
- RQ4: 特定のタイプのクローンはよりリスクが大きい脆弱性を含むか
- RQ5: クローンにより頻繁に現れる脆弱性を区別することが出来るか

RQ4の分析結果

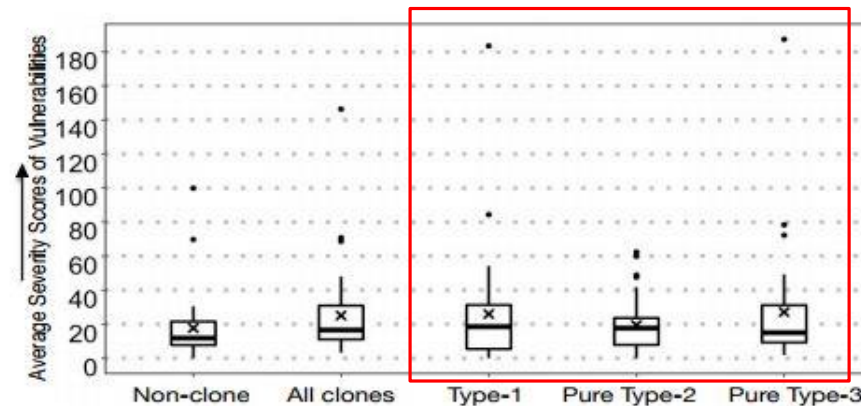


Fig. 9. Risk Severity Scores per KLOC in Cloned and Non-cloned Code

$$p = 0.5901 > \alpha$$

- クローンタイプの間で脆弱性のリスクの重大さに有意差がないことが分かり、特定のクローンタイプがリスクの高い脆弱性を含むことはないことが分かる

リサーチクエスチョン

- RQ1: 脆弱性を含む数が多いのはクローンか非クローンか
- RQ2: 特定のタイプのクローンは多くの脆弱性を含むか
- RQ3: クローンの方がよりリスクが大きい脆弱性を含むのか
- RQ4: 特定のタイプのクローンはよりリスクが大きい脆弱性を含むか
- RQ5: クローンにより頻繁に現れる脆弱性を区別することが出来るか

RQ5の分析結果 (FDV)

- FDVが多く検出したトップ6の脆弱性のカテゴリについてクローンと非クローンとの間に個数の有意差があるかを見る
- クローンと非クローンとの間で有意差があるのはCWE-78とCWE-807の2つ
- CWE-78はクローンで頻繁に現れ, CWE-807は非クローンで頻繁に現れることがわかる

TABLE VI
MWW TESTS OVER DENSITIES OF TOP SIX CWE CATEGORIES OF *FDV* PER KLOC IN CLONED AND NON-CLONED CODE

Categories of <i>FDV</i>	<i>P</i> -Value	Significant?	Cliff's delta <i>d</i>
CWE-78	0.0233	Yes ($p < \alpha$)	0.384 (medium)
CWE-120	0.3264	No ($p > \alpha$)	not applicable
CWE-134	0.2809	No ($p > \alpha$)	not applicable
CWE-190	0.4681	No ($p > \alpha$)	not applicable
CWE-362	0.4051	No ($p > \alpha$)	not applicable
CWE-807	0.0012	Yes ($p < \alpha$)	0.379 (medium)

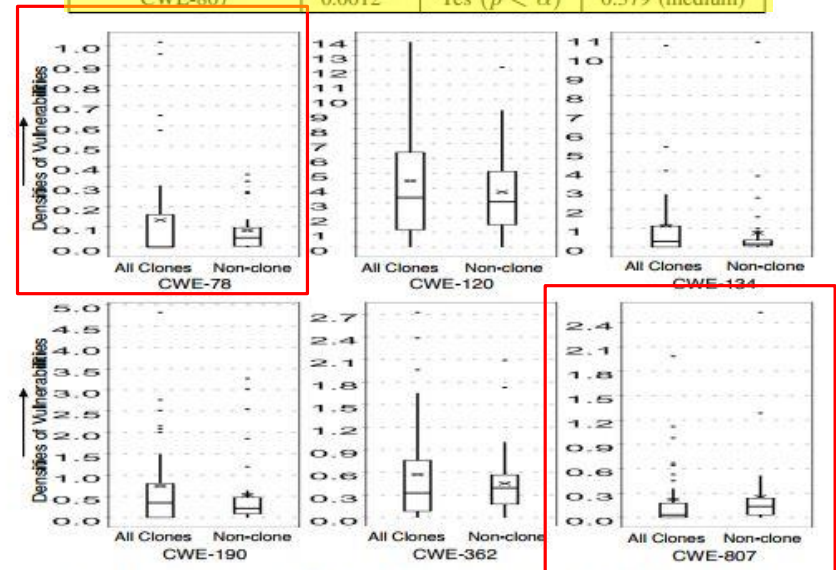


Fig. 10. Densities of top six CWE categories of *FDV* per KLOC

RQ5の分析結果 (CDV)

- CDVが多く検出したトップ5の脆弱性のカテゴリについてクローンと非クローンとの間に個数の有意差があるかを見る

- クローンと非クローンとの間で有意差があるのはCWE-561とCWE-686の2つ

- どちらも非クローンで頻繁に現れることがわかる

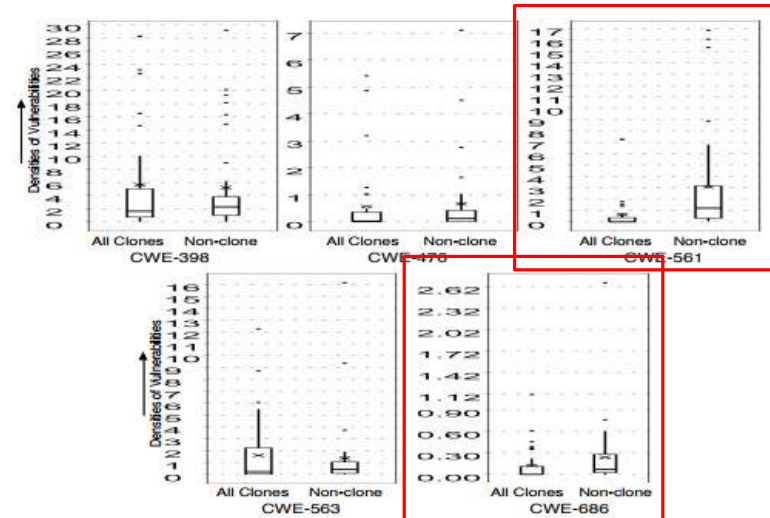


Fig. 11. Densities of top five CWE categories of *CDV* per KLOC

TABLE VII
MWW TESTS OVER DENSITIES OF TOP FIVE CWE CATEGORIES OF *CDV*
PER KLOC IN CLONED AND NON-CLONED CODE

Categories of <i>CDV</i>	<i>P</i> -Value	Significant?	Cliff's <i>delta d</i>
CWE-398	0.2980	No ($p > \alpha$)	not applicable
CWE-476	0.0630	No ($p > \alpha$)	not applicable
CWE-561	0.0321	Yes ($p < \alpha$)	0.692 (large)
CWE-563	0.1538	No ($p > \alpha$)	not applicable
CWE-686	0.0012	Yes ($p < \alpha$)	0.372 (medium)

カテゴリの内容

CWE	内容
CWE-78	OSコマンドインジェクション
CWE-120	バッファオーバーフロー
CWE-134	外部制御形式の文字列の使用
CWE-180	整数オーバーフロー
CWE-362	競合状態
CWE-807	セキュリティ上不適切な入力

FDVのトップ6カテゴリ

CWE	内容
CWE-398	コードの質
CWE-476	ヌルポ参照
CWE-561	実行されないコード
CWE-563	使わない変数の評価
CWE-686	不正な引数による関数呼び出し

CDVのトップ5カテゴリ

まとめ, 所感

まとめ

- クローンと非クローン間, クローンタイプ間でセキュリティの脆弱性の比較を行った
- クローンと非クローンの間で脆弱性の多さに有意な差はなかった
- クローンのほうが非クローンよりリスクの高い脆弱性を含みやすいことがわかった
- クローンのほうが頻繁に現れる脆弱性はOSコマンドインジェクションだけであることがわかった

所感

- プロジェクトやドメインでどのような違いが出るのか気になる
- リスクが大きい脆弱性がクローンに含まれやすいという発見は、コピーを注意深くさせるものであると思った
- OSコマンドインジェクションがクローンに含まれやすいのはなぜか気になった