coverity

LAMP スタック:
品質およびセキュリティ

Rich Cerruto
日本アジア担当ディレクター

Coverity のベーシックなアプローチ

- 静的ソース コード解析

検出できる問題

- セキュリティ上の脆弱性
- システムとプロセスのクラッシュ
- 無限ループ
- パフォーマンスの劣化
- サービス拒否
- 権限昇格
- データ、メモリ、およびファイルの破損
- 予期しない動作
- 並列処理の問題

アプローチ

- コードを実行しない
- テスト ケースを作成しない
- すべてのパスを解析する

オープン ソース + Coverity: 静的解析技術の改善

coverity

飛躍的な研究所技術

1999

Stanford

- メタレベル コンパイル チェッカー ("スタンフォード チェッカー")
- Linux における 2000 以上のバグの検出

Linux でのソース コード静的解析

2001

- 初期バージョンの Linux における数百のバグの公表
- Linux コミュニティによるほとんどの欠陥の修正

Coverity の設立 -- Linux での強化

2002

coverity

- 当社の技術を求める要望が殺到
- 継続する Linux 作業:2000 以上のバグの検出
- 無料サービスを提供する Linuxbugs サイトの作成

会社の成長と拡大

2003-06

- 150 以上の顧客 -- Juniper、Synopsys、Symantec、nVidia、PalmOne、およびその他多数
- 米国国土安全保障省の研究 -- 次世代解析技術の開発とオープン ソースへの適用
- scan.coverity.com、FreeBSD、Mozilla によるオープン ソース関与の拡大

トピック

- 32 のオープン ソース パッケージの品質とセキュリティに関する Coverity の評価方法
- LAMP スタックおよび研究の対象となった他のパッケージとの比較
- この研究が開発コミュニティに公表された後の反応

サンプル バグ No. 1: リソース リーク

- Perl: perl/ext/Storable/Storable.xs
- scan.coverity.com の開発者が確認済み

Event **alloc_fn**: Called allocation function "Perl_safesysmalloc" [model]

Event **var_assign**: Assigned variable "classname" to storage returned from "Perl_safesysmalloc"

Also see events: [var_assign][leaked_storage]

```
4157                            New(10003, classname, len+1, char);
4158                    }
4159
```

Event **leaked_storage**: Returned without freeing storage "classname"

Also see events: [alloc_fn][var_assign]

At conditional (1): "(cxt)->fio == 0" taking true path

At conditional (2): "(((cxt)->membuf).aptr + len) <= ((cxt)->membuf).aend" taking false path

```
4160                    READ(classname, len);
```

サンプル バグ No. 2: 配列のオーバーラン

coverity

- php-src/ext/standard/ftp_fopen_wrapper.c
- off by 1 (1 つ違い) エラー

Event **assignment**: Assigning "4096" to "tmp_len"

Also see events: [overrun-local]

At conditional (6): "4096 < (basename_len - 1)" taking true path

```
612             tmp_len = MIN(sizeof(ent->d_name), basename_len - 1);
613             memcpy(ent->d_name, basename, tmp_len);
```

Event **overrun-local**: Overrun of static array "(ent)->d_name" of size 4096 at position 4096 with index variable "tmp_len"

Also see events: [assignment]

```
614             ent->d_name[tmp_len] = '\0';
```

Coverity のアプローチ: scan.coverity.com

- オープン ソース パッケージを自動的にスキャンするためのインフラストラクチャ
  - 毎晩スキャン
  - Coverity *Prevent 2.4.4* の使用
- 結果を scan.coverity.com に自動的に送信
- 始動
  - 2006 年 3 月 6 日
  - 32 パッケージ
  - 今後さらにパッケージを追加

パッケージ一覧の概要

coverity

Amanda	Gnome	OpenSSL	Snort
Apache	Icecast	OpenVPN	SQLite
Ethereal	Inetutils	Perl	Squid
Firebird	Linux	PHP	TCL
Firefox	Mplayer	PostgreSQL	WxWidgets
FreeBSD	MySQL	ProFTPD	X
GAIM	NetSNMP	Python	Xine
Gcc	OpenLDAP	Samba	XMMS

最初の実行結果

- 比較のための主な評価基準として欠陥密度を選択
- 発表する結果:
  - LAMP スタックの “L” および “P” 部分の詳細データ
  - パッケージ全体の傾向
  - LAMP スタック全体の比較データ

生の数字 – “L” および “P”

coverity

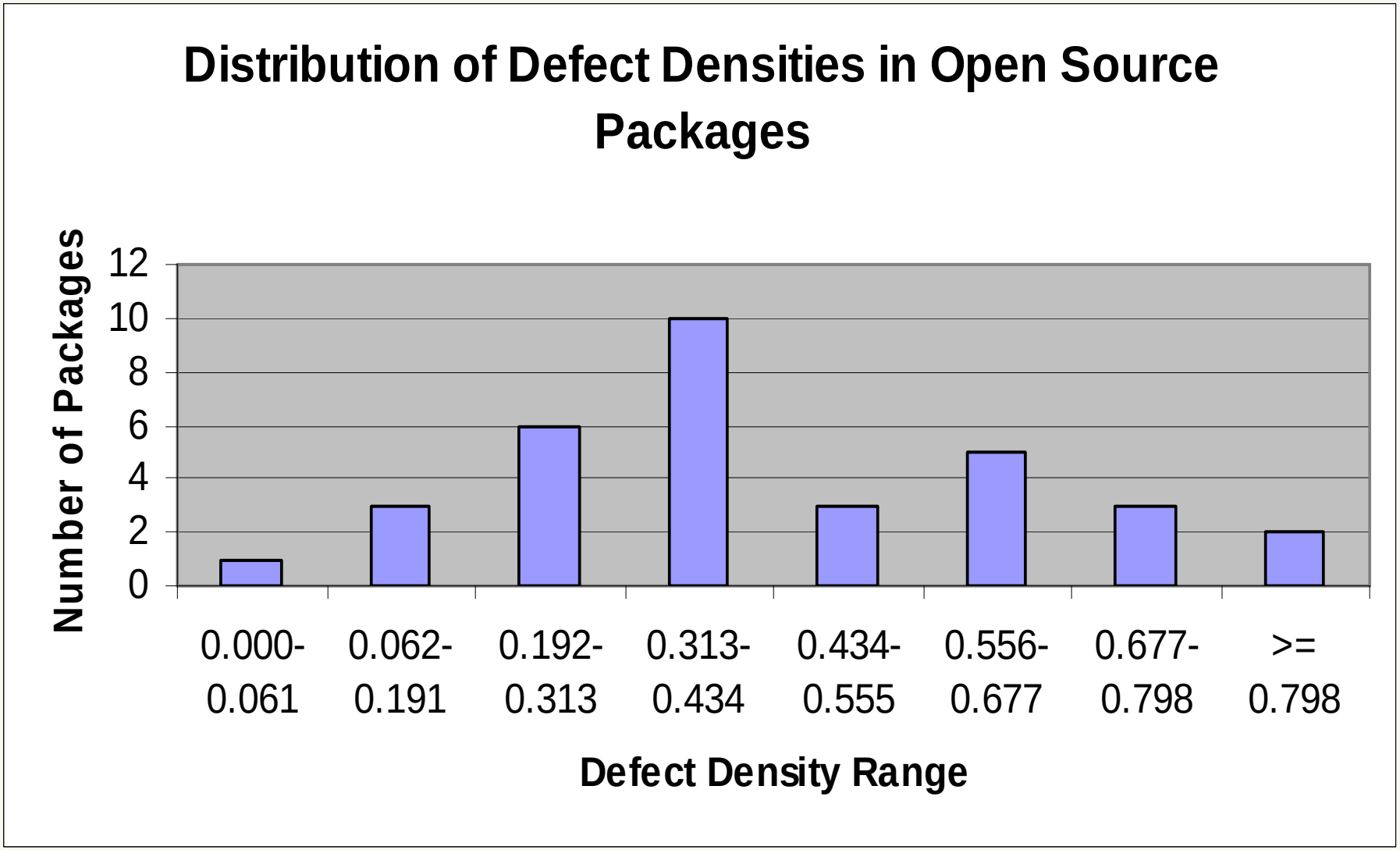
	Linux	PHP	Perl	Python
バグ合計数	1062	205	89	96
システム/プロセスのクラッシュ	585	126	40	54
予期しない動作	117	40	8	7
メモリ リーク	173	23	26	21
バッファオーバーラン	187	16	15	14

欠陥密度 - “L” および “P”

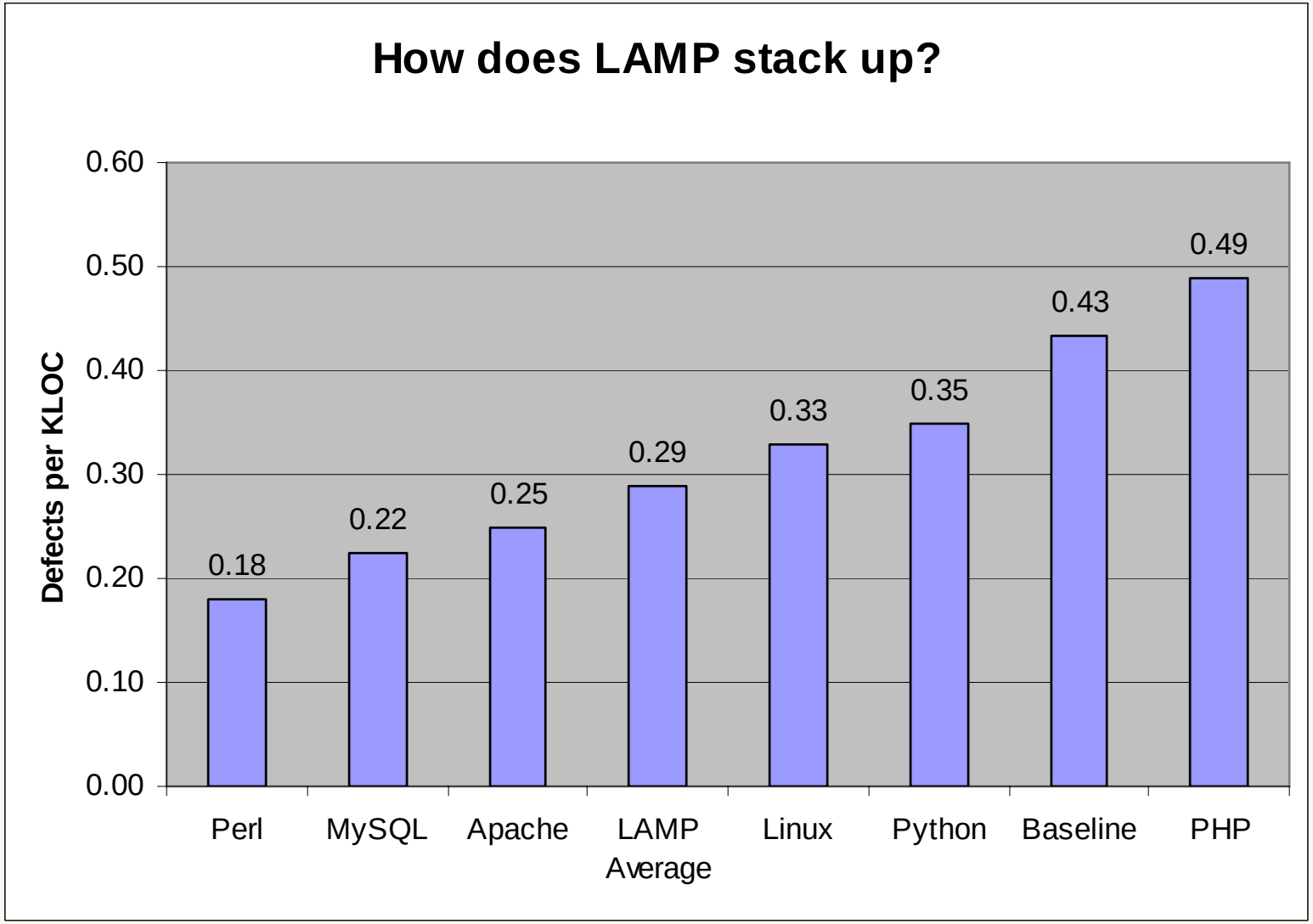
coverity

パッケージ	密度 (コード 1000 行あたりのバグ)
Linux	.33
PHP	.49
Perl	.18
Python	.35

傾向: 欠陥密度の基準


Distribution of Defect Densities in Open Source Packages
Number of Packages
12
10
8
6
4
2
0
0.000-0.061
0.062-0.191
0.192-0.313
0.313-0.434
0.434-0.555
0.556-0.677
0.677-0.798
>= 0.798
Defect Density Range

比較: LAMP 対基準


How does LAMP stack up?
Defects per KLOC
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0.18
0.22
0.25
0.29
0.33
0.35
0.43
0.49
Perl
MySQL
Apache
LAMP Average
Linux
Python
Baseline
PHP

PHP – 値が最も高い理由

- 88 の “FORWARD_NULL” バグ (合計 205 中)
- 非標準アサーションによるフォールス ポジティブ (10 のサンプルでは 3/10)

主な問題点

- コード内のいくつかの部分における NULL ポインタの不適切な取り扱い

例

```
5585    static int ZEND_ADD_ARRAY_ELEMENT_SPEC_TMP_TMP_HANDLER(ZEND_OPCODE_HANDLER_ARGS)
5586    {
5587            zend_op *opline = EX(opline);
5588            zend_free_op free_op1, free_op2;
5589            zval *array_ptr = &EX_T(opline->result.u.var).tmp_var;
5590            zval *expr_ptr, **expr_ptr_ptr = NULL;
5591            zval *offset= _get_zval_ptr_tmp(&opline->op2, EX(Ts), &free_op2 TSRMLS_CC);
5592
```

At conditional (1): "(opline)->extended_value != 0" taking true path

```
5593            if (opline->extended_value) {
```

Event **assign_zero**: Variable "expr_ptr_ptr" assigned value 0.

Also see events: [var_deref_op]

```
5594                    expr_ptr_ptr=NULL;
```

Event **var_deref_op**: Variable "expr_ptr_ptr" tracked as NULL was dereferenced.

Also see events: [assign_zero]

```
5595                    expr_ptr = *expr_ptr_ptr;
```

考慮すべき他の要素

coverity

	Linux	PHP	Perl	Python
コードの行数	3,171,631	419,192	495,100	272,118
保守担当者数	464	57	846	64
1.0 以降の経過期間 (日数)	4,380	3,928	6,613	5,516
欠陥密度	0.33	0.49	0.18	0.35

“L” および “P” の推定される相関関係

- 経過期間対欠陥密度
  - ほとんど完璧な相関関係
  - 古いプロジェクトほど欠陥が少ない
- LOC (コード行数) 対欠陥密度
  - 相関関係なし
  - 最高密度は 419,192 の LOC を持つ PHP
  - 最低密度は 495,100 の LOC を持つ Perl
- 保守担当者数対欠陥密度
  - 保守担当者数が多いほど欠陥密度が低い
  - Python はこの限りではない
- 今後の作業 – さらに大きなサンプル サイズでこれらの相関関係を調査する

欠陥の公表

- 2006 年 3 月 6 日、欠陥データベースをオンライン化
- すべてのオープン ソース保守担当者に発表を通知
- 反応

5 日後

coverity

確認済み (修正済み)	Linux	PHP	Perl	Python
システム/プロセス クラッシュ	73 (3)	0	3 (0)	46 (36)
予期しない 動作	43 (0)	2 (0)	3 (2)	6 (0)
リソース リーク	12 (1)	0	4 (2)	21 (12)
バッファ オーバーラン	15 (1)	0	1 (1)	9 (0)
合計 修正数	5	0	5	48

5 日後の勝者

- Python
  - 48 のバグを修正
  - 残りの 38 のバグのうち、23 は “影響少” とマーク付けされ、修正に値しないものと判断

30 日後

coverity

確認済み (修正済み)	Linux	PHP	Perl	Python
システム/プロセス クラッシュ	87 (49)	27 (7)	15 (12)	40 (44)
予期しない 動作	32 (23)	10 (6)	3 (2)	5 (2)
リソース リーク	31 (18)	6 (6)	5 (3)	21 (18)
バッファ オーバーラン	39 (21)	14 (3)	1 (1)	9 (2)
合計 修正数	111	22	18	66

30 日後の概要

- 4,272 の欠陥を確認
- 2,594 の修正
- 2 のセキュリティ勧告 (X.org および NetBSD)
- 3 つの “クリーン” なパッケージ – Amanda、Python、XMMS
  - Amanda は初期の欠陥密度が最大
- 5 つのほとんどクリーンなパッケージ – SQLite、Ethereal、Icecast、Squid、Samba
- 500 人以上の開発者が結果を見るために登録

最新の結果は、次のサイトを参照。

scan.coverity.com

まとめ

- オープン ソース開発者は品質を重視している
- LAMP スタックの平均欠陥密度は 32 の OSS パッケージの基準を下回っている
- PHP には最大の欠陥密度があり、その主な原因は NULL ポインタの不適切な取り扱いにある
- 今後の作業
  - パッケージの経過期間、保守担当者数、LOC と欠陥密度との相関関係を調査する
  - 特定のバグ カテゴリを詳細にわたって調査し、アーキテクチャ上の機能と相関させる