岡山PHP勉強会04

PHPインフラ周りのセキュリティ

自己紹介

- 大垣 靖男（おおがき　やすお）
- yohgaki@ohgaki.net / yohgaki@php.net
- twitter/facebook: yohgaki
- http://www.ohgaki.net/

BOSS-CON JAPAN

BOSS-CON JAPAN
ビジネスOSSコンソーシアム・ジャパン

PHP SECURITY ALIANCE　Rails Advanced Partners　BOSS-CON JAPAN BUSINESS OSS CONSORTIUM JAPAN

BOSS-CON JAPAN | 協賛会社一覧 | ニュース | PHPコミュニティ | Railsコミュニティ | お問合せ

HOME

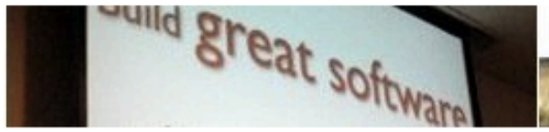
- BOSS-CON JAPAN
- PHPコミュニティ
- Railsコミュニティ
- お問合せ

RubyWorld Conference 2012 11/8-9 SPONSOR
※Railsアドバンスド・パートナーズとして協賛しています。

php
※PHPセキュリティ・アライアンスがPHPカンファレンス2012にシルバースポンサーとして協賛しています。

BOSS-CON JAPAN
bosscon_japan

セミナー情報



11月8日開催　RubyWorld Conference開催実行委員主催
RubyWorld Conference2012に協賛しました

9月15日開催
PHPカンファレンス2012に協賛しました。

9月13日開催　テンプスタッフ・テクノロジー主催
「オープンソースで構築する認証基盤」
～ハイブリッド・クラウド時代の統合認証からシングルサインオンまで～
に協賛し、理事長の吉政忠志がプレゼンテーションを行います。

9月5日開催　マルチメディアスクールWAVE主催
まつもとゆきひろ　と増井雄一郎が創る「新しい"Ruby"」～mrubyとMobiRubyの今～に協賛しました。

8月24日開催　テンプスタッフ・テクノロジー主催
『PHP初心者が陥りやすい10のトラブルと解決方法』セミナーに協賛しました。

8月24日開催　CTCテクノロジー主催
「クラウド時代のLinuxセミナー～クラウド時代のエンジニアのLinux/OSSスキルとは？～」に協賛し、理事長の吉政忠志がプレゼンテーションを行いました。

PHP技術者認定機構


特定非営利活動任意団体
PHP技術者認定機構
スポンサー
広告枠
OSS
Consortium 案件相談 受付中
オープンソースでビジネスに加速を!
試験概要
認定教材
認定スクール
協賛について
当機構について
お問合せ・FAQ
PHP技術者認定ウィザード2012
いよいよ投票開始!初級・上級合格者の一票で
今年のウィザードを選んで下さい。投票ページへ
学割
Webデザイナーやソーシャル系／GAMEプログラマーを目指す
学生に朗報！PHPスタンダード・スキルの習得証明ができます。
セミナー
PHPMATSURI 2012
11月3日、4日に福岡で開催されるPHPMATSURI2012 in 福岡に協賛しました。
詳細はこちらをごらんください。
特別協賛会社
O'REILLY®
NIFTY Cloud
ニフティ クラウド
協賛会社（無料）を募集しています。
全協賛会社リストはこちらをご覧ください。
受験予約/申込
全国100か所以上の受験会場で1年中受験ができます。受験の予約と申込は以下のこちらから。

PROVE for PHP

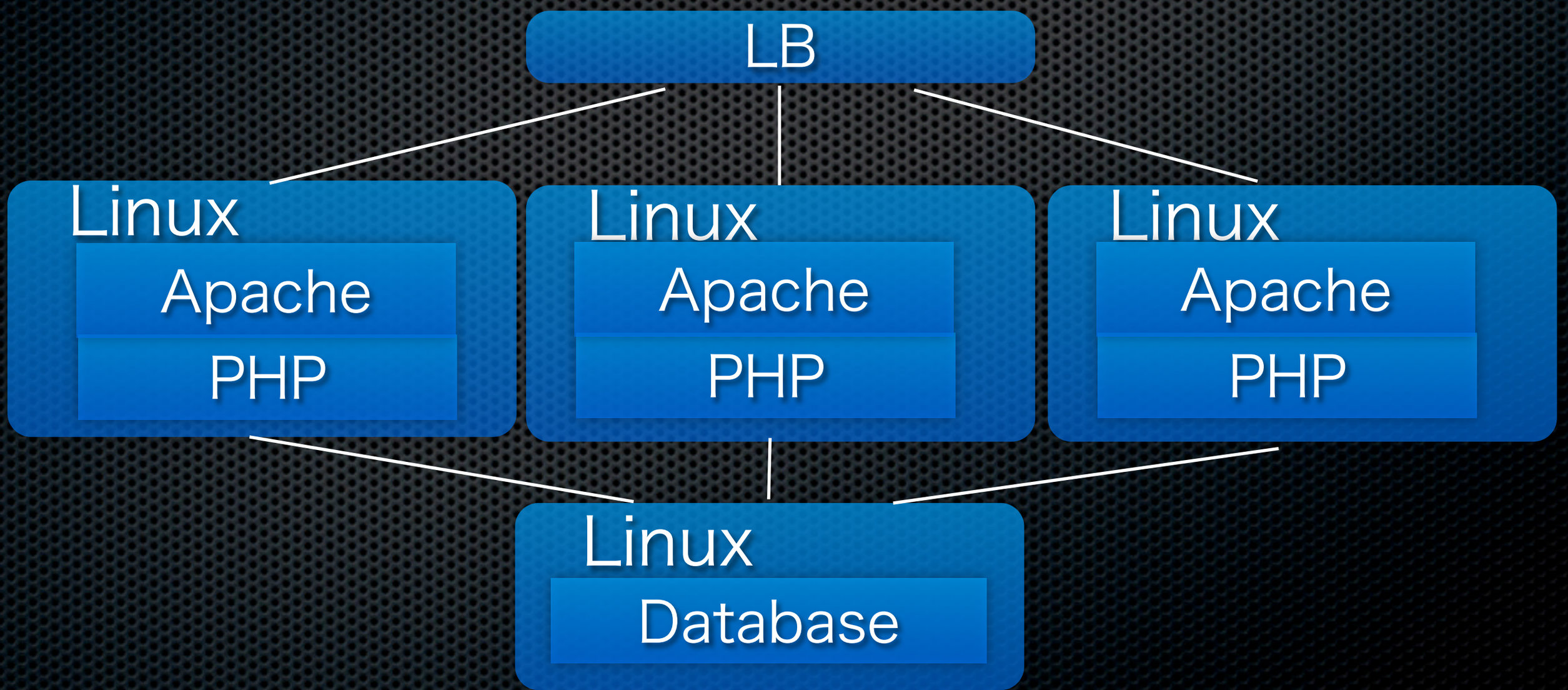
その他

セキュリティの基本

セキュリティ対策はトータルな対策

下回りの対策が疎かでは安全性は保てない

典型的なPHPシステム



守るべき物

- サーバサービス
  - SSH、HTTP、Mail、DBMS、etc
- リソース
  - ファイル、DBMS、ネットワーク、etc

脅威

脅威

- DoS

脅威

- DoS
- 脆弱性

脅威

- DoS
- 脆弱性
- 構成ミス

脅威

- DoS
- 脆弱性
- 構成ミス
- 設定ミス

DoS

DoS

- DoSとDDoS

DoS

- DoSとDDoS
- ネットワークとアプリケーション

脆弱性

- 脆弱性の除去
  - アップデート
- 脆弱性の隠蔽
  - WAF、IPS

構成ミス

- ネットワーク構成自体の問題
  - 経路、ARPスプーフィング
- SSLの未利用
  - 認証が必要なサービスはSSLであるべき
  - 全ての通信をSSL化

設定ミス

- Webサーバの設定ミス
  - ディレクトリリスティング／ファイルアクセス
  - 認証設定／アドレス・ドメイン制限
- PHPの設定ミス
  - php.ini
  - 危険なモジュールの利用

DoS

LB（Load Balancer）

- UltraMonkeyL7 - L7 switch
- iptablesが手軽 - L3
  - DNAT
- DNS
  - 地域によって分散
  - BCPとしても有効

接続制限

- iptables
  - hashlimit - IP/Portでの接続制限
  - connlimit - IPでの接続数制限
  - recent - 接続数制限
  - limit - IPでの接続制限
- mod_security - Apache HTTPD/Nginx/IIS
  - URLベースで動作可能 - WAFとして機能
  - Alternative: mod_cband, mod_evasive

iptables - hashlimit

```
/sbin/iptables -A INPUT -p tcp -m state --syn --state NEW --dport
22 -m hashlimit --hashlimit-name t_sshd --hashlimit 1/m --
hashlimit-burst 1 --hashlimit-mode $srcip --hashlimit-htable-
expire 120000 -j ACCEPT
```

【オプション内容】

```
-m hashlimit                ：hashlimitモジュールの利用
--hashlimit-name t_sshd     ：ハッシュテーブル名の指定
--hashlimit 1/m             ：1分間に1回
--hashlimit-burst 1         ：最大1回
--hashlimit-mode $srcip     ：送信元アドレスでリクエスト数を管理する
--hashlimit-htable-expire 120000  ：管理テーブル中のレコードの有効期間
（単位：ms）
```

iptables - connlimit

```
/sbin/iptables  -A INPUT -p tcp --syn --dport
22 -m connlimit --connlimit-above 3 -j REJECT
```

【オプション内容】

```
-m connlimit                 ：connlimitモジュールの利用
--connlimit-above 3          ：３以上
```

```
-A INPUT -p tcp --dport 80 -i eth0 -m state
--state NEW -m recent --set
/sbin/iptables -A INPUT -p tcp --dport 80 -i
eth0 -m state --state NEW -m recent --
update --seconds 100 --hitcount 10 -j
REJECT
```

脆弱性

脆弱性の除去

- とにかくアップデート
  - PROVE for PHP :)

脆弱性の隠蔽

- バーチャルパッチ
  - mod_securityなどのWAFは仮想的にパッチを当てることが可能
  - mod_rewrite
  - iptables string module
- 仮想パッチは「取り敢えず」の対策

構成ミス

ネットワーク構成の問題

- 経路
  - そもそもサーバが設置されている環境が信用できない
- ARPスプーフィング
  - ARPスプーフィングによるMTIMは容易

SSLの利用

- 認証が必要なサービスはSSLを利用すべき
  - 全ての接続をSSL化する事が望ましい
    - SSL以外ではセッションIDを送信しない
      - secure属性＋httponly属性
- SSLは完璧ではない
  - SSLのストリップは簡単

ユーザ

- 最大の構成ミスと言えるのは利用者
- パスワードの使い回し
- 危険な経路の利用（勉強会のWiFi利用）
- プロトコルの確認

設定ミス

Webサーバの構成ミス

- ディレクトリリスティング／ファイルアクセス
- 認証設定／アドレス・ドメイン制限
- そもそもPHPスクリプトを実行可能に設定していない
- PHPのソースファイルが参照可能
  - .phpsの有効化

php.ini

- php.iniは難しくはないが、意味を正しく理解しておく必要がある

php.iniの設定場所

- php.iniファイル
  - PHPIniDir設定 - Apache httpd
  - PHPRC環境変数
    - php-SAPI名.ini
- Apache設定ファイル(httpd.conf, .htaccess)
  - php_admin_flag/value, php_flag/value
- PHPスクリプト

register_globals

- ユーザ入力をグローバル変数として初期化
- デフォルト：Off
- 推奨：Off
- PHP5.4では設定自体が削除されている

magic_quote_pgc

- ユーザ入力にaddslahses関数を自動的に適用
- デフォルト：Off (PHP5.3>=)
- 推奨：Off
- PHP5.4では設定自体が削除されている

magic_quote_runtime

- 実行時にaddslahses関数を自動的に適用
- デフォルト：Off
- 推奨：Off
- PHP5.4では設定自体が削除されている

expose_php

- PHP情報をヘッダなどに表記
- デフォルト：On
- 推奨：Off
- PHP5.5からGUID設定は削除されている

max_execution_time

- PHPスクリプトの最大実行時間
- デフォルト：30 秒
- 推奨：10秒
- 10秒以上レスポンスが必要なWebページは論外に遅い。5秒程度に設定しても構わないように作るべき。

max_input_time

- 入力を受け付ける時間
- デフォルト：60秒
- 推奨：10秒
- PERDIRの設定。アップロードスクリプト以外に長い時間は必要ない。

max_input_nesting_level

- 入力で許可する配列のネストレベル
- デフォルト：64
- 推奨：3
- 3次元配列以上が必要な事がある？？

max_input_vars

- GET/POST/COOKIEで許容するパラメータ数
- デフォルト：1000
- 推奨：20
- 20以上のパラメータを受け付ける必要がある？？

memory_limit

- PHPが利用するメモリ量
- デフォルト：128MB
- 推奨：8MB
- ファイルアップロードや特に複雑なスクリプト以外で8MB以上必要なケースは少ない
- PHP4のデフォルトは8MB

open_basedir

- ローカルファイル操作のベースディレクトリ
- デフォルト：なし
- 推奨：最も限定的な設定にする
- フェイルセーフ機能は有効に活用する

disable_functions

- 無効にする関数
- デフォルト：なし
- 推奨：シェル実行、パイプなどの関数を指定
- 使わない危険な関数は無効化

disable_classes

- 無効にするクラス
- デフォルト：なし
- 推奨：特に無し

display_errors
display_startup_errors

- クライアントへのエラー出力
- デフォルト：Off
- 推奨：Off
- エラーはクライアントに送信してはならない

log_errors

- エラーを記録する
- デフォルト：On
- 推奨：On
- 全てのエラーは記録されなければならない

log_error_max_len

- エラーログの最大長
- デフォルト：1024
- 推奨：4096
- 攻撃者がどのような攻撃を行ったのか、手口を知る必要がある

ignore_repeated_errors
ignore_repeated_source

- 同じエラー/同じソースのエラーを無視
- デフォルト：Off
- 推奨：Off
- 全てのエラーは記録すべき

default_charset

- デフォルトのテキストエンコーディング
- デフォルト：なし
- 推奨：UTF-8（使用中のエンコーディング）
- 文字エンコーディングは明示的に指定しなければならない

file_upload

- ファイルアップロードを有効にする
- デフォルト：On
- 推奨：Off
- 99%以上のスクリプトはファイルのアップロード機能は必要としていない

upload_tmp_dir

- アップロードに利用する一時ディレクトリ
- デフォルト：なし（システムデフォルト）
- 推奨：安全なディレクトリに設定
- 共用システムでは安全なディレクトリに設定する必要がある

max_file_uploads

- 同時にアップロードできるファイル数
- デフォルト：20
- 推奨：1〜必要な数
- 同時に20もアップロードを許可するアプリケーションはほぼ無い

allow_url_fopen

- http://, ftp://などでファイルをオープン
- デフォルト：On
- 推奨：Off
- 必要な場合にのみ有効化すればよい

allow_url_include

- include/require文でURL形式のリモートスクリプトの読み込み
- デフォルト：Off
- 推奨：Off
- 99%のスクリプトはリモートスクリプトの読み込みは必要としていない

default_socket_timeout

- ソケット操作のタイムアウト時間
- デフォルト：60秒
- 推奨：10秒
- 10秒でも長すぎるくらい

extension

- モジュールをロードする
- デフォルト：いろいろ
- 推奨：必要なモジュールのみロード
- PECLのモジュールはメンテナンス状態が悪いものも

session_save_path

- セッションデータの保存場所
- デフォルト：/tmp
- 推奨：専用のディレクトリ
- セッションデータが他のユーザから読めるのは好ましくない

session.name

- セッション名
- デフォルト：PHPSESSID
- 推奨：任意の名前
- セッション名は独自に付ける方が好ましい

session.use_cookies

- セッション管理にCOOKIEを使う
- デフォルト：1
- 推奨：1
- 普通はセッション管理にクッキーを使うべき

session.cookie_secure

- HTTPS専用クッキーデフォルト：０
- 推奨：1
- HTTPS以外でセッションクッキーを送信しては、意味が半減

session.cookie_httponly

- HTTPのみでクッキーを利用
- デフォルト：０
- 推奨：1
- Javascriptにセッションidクッキーが必要？必要なアプリであるならアプリを改修する

session.cookie_lifetime

- セッションクッキーの有効期限
- デフォルト：０
- 推奨：０
- 有効期限０のクッキーはブラウザ終了と同時に削除される

session.cookie_path

- セッションIDクッキーのパス
- デフォルト：/
- 推奨：/
- これを変える意味はあまりない

session.cookie_domain

- セッションIDクッキーのdomain
- デフォルト：なし
- 推奨：なし
- これを変える意味はあまりない

session.gc_probability
session.gc_divisor

- GCが発生する確率を調整
  gc_probability/gc_divisor
- デフォルト：
  pc_probability = 1, gc_divisor = 1000
- 推奨：調整
- アクセスが少ないサイトには確率は低すぎ、多いサイトには確率が高すぎる

session.gc_maxlifetime

- GCされるまでの時間
- デフォルト：1440秒
- 推奨：調整
- 高い安全性が求められるサイトでは24分は長すぎ、そうでないサイトでは短すぎる
- この設定を使って自動ログインの代わりにしない事

session.referer_check

- URLベースのセッションIDでリファラーをチェック
- デオフォルト：なし
- 推奨：設定する
- URLベースのセッション管理を行う場合、自分のドメイン名を設定する

session.cache_limiter

- セッション管理を行うページでのキャッシュ設定
- デフォルト：nocache
- 推奨：nocache
- 通常はnocacheのままで良いが、キャッシュさせる場合はprivate（プロキシにキャッシュさせない）に設定する

session.cache_expire

- キャッシュさせた文書の有効期限
- デオフォルト：180分
- 推奨：調整
- 文書によって調整する

session.use_trans_sid

- URLベースのセッション管理を有効にする
- デオフォルト：0
- 推奨：0
- 通常はURLベースのセッション管理は行わない

session.hash_function

- セッションIDのハッシュ関数
- デオフォルト：0　（MD5）
- 推奨：1　（SHA1）
- MD5を利用する意味はあまりない

session.hash_bits_per_character

- セッションIDに利用する文字
- デフォルト：5 (0-9, a-v)
- 推奨：6 (0-9,a-z, A-Z, “-”, “,”)
- 若干だが毎回送信されるセッションIDの長さが減らせる

mbstring.internal_encoding

- mbstringの内部文字エンコーディング
- デフォルト：なし
- 推奨：UTF-8 (利用する文字エンコーディング)
- 文字エンコーディングは明示的に指定しておく

mbstring.strict_detection

- 文字エンコーディングの検出を厳格に行う
- デフォルト：Off
- 推奨：On
- 出来る限り厳格にチェックする方が好ましい

まとめ

- インフラ周りは出来る限り、限定的に設定し、アップデートを確実に行う
- スライドのPDF
  - http://blog.ohgaki.net/ に公開予定

おまけ - PHP5.5情報

おまけ - PHP5.5情報

- opcache - ZendOptimizer+がPHPに統合

おまけ - PHP5.5情報

- opcache - ZendOptimizer+がPHPに統合
- Generator - yieldが追加

おまけ - PHP5.5情報

- opcache - ZendOptimizer+がPHPに統合
- Generator - yieldが追加
- finally - finally, there is finally

おまけ - PHP5.5情報

- opcache - ZendOptimizer+がPHPに統合
- Generator - yieldが追加
- finally - finally, there is finally
- foreach($ary as list($a, $b, $c))

おまけ - PHP5.5情報

- opcache - ZendOptimizer+がPHPに統合
- Generator - yieldが追加
- finally - finally, there is finally
- foreach($ary as list($a, $b, $c))
- Password hash API - cryptラッパー

おまけ - PHP5.5情報

- opcache - ZendOptimizer+がPHPに統合
- Generator - yieldが追加
- finally - finally, there is finally
- foreach($ary as list($a, $b, $c))
- Password hash API - cryptラッパー
- 残念ながらC#ライクなget/setは無し

ご清聴ありがとう

ございました