

Wiki小話Vol.4  
Wikiセキュリティオムニバス  
～下(インフラ)から上(Wiki)まで～

---

2005/12/22

宮本 久仁男 a.k.a. wakatono

wakatono@todo.gr.jp

# 話の前に...ちょっと質問

- まず、顔を伏せてください
  - 拳手してる様子が見えないように(汗)
- 自分がサーバを管理してる人
  - WikiサーバにLinuxを使ってる人
  - WikiサーバにWindowsを使ってる人
  - OSインストール後アップデートをしたことがある人
  - 「意識して」パケットフィルタ設定をしてる人
- 自分が使ってるソフトのChangelogを見たことがある人
- 侵入されたことがある人
  - 侵入された後に再インストールをした人
- 自分で自分のシステムを攻撃したことがある人
- 自分が作ったプログラムの脆弱性試験をしたことがある人

# Agenda

- とりあえず知っておこう  
～上を守るにはまず下から
- 歴史に学ぼう～Changelogは教訓の宝庫？
  - Case Study:見えるファイル
- 簡単にでも環境のチェックを

とりあえず知っておこう  
～上を守るにはまず下から

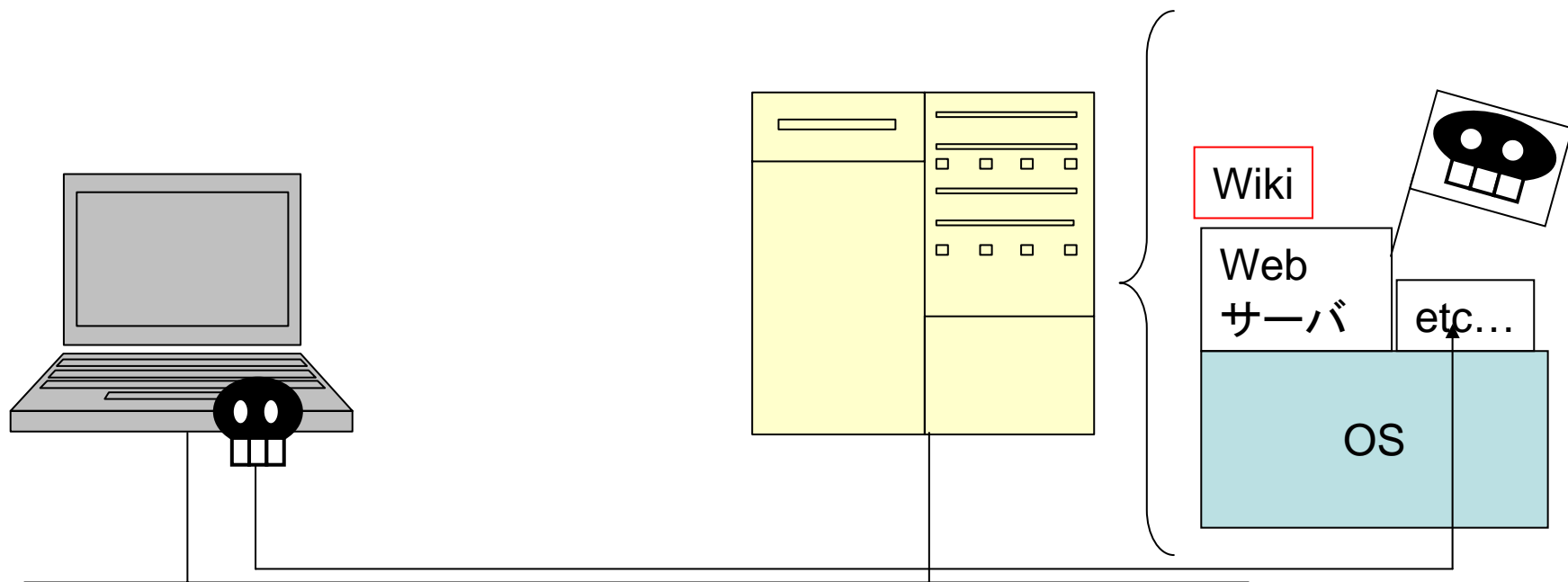
全部自分でやる人は特に注意

# Wikiは何？

- WikiはWiki
- WikiはWebアプリケーション
- Wikiの下にはHTTPサーバ
- HTTPサーバの下にはOS
- そしてそれらはネットワーク経由で使われる

アプリケーションの作り方だけ知ってても  
きつい...

# おーざっぱにはこんな感じ？



- Wiki「だけ」とかWiki～Webサーバ「だけ」とかを頑丈にしても、迂回路が...

# 動かすものの例

- Apache2
  - Wiki動作のためのプラットフォーム
- 各種言語処理系 (PHP, Ruby, Perl, etc...)
  - これまたWiki動作のためのプラットフォーム
- OpenSSH Server
  - サーバメンテナンス用のログインの口提供
- BIND9等
  - 自分でサーバを運用してる人が名前解決のために起動させている

# 最近発見された脆弱性の例

- Apache2
    - CAN-2005-1268
    - CAN-2005-2088
    - CAN-2005-2700
    - CAN-2005-2728
  - OpenSSL
    - CVE-2005-2969
    - CVE-2005-2969
  - OpenSSH
    - CAN-2003-0693
    - CAN-2003-0695
    - CAN-2003-0682
  - BIND9
    - CAN-2003-0914
- DoS系のものが大半
    - だからといって安心はできない
  - たまにBuffer Overflow系のものも
    - 脆弱性が発見されると、今だと早期にPoCが公開される
    - PoC: 検証コードの意味
  - いきなり飛んできてくることは稀
    - ポートスキャンなどで品定めw

# 脆弱性によらなくても...

- 力技でやってくるお客様もいらっしゃいます
  - /var/log/auth.log の例
    - tail -10000 auth.log > /tmp/sshlog
    - grep Illegal /tmp/sshlog | wc -l
      - どんくらいでしょw
    - 8815...orz
    - 期間:2005/12/9 12:45~12/12 01:49
- パスワード認証使ってなければ恐くない
  - 負荷が上がるのがうざいけど...
- 次のスライドに力技の例を...

# 力技の例

```
Dec 12 01:34:20 kid sshd[21109]: Illegal user asante from 66.xx.xx.xx
Dec 12 01:34:22 kid sshd[21111]: Illegal user x from 66.xx.xx.xx
Dec 12 01:34:27 kid sshd[21113]: Illegal user y from 66.xx.xx.xx
Dec 12 01:34:29 kid sshd[21115]: Illegal user w from 66.xx.xx.xx
Dec 12 01:34:31 kid sshd[21117]: Illegal user z from 66.xx.xx.xx
Dec 12 01:34:33 kid sshd[21119]: Illegal user ashley from 66.xx.xx.xx
Dec 12 01:34:35 kid sshd[21121]: Illegal user ashmore from 66.xx.xx.xx
Dec 12 01:34:37 kid sshd[21123]: Illegal user askin from 66.xx.xx.xx
Dec 12 01:34:39 kid sshd[21125]: Illegal user assan from 66.xx.xx.xx
Dec 12 01:34:41 kid sshd[21127]: Illegal user asta from 66.xx.xx.xx
Dec 12 01:34:43 kid sshd[21129]: Illegal user astrid from 66.xx.xx.xx
Dec 12 01:34:46 kid sshd[21131]: Illegal user astor from 66.xx.xx.xx
Dec 12 01:34:48 kid sshd[21133]: Illegal user ates from 66.xx.xx.xx
Dec 12 01:34:50 kid sshd[21135]: Illegal user aten from 66.xx.xx.xx
Dec 12 01:34:52 kid sshd[21137]: Illegal user atila from 66.xx.xx.xx
Dec 12 01:34:54 kid sshd[21139]: Illegal user attila from 66.xx.xx.xx
Dec 12 01:34:56 kid sshd[21141]: Illegal user aude from 66.xx.xx.xx
Dec 12 01:34:58 kid sshd[21143]: Illegal user aura from 66.xx.xx.xx
Dec 12 01:35:00 kid sshd[21145]: Illegal user ammount from 66.xx.xx.xx
Dec 12 01:35:02 kid sshd[21147]: Illegal user account from 66.xx.xx.xx
Dec 12 01:35:04 kid sshd[21151]: Illegal user acount from 66.xx.xx.xx
Dec 12 01:35:06 kid sshd[21153]: Illegal user aurel from 66.xx.xx.xx
Dec 12 01:35:08 kid sshd[21155]: Illegal user aurelio from 66.xx.xx.xx
Dec 12 01:35:10 kid sshd[21157]: Illegal user aurora from 66.xx.xx.xx
Dec 12 01:35:12 kid sshd[21159]: Illegal user austin from 66.xx.xx.xx
Dec 12 01:35:14 kid sshd[21161]: Illegal user aust from 66.xx.xx.xx
```

こんなの2~3日で8000over

...自分でサーバ運営されてる方は  
チェックをお勧めします...

# 大原則

- 普通に公開サーバを管理すべし
  - 何でもいっしょ
  - 自分が使ってるOSのSecurity Advisoryなどはチェックしよう
- 不要なモンは入れるな
  - 必要なモンは入れるとして、管理をきちんと
- アップデートはきちんと実施しよう
  - メンテナンスの一環
  - 必要なモンに脆弱性が発見された時の対処
- 何が動いてるかは把握しよう
  - 不要なモンが入ってなければある程度は把握可能

# 歴史に学ぼう

Changelogは教訓の宝庫？

# Wikiエンジンを作ろうという人

- プログラムが組める人
  - Wikiの仕様をある程度把握してる人
  - 既存のWikiエンジンの仕様／ライセンスングが気に入らない人
  - 勉強をしようという人
- etc...

きっかけはいろいろございます

# まずは原則から

- 安全なWebアプリ開発の鉄則 2004
  - <http://www.soi.wide.ad.jp/class/20040031/slides/09/>
  - 多分、2005年版 (Internet Week 2005で講演された内容) もいずれ公開されるでしょう
  - Webアプリ開発の原則の宝庫
  - 全てのWebアプリケーション作成者は読んで遵守すべき内容
- WikiもWebアプリケーションの1つである

# 作った、さて現場投入／配布？

- ちょっとまとう...
  - ちゃんと試験しましたか？
- XSS対応
  - 入力文字列のチェックはやってますか？
- コマンドインジェクション対応
  - 入力文字列のチェック (ry
- その他いろいろ
  - 例：動かす環境は大丈夫？

# Wikiをどこで動作させる？

---

- Windows？UNIX？
- Apache？その他のWebサーバ？
- 導入手順は？
  - ...敷居を下げれば下げるほど、リスクも増大する（私見）
    - リスクが顕在化した例を後述します（汗）

# 変更履歴/Changelogは教訓の宝庫

- PukiWikiの場合（あくまで例）
  - .htaccessの追加、index.htmlの追加  
<http://pukiwiki.sourceforge.jp/?PukiWiki%2FDownload%2F1.4.4>
  - わりと最近のような気もするけど... > この改修

# Case Study1:見えるファイル

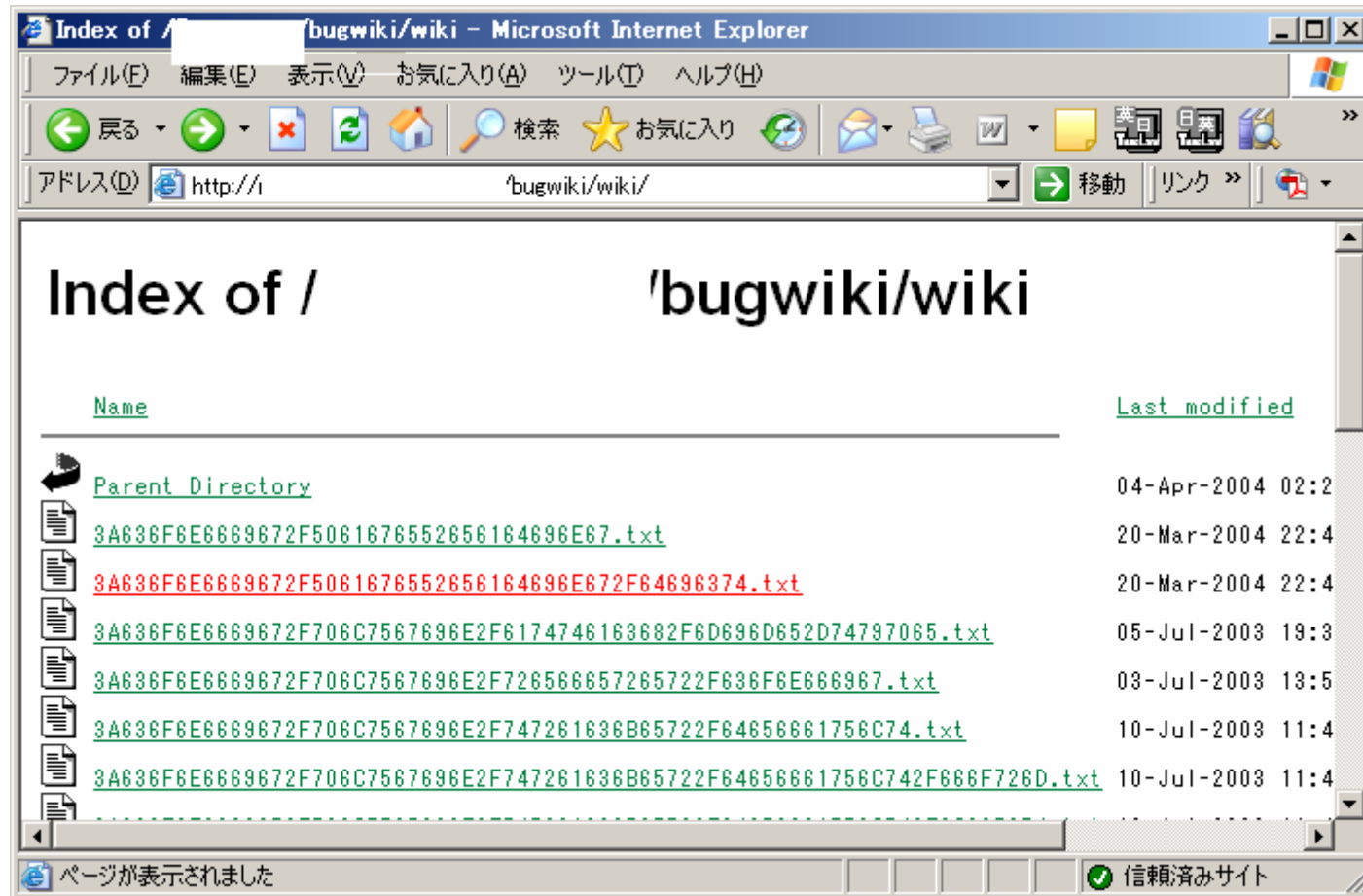
単純に設定ミス

# 古いPukiwikiの話

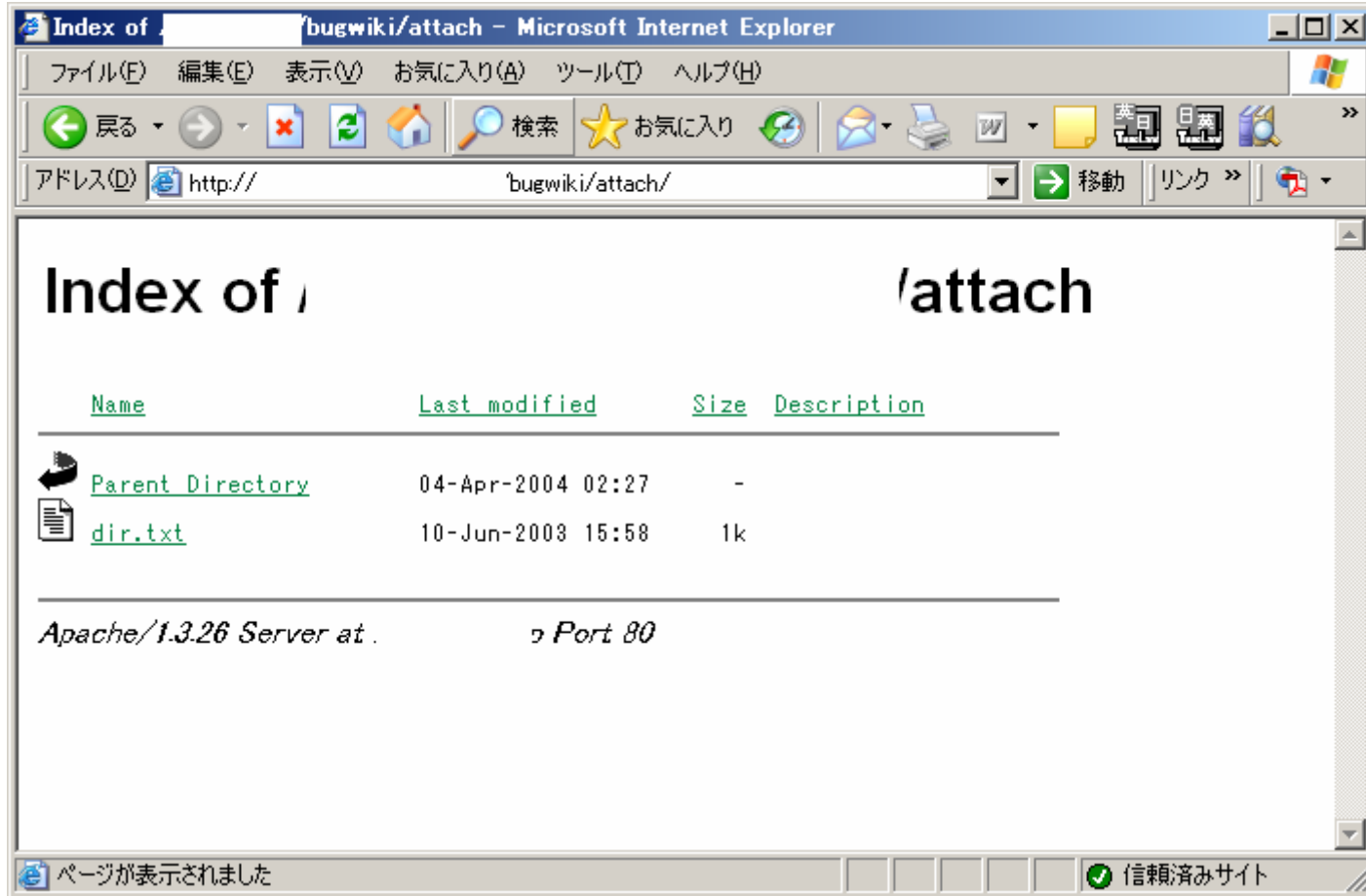
---

- データフォルダの下丸見えorz
- ファイル消しても残ってるorz
- 例を以下に...

# pukiwiki/wikiの下丸見えorz



# pukiwiki/attachの下も...orz



# 3分で検討した問題点...

- Webアプリケーションで使う(見えちゃいけない)領域が、public\_html配下に展開されている
  - 展開されている領域は見えないように縛らなきゃいけない
- 本来は、Webサーバ経由で見えるところに置くのが「間違い」
  - インストールを平易にしたトレードオフの例

# 救い: 現在は対策されている

- これらのディレクトリを含め、ディレクトリには index.htmlと.htaccessを作って「見えない」ようにしてある
- 古いものをいまだに使ってる人とかは注意を
- それ以前に、Webサーバのソフトウェアセキュリティとかわかってれば、こんな状態での配布はしない...ハズorz



# Case Study 2:

(影響を鑑みて削除(sorry))

# Changelogからわかること

- ありがちなバグ
  - (fixedとなっているところ)
- ありがちなミス
  - Force browsingを許してしまう例
- 変更された機能
  - 場合によっては理由も記述されている
- **明日はわが身**
  - 作る人にとっては他人事ではない

簡単にでも環境のチェックを  
～nmap,Nessus, Nikto,etc...

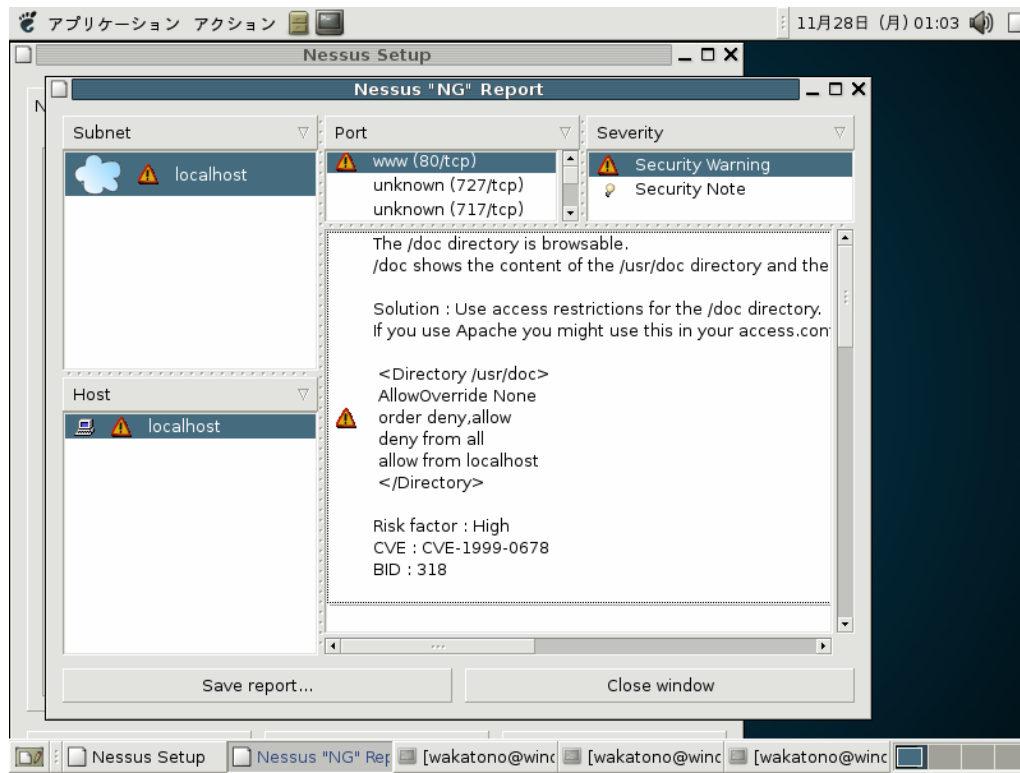
やるとやらないとでは、心構えの  
持ち方が変わってくる(はず)

# nmap:ポートスキャナ

- 基本的にはTCPポートのスキャン(探知)
  - どのポートが開いているか? をチェック
    - 攻撃者が手がかりをつかむためにも使われる
      - 最近のトレンドは、ポート決めうちではあるが...用心するにこしたことはなし(ツールは現に出回ってるし)
- 必要ないポートは閉じておこう
  - ただ、ICMPまで全て閉じるのは逆効果
- どのように使う?
  - 自分の環境に対して自分からnmap
  - 外部からポートスキャンをかけてくれるサービスも

# Nessus: 脆弱性スキャナ

- 監査対象となるホストの（既知の）脆弱性を監査



# Nikto:Webスキヤナ

- 脆弱性スキヤナ

```
wakatono@kid:~$ nikto -host www.example.com
```

```
-----  
- Nikto 1.34/1.31 - www.cirt.net  
+ Target IP:      192.168.0.1  
+ Target Hostname: www.example.com  
+ Target Port:    80  
+ Start Time:     Sat Dec 17 14:40:35 2005
```

```
-----  
- Scan is dependent on "Server" string which can be faked, use -g to override  
(略)
```

```
+ End Time:       Sat Dec 17 14:40:57 2005 (22 seconds)
```

```
-----  
+ 1 host(s) tested
```

# Snort:IDS(侵入検知システム)

- シグネチャベースのIDS
  - 攻撃パターンはシグネチャとして提供される
- ルールファイルは別途配布される
  - 有償サービスもある
  - コミュニティ向けのルール配布も行われる
  - Snort.org 以外からも配布される
- ...マニア向け(笑)
  - 侵入検知システムとしては強力
    - 誤検知もあるけど...
- 最近だと、検知できた時点では遅いということもあり(わかった時点ではやられてるw)、さらに進んだIDP(Intrusion Detection and Prevention やIPS(Intrusion Prevention System)も

# おまけ: HoneyPot, HoneyNet

---

- HoneyPot
  - 攻撃者の手口やツールの収集の手段
  - お客様に対してリアルな環境を提供
  - ツールも多く出ている
- Honeynet
  - ネットワーク監視をはじめとする技術を用いて、「なにが起きているか」を収集
  - <http://www.honeynet.org/>
- こんな手段もある、ということで...

# まとめ

プログラミング以外にも  
知らなきゃいけないことは多い

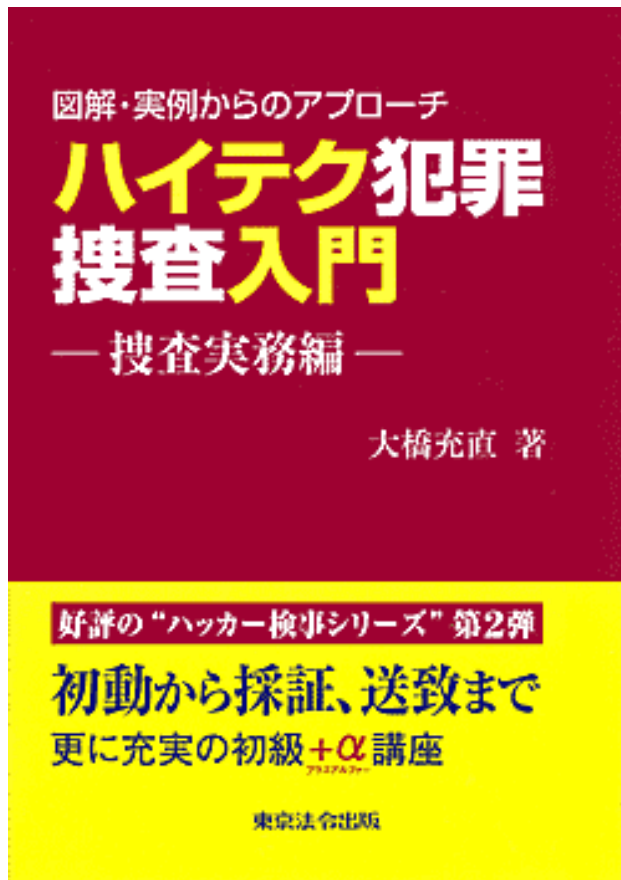
# プログラミング以外にも...

- 動作環境のこと、過去のバグ／脆弱性のことはある程度はリサーチしておいた方がよい
  - 動かすだけの人でも、最低限の運用管理(メンテナンス)については知っておくべき(というか知ってください)
    - つうか、知るだけじゃなくって実践を...
  - 携わる人は違っても似たようなミスはする可能性
    - Changelogは教訓の宝庫
      - ものによるけど
    - ソフトウェアは思ったとおりに動かない
      - ミスは必ずある(!)
      - ソフトウェアは書いてあるとおりに動く
- 作ったソフトウェアの機能と、危険性を考えよう
  - 悪用できそうな機能はデフォルトでは動作しないようにするなど
- 「動けばよい」という考えはもう古い
  - 脆弱性を作りこんでしまったら、多くの人に迷惑がかかる
  - 作るだけでなくフィードバックも受け付けよう
    - 想定外の意見があったら謙虚に受け止めよう

# ツールを使ったチェックにも限界

- やっぱり脆弱性は作らないのがベスト
  - 後付けのセキュリティから作りこみのセキュリティへ (by LAC 三輪社長)
- かといって、どうやったら作れるのか？
  - どうやったら安全なアプリケーションが作れる？
- 一方、攻撃者はどんな観点でWebサーバアプリケーションを狙う？
  - To Be Continued to 国分さんのセッション(汗)

# 不幸にもやられてしまった場合に 備えて



- ハイテク犯罪捜査入門  
捜査実務編
  - 不幸にも「不正アクセスされた」とか「結果として犯罪の被害にあってしまった」とかその他「ハイテク犯罪」に関連した情報提供を行う人は読んでおこう

To be continued to  
Mr. Kokubu.

