



IBM Software Group

開発工程からのセキュリティ検査

安心・安全なWebアプリケーションを実現する Rational ソリューション

2008/1

Rational software

→ Go to IBM

© 2007 IBM Corporation

Agenda

- Webアプリケーションのセキュリティ問題
- Webアプリケーションのセキュリティ対策の現状
- AppScan の紹介
- 開発工程におけるセキュリティ検査





IBM Software Group

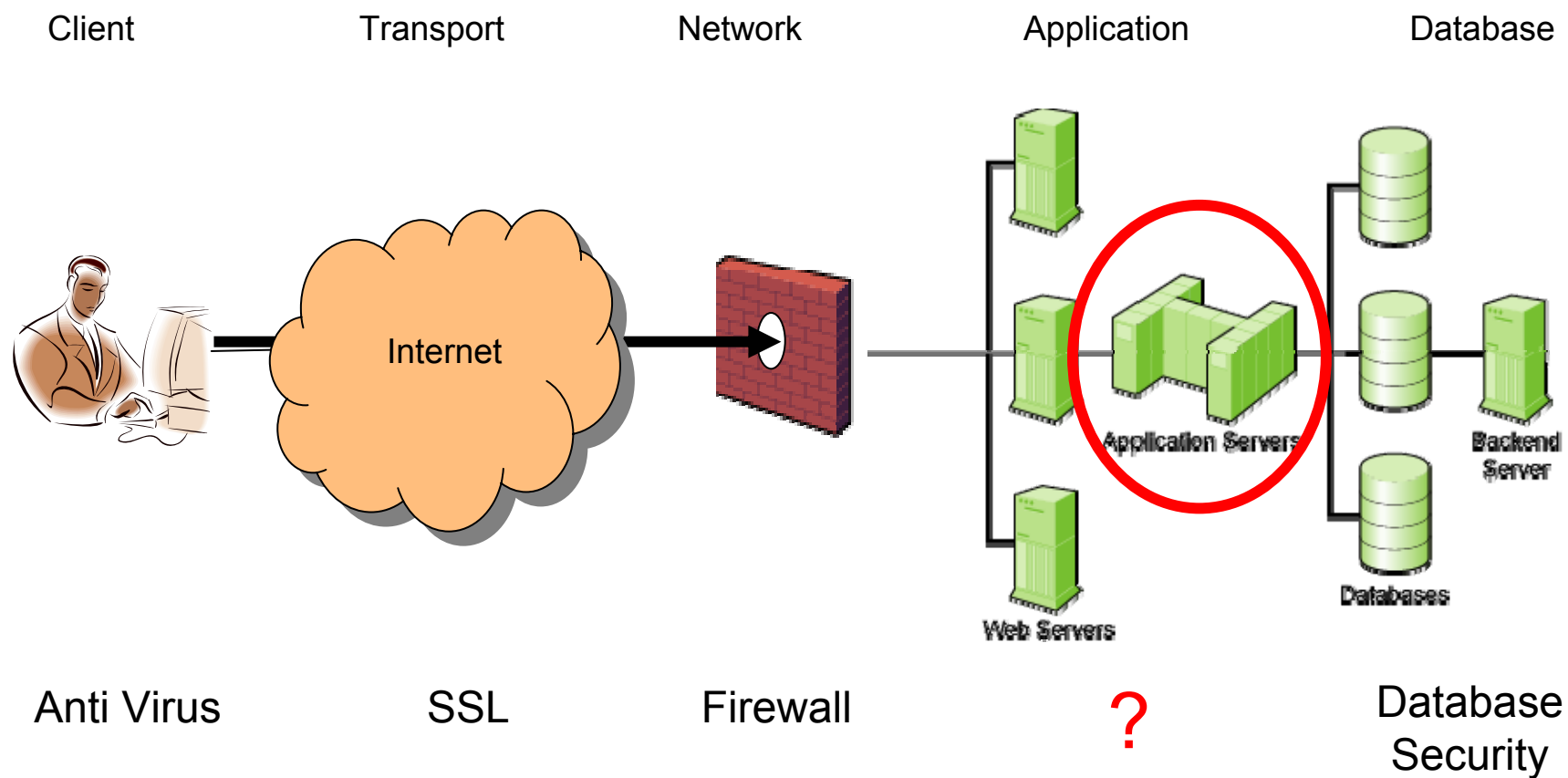
Web アプリケーションのセキュリティ問題

Rational software

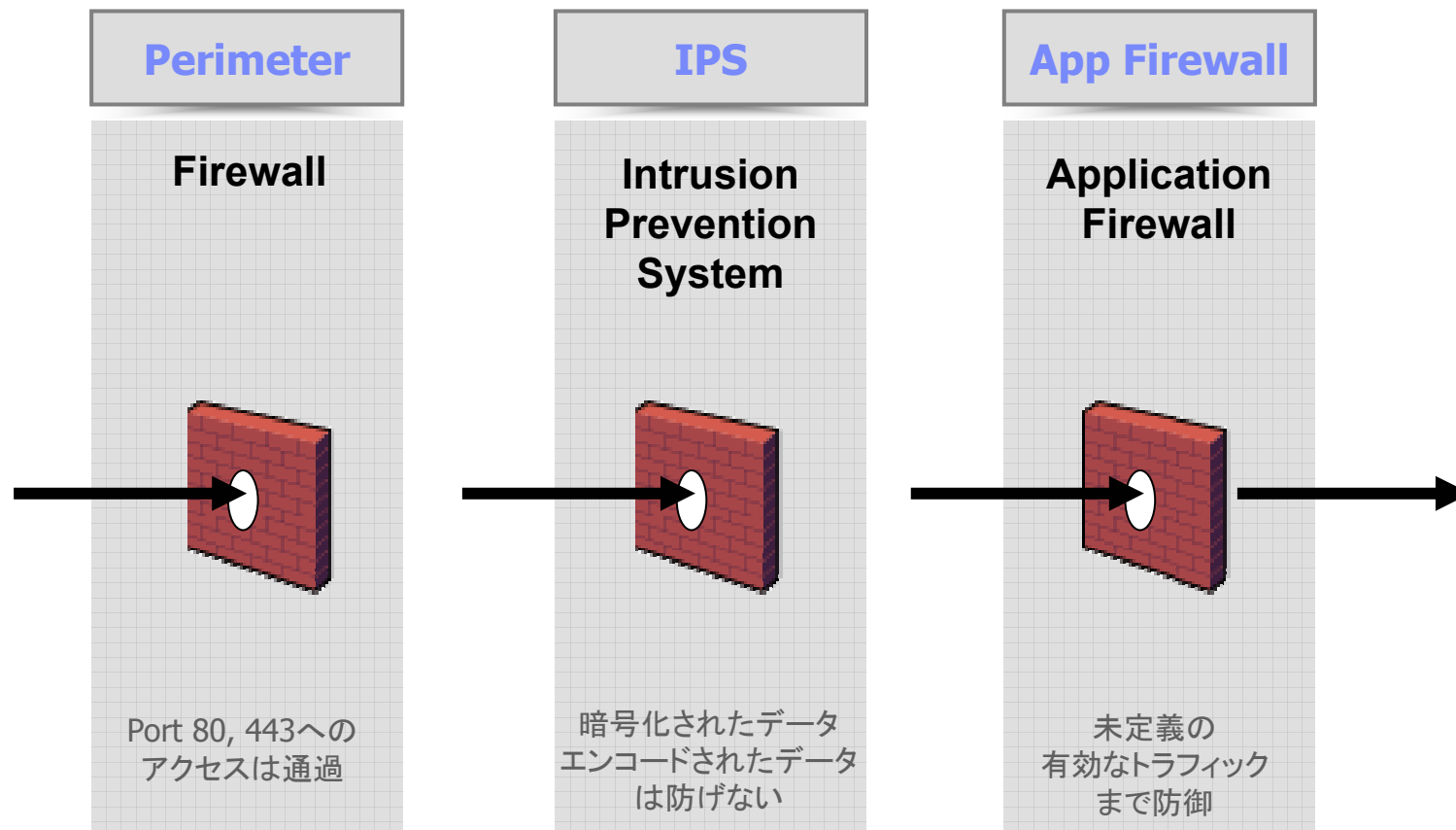
→ Go to **IBM**

© 2007 IBM Corporation

Web アプリケーション アーキテクチャ



Firewall によるプロテクション



Web アプリケーションに対する攻撃

- SQLインジェクション (SQL Injection)
- クロスサイトスクリプティング (Cross-Site Scripting – XSS)
- パラメータの改竄 (Parameter Tampering)
- Hiddenフィールドの不正操作 (Hidden Field Manipulation)
- クッキーの濫用 (Cookie Poisoning)
- バッファオーバーフロー (Buffer Overflow)
- 強制ブラウジング (Forceful Browsing)
- ステルスコマンド (Stealth Command)
- バックドア(デバッグオプションなど) (Debug Option and Backdoor)
- 設定ミスや既知の脆弱性を利用 (Know Vulnerability)



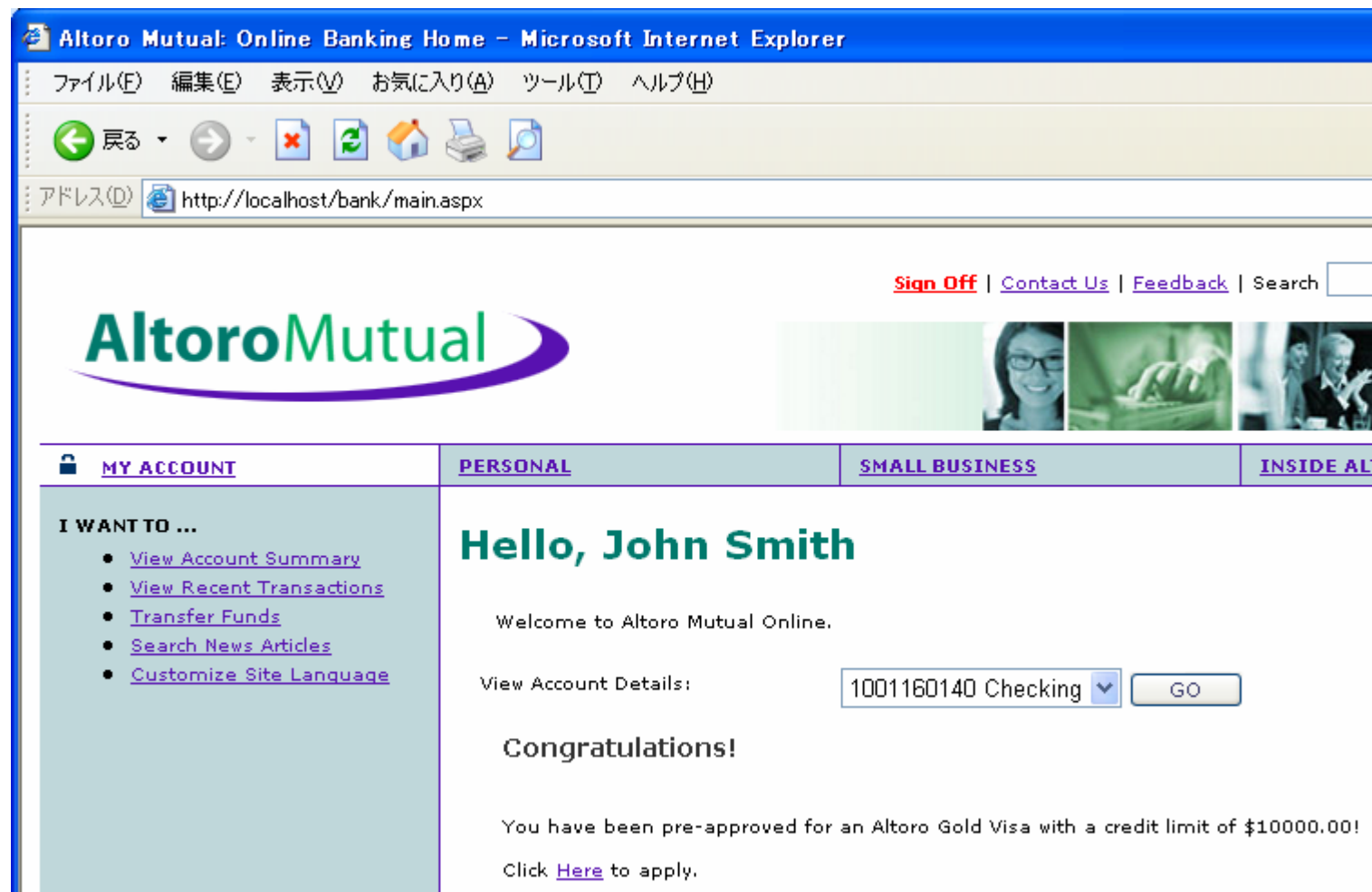
SQL インジェクション

```
SELECT * FROM users WHERE  
username='jsmith' AND password=' OR '1' = '1'
```

The screenshot shows a web application interface for online banking. At the top, there are tabs for 'PERSONAL', 'SMALL BUSINESS', and 'INSIDE AL'. The 'PERSONAL' tab is selected. On the left, there is a sidebar menu with links for 'Deposit Product', 'Checking', 'Loan Products', 'Cards', 'Investments & Insurance', and 'Other Services'. The main content area is titled 'Online Banking Login'. It contains a 'Username:' field with the value 'jsmith' and a 'Password:' field with masked characters. A blue arrow points from the password field to the text 'OR '1' = '1', indicating the injected payload. Below the password field is a 'Login' button.

PERSONAL	SMALL BUSINESS	INSIDE AL
<p>PERSONAL</p> <ul style="list-style-type: none">• Deposit Product• Checking• Loan Products• Cards• Investments & Insurance• Other Services <p>SMALL BUSINESS</p> <ul style="list-style-type: none">• Deposit Products• Lending Services• Cards• Insurance• Retirement	<p>Online Banking Login</p> <p>Username: <input type="text" value="jsmith"/></p> <p>Password: <input type="password" value="*****"/> → ' OR '1' = '1'</p> <p><input type="button" value="Login"/></p>	

SQL インジェクション



SQL インジェクション テストパターン (一部)

- ☒ Cookie を汚染する SQLインジェクション
 - ☒ Cookie の値を以下のように設定します: '
 - ☒ Cookie の値を以下のように設定します: \'
 - ☒ Cookie の値を以下のように設定します: ;
 - ☒ Cookie の値を以下のように設定します: "
 - ☒ Cookie の値を以下のように設定します: \"
 - ☒ Cookie の値を以下のように設定します:)
- ☒ HTTP ヘッダーの SQLインジェクション
 - ☒ HTTP 'X-Fowarded-For' ヘッダの値を以下に変更します: '
 - ☒ HTTP 'X-Fowarded-For' ヘッダの値を以下に変更します: %27
 - ☒ HTTP 'X-Fowarded-For' ヘッダの値を以下に変更します: %2527
 - ☒ HTTP 'X-Fowarded-For' ヘッダの値を以下に変更します: "
 - ☒ HTTP 'X-Fowarded-For' ヘッダの値を以下に変更します: %c0%a7
 - ☒ HTTP 'Referer' ヘッダの値を以下に変更します: '
 - ☒ HTTP 'Referer' ヘッダの値を以下に変更します: %27
 - ☒ HTTP 'Referer' ヘッダの値を以下に変更します: %2527
 - ☒ HTTP 'Referer' ヘッダの値を以下に変更します: "
 - ☒ HTTP 'Referer' ヘッダの値を以下に変更します: %c0%a7
- ☒ Probe SQL
 - ☒ SQLインジェクション
 - ☒ 元のパラメータ値に以下の文字列を追加します: ;
 - ☒ 元のパラメータ値に以下の文字列を追加します: having 1=1--
 - ☒ 元のパラメータ値に以下の文字列を追加します: 1 having 1=1--
 - ☒ 元のパラメータ値に以下の文字列を追加します: \' having 1=1--
 - ☒ 元のパラメータ値に以下の文字列を追加します:) having 1=1--
 - ☒ Append the following string to the original parameter value: %a5' having 1=1--
 - ☒ 元のパラメータ値に以下の文字列を追加します: %uFF07
 - ☒ 元のパラメータ値に以下の文字列を追加します: '
 - ☒ 元のパラメータ値に以下の文字列を追加します: '; select @@version,1,1,1--
 - ☒ 元のパラメータ値に以下の文字列を追加します: ; select * from master..sysmessages--
 - ☒ 元のパラメータ値に以下の文字列を追加します: ; select * from dbo.sysdatabases--
 - ☒ 元のパラメータ値に以下の文字列を追加します: ; select * from sys.dba_users--
 - ☐ SQLインジェクション (ポートリスナー テスト - Oracle /ヴァリエント)
 - ☐ ポート リスナーおよび OpenRowSet (MS-SQL) を使用した SQLインジェクション
 - ☒ SQLインジェクション コマンドの実行
 - ☒ 元のパラメータ値に以下の文字列を追加します: %20exec%20master..xp_cmdshell%20'vol'--
 - ☐ 元のパラメータ値に以下の文字列を追加します: '; exec master..xp_cmdshell 'echo [param name] in [path] is vulnerable >> C:\AppScanSQLTest.txt'-- (SQL Injection Probe)
 - ☐ 元のパラメータ値に以下の文字列を追加します: exec master..xp_cmdshell 'echo [param name] in [path] is vulnerable >> C:\AppScanSQLTest.txt'-- (SQL Injection Probe)
 - ☐ SQLインジェクションによるコマンドの実行 (ポートリスナー - MS-SQL /ヴァリエント)

XSS テストパターン (一部)

```

>%22%27><img%20src%3d%22javascript:alert([VARIANT ID])%22>
>""><img%20src%3D%26%23x6a;%26%23x61;%26%23x76;%26%23x61;%26%23x73;%26%23x63;%26%23x72;%26%23x69;%26%23x7
%22%20style%3D%22background:url(javascript:alert([VARIANT ID])%22%20OA%3D%22
--><script>alert([VARIANT ID])</script>
[ORIGINAL VALUE]+'+alert([VARIANT ID])+'
[ORIGINAL VALUE]%27%2Balert%28[VARIANT ID]%29%2B%27
[ORIGINAL VALUE]" + alert([VARIANT ID]) + "
[ORIGINAL VALUE]%22%2Balert%28[VARIANT ID]%29%2B%22
>'><script>alert([VARIANT ID])</script>
>"><script>alert([VARIANT ID])</script>
>'><%00script>alert([VARIANT ID])</script>
>"><%00script>alert([VARIANT ID])</script>
"><STYLE>@import"javascript:alert([VARIANT ID]);</STYLE>
"></IFRAME><script>alert([VARIANT ID])</script>
"></style><script>alert([VARIANT ID])</script>
"></title><script>alert([VARIANT ID])</script>
\u003Cscript\u003Ealert\u0028[VARIANT ID]\u0029\u003C/script\u003E
</TextArea><script>alert([VARIANT ID])</script>
>+ACJ--+AD4APB-SCRIPT+AD7-alert(1234)+ADz-/SCRIPT+AD7-
>+ACJ--+AD4APB-SCRIPT+AD7-alert(1234)+ADz-/SCRIPT+AD7-
onMouseOver=alert([VARIANT ID])><
%27%20onMouseOver=alert([VARIANT ID])><
%22%20onMouseOver=alert([VARIANT ID])><
[ORIGINAL VALUE]+alert([VARIANT ID])+
%A7%A2%BE%Bc%F3%E3%F2%E9%F0%F4%Be%E1%EC%E5%F2%F4%A8[VARIANT ID]%A9%Bc%Af%F3%E3%F2%E9%F0%F4%Be
[ORIGINAL VALUE]%22)%0d%0aalert([VARIANT ID])%27
<script>alert([VARIANT ID])</script>

```



参考

- Watchfire CBT
 - ▶ "Web Application Security Hacking 101"
- Watchfire ホワイト ペーパー
 - ▶ 「アプリケーション レベルでの一般的な 12 種類のハッカー攻撃」
<https://www.watchfire.com/securearea/whitepapers.aspx?id=83>
- 外部情報
 - ▶ WASC 脅威の分類
<http://www.webappsec.org/projects/threat/>
 - ▶ OWASP
http://www.owasp.org/index.php/OWASP_Top_Ten_Project
 - ▶ IPA
http://www.ipa.go.jp/security/vuln/20050623_websecurity.html





IBM Software Group

Web アプリケーションセキュリティ対策の現状

Rational software

→ Go to IBM

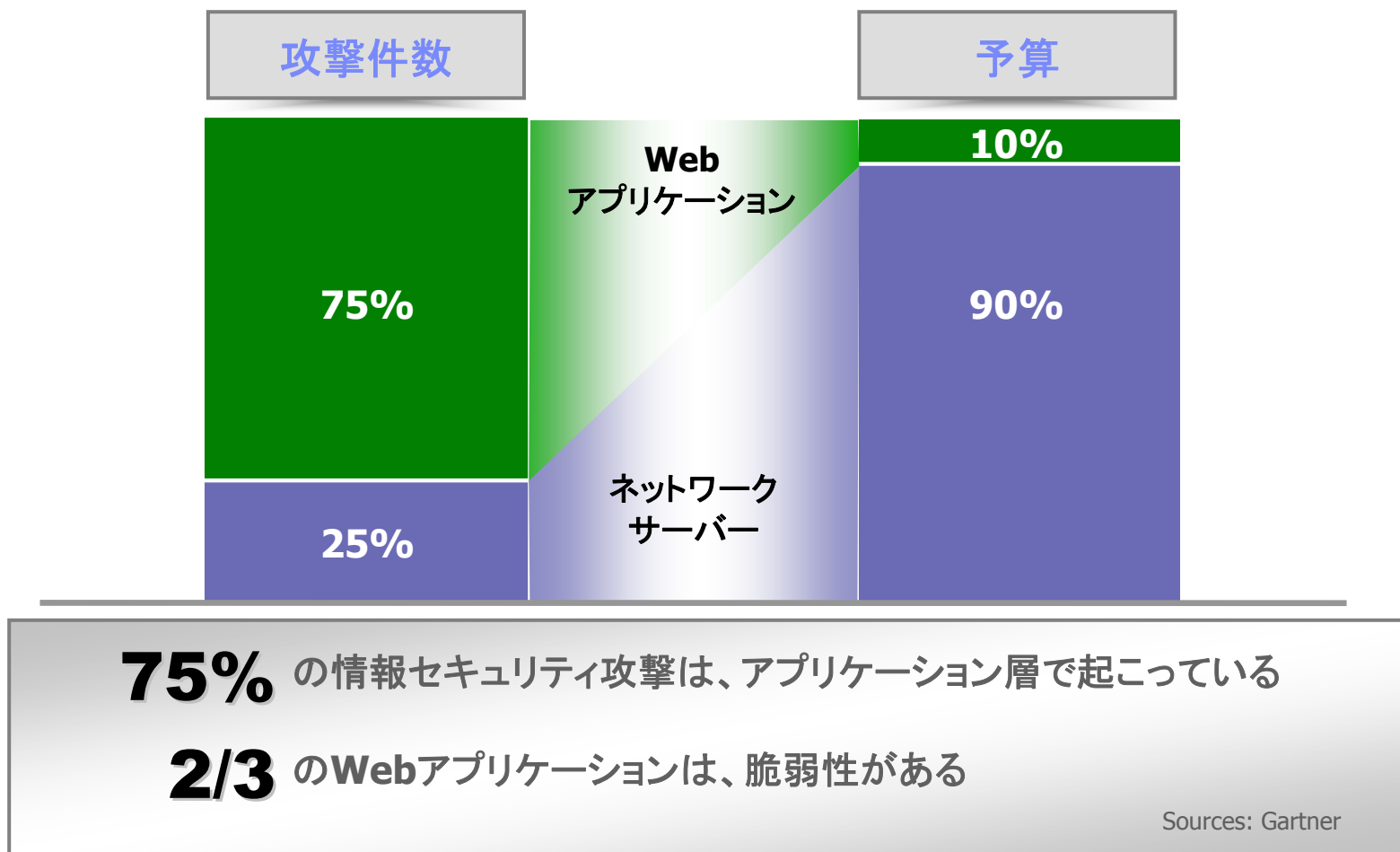
© 2007 IBM Corporation

アプリケーション セキュリティが重要な理由

- Webアプリケーションは、ハッカーの間で最も注目されている
 - ▶ 75% の攻撃はアプリケーション層で見つまっている (Gartner)
 - ▶ XSSとSQLインジェクションは、報告されている脆弱性の#1と#2である (Mitre)
- ほとんどのWebサイトは脆弱である
 - ▶ 90% のWebサイトはアプリケーションの脆弱性がある (Watchfire)
 - ▶ 78% の容易に利用可能な脆弱性は、Webアプリケーションにある (Symantec)
 - ▶ 80% の組織は、2010年までにアプリケーションセキュリティ事件に遭遇する (Gartner)
- Webアプリケーションはハッカーにとって価値のあるターゲット
 - ▶ 顧客データ、クレジットカード、IDの盗難、詐欺、サイトの改竄、など
- コンプライアンスの要求
 - ▶ GLBA、HIPAA、FISMA、PCI データセキュリティ標準、SOX、個人情報保護法



Web アプリケーションに対する攻撃の増加と対策予算



Web アプリケーション開発におけるセキュリティ

64% の開発者は、セキュアなアプリケーションを書く自信が無い

Microsoft Developer Research

70% の企業は、SDLC にセキュアなアプリケーションを書くための技術を導入していない

Aberdeen Group, May 2007

90% のアプリケーションには脆弱性がある

Watchfire



セキュリティ対策の手法

手法		利点	欠点
検査サービス		<ul style="list-style-type: none"> ■ 専門家による詳細な調査 	<ul style="list-style-type: none"> ■ コストが高いため、検査頻度、検査範囲が限られる ■ 運用開始後のテストではサービスに影響も
Web アプリケーション ファイアウォール		<ul style="list-style-type: none"> ■ 既存のアプリケーションに対してしても対応可能 	<ul style="list-style-type: none"> ■ アプリケーションの更新に伴う設定変更が煩雑 ■ ネットワークの経路上に置かれる
脆弱性検査ツール	ホワイトボックス	<ul style="list-style-type: none"> ■ 問題の修正位置が的確に把握できる 	<ul style="list-style-type: none"> ■ ソースコードが必要 ■ 実際には脆弱性でないものも指摘 ■ プラットフォーム依存の問題は指摘できない
	ブラックボックス	<ul style="list-style-type: none"> ■ 現実に即した検査 ■ アプリケーションに関する詳細な知識は不要 	<ul style="list-style-type: none"> ■ アプリケーションが稼動する必要がある ■ 実環境では副作用も





IBM Software Group

AppScan

Rational software

→ Go to **IBM**

© 2007 IBM Corporation

AppScan

- Webアプリケーション セキュリティ テスト ツール
 - ▶ Webアプリケーションの脆弱性と、インフラ(OS、Webサーバー等)の設定ミス、既知の問題を検知
 - ▶ テストを自動化し、手作業に比べて圧倒的な時間とコストの削減が可能
 - ▶ 脆弱性の指摘、修正方法の提示、レポートの作成
 - ▶ ブラックボックス テスト
 - ハッカーの視点でセキュリティ問題を検査 - **"Hacker in the box"**
 - "現実的な" 問題点の指摘 - **"Low hanging fruits"**





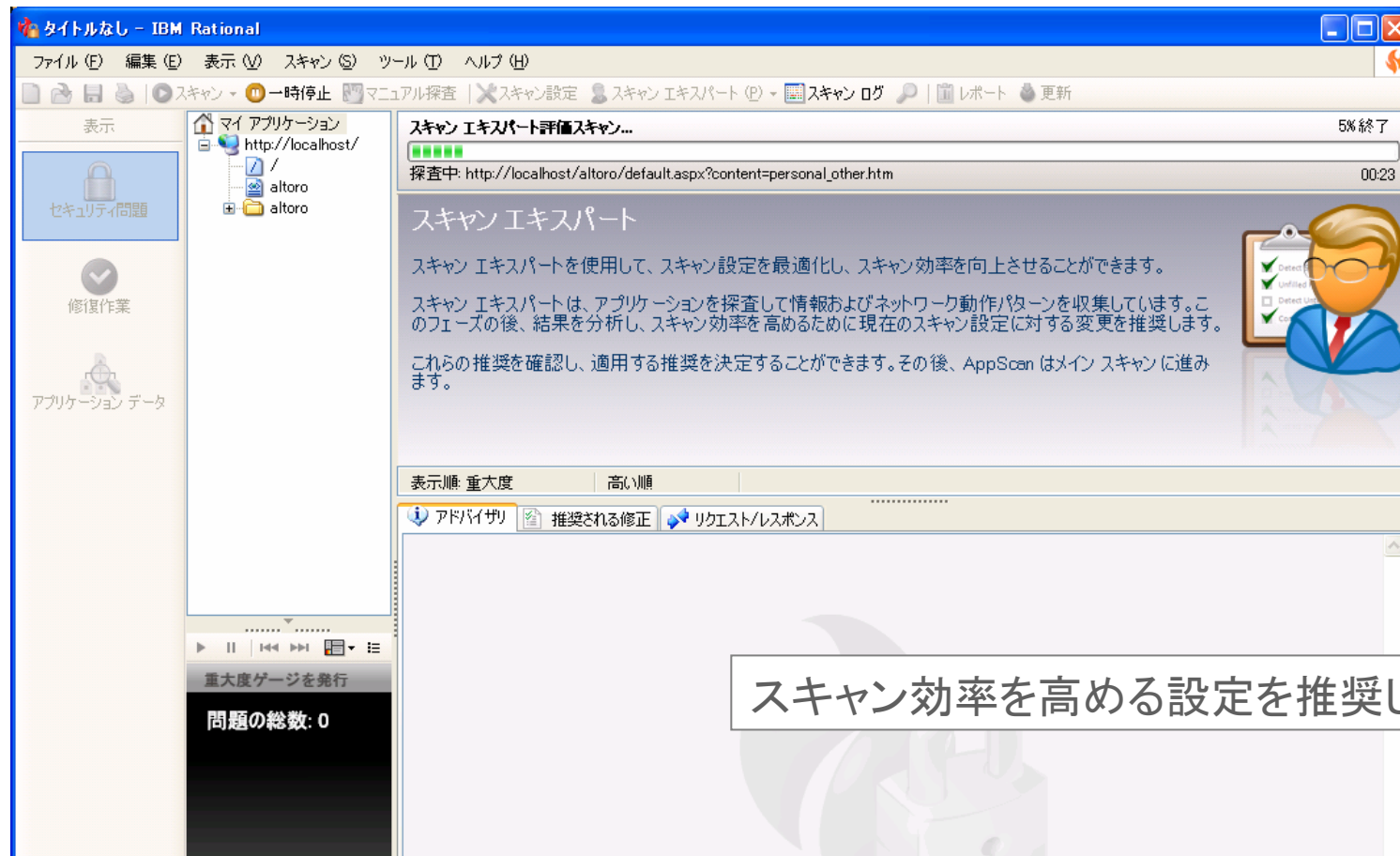
Demo



AppScan 7.7 - ウィザードによる容易な設定



AppScan 7.7 - スキャン エキスパート(New)



スキャン効率を高める設定を推奨します

AppScan 7.7 : 見易い GUI

The screenshot displays the IBM AppScan 7.7 interface. The top menu bar includes options like 'ファイル (F)', '編集 (E)', '表示 (V)', 'スキャン (S)', 'ツール (T)', and 'ヘルプ (H)'. The left sidebar, titled '表示', contains icons for 'セキュリティ問題' (Security Issues), '修復作業' (Remediation), and 'アプリケーション データ' (Application Data). The main area shows a list of security issues for the target 'http://demo.testfire.net/'. The issues are sorted by severity, with '高' (High) at the top. A detailed view of an 'SQL インジェクション' (SQL Injection) issue is shown, including its severity ('高'), type ('アプリケーションレベル テスト'), WASC category ('コマンドの実行: SQLインジェクション'), and CVE reference ('N/A'). A bottom bar provides a summary: '問題の総数: 93' (Total number of issues: 93), with a breakdown by severity: 39 High, 18 Medium, 26 Low, and 10 Informational. The status bar at the bottom indicates '93 セキュリティ問題' (93 Security Issues), '39' (High), '18' (Medium), '26' (Low), and '10' (Informational).

問題が重大度分けして表示される

サイトの構成が一目で分かる

セキュリティアドバイザリが日本語で

AppScan 7.7 - 推奨される修正の提示

The screenshot displays the IBM AppScan 7.7 interface. The main window shows a list of security issues, with 'SQLインジェクション' (SQL Injection) highlighted. A red circle highlights the '推奨される修正' (Recommended Fixes) tab. The interface also shows a sidebar with 'マイ アプリケーション' (My Application) and a bottom status bar indicating 93 security issues.

推奨される修正

SQLインジェクション

推奨される修正

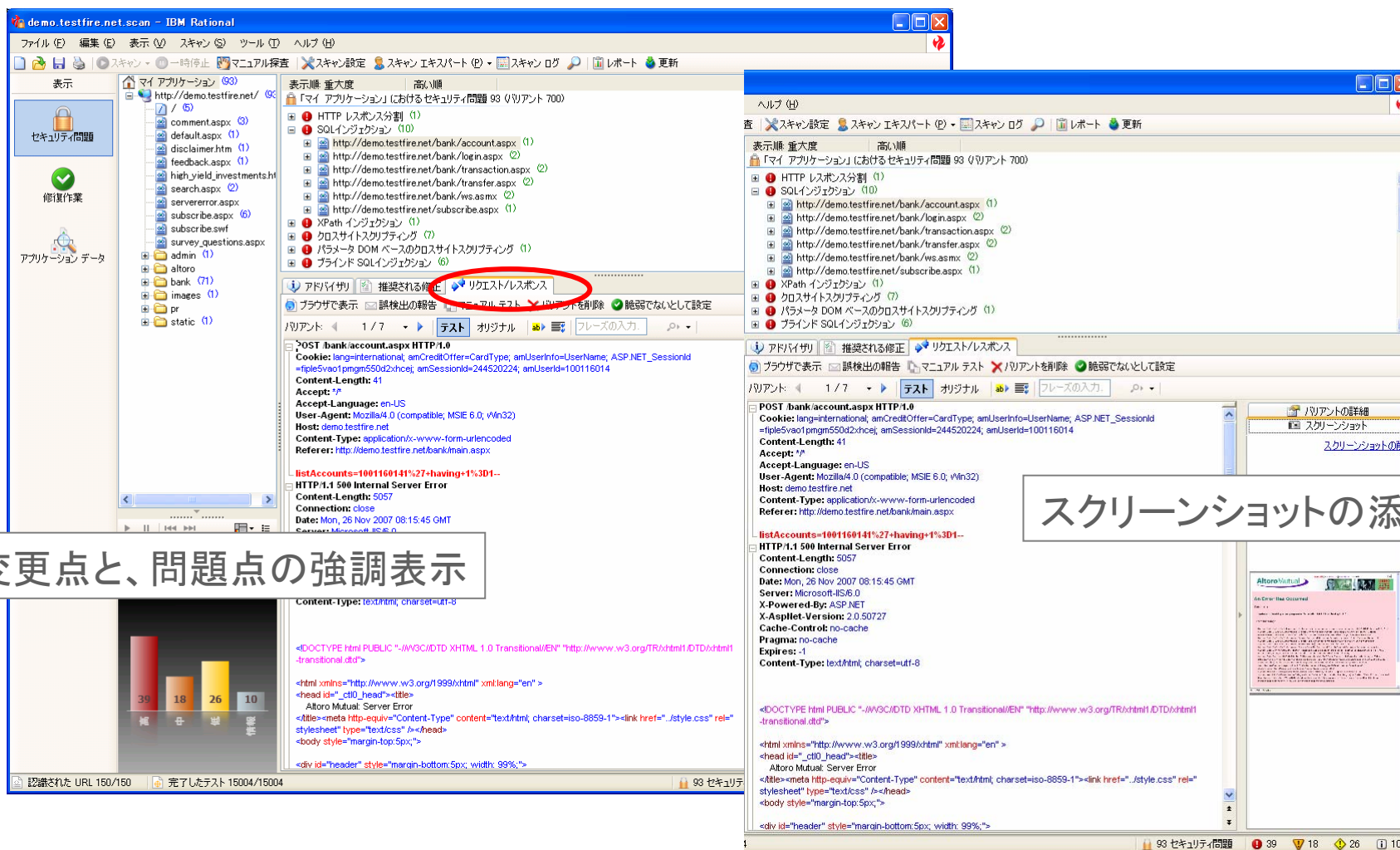
全般

ユーザーの入力のサニタイズによって修復できる問題がいくつかあります。ユーザーの入力に危険性のある文字が含まれていないことを検証することによって、悪意のあるユーザーがアプリケーションに SQL クエリー、クライアント側で実行される Javascript コードの埋め込み、さまざまなオペレーティング システム コマンドの実行など、意図しない動作を行わせるのを防止することができます。

以下の文字はすべてフィルタで取り除いてください:

- [1] | (パイプ)
- [2] & (アンパサンド)
- [3] ; (セミコロン)
- [4] \$ (ドル記号)
- [5] % (パーセント記号)
- [6] @ (アットマーク)
- [7] ' (シングルクォート)
- [8] " (ダブルクォート)
- [9] \ (バックスラッシュ エスケープ シングルクォート)
- [10] \ (バックスラッシュ エスケープ ダブルクォート)
- [11] <> (三角括弧)

.NET、J2EE、PHP に特化した修正方法も提示



AppScan 7.7 - 修復作業

タイトルなし - IBM Rational

ファイル (F) 編集 (E) 表示 (V) スキャン (S) ツール (T) ヘルプ (H)

表示: セキュリティ問題, 修復作業, アプリケーション データ

マイ アプリケーション (61)

http://demo.testfire.net/ (4)

- comment.aspx (2)
- default.aspx (1)
- disclaimer.htm (1)
- feedback.aspx (1)
- search.aspx (1)
- servererror.aspx
- subscribe.aspx (3)
- subscribe.swf
- survey_questions.aspx
- admin (1)
- bank (45)
- images (1)
- pr
- static (1)

表示順: 優先度, 高い順

「マイ アプリケーション」の修復作業 61

- 1 アクセスされるファイルが仮想パスにあり、特定の拡張子を持つようにします。ユーザーの入力から特殊な文字を削除します。 (1)
- 2 クライアント側のコードを分析して、入力ソースをサニタイズします (1)
- 3 ユーザーがサインアウトしたときに、対応するセッション識別子を無効化します (1)
- 4 ユーザーの入力から有害な文字をフィルタで取り除きます (17)
- 5 ログイン証明書をより強力な組み合わせに変更します (1)
- 6 すべてのログインリクエストを暗号化します (1)
- 7 ディレクトリの一覧表示を拒否するようにサーバの設定を変更し、使用できる最新のセキュリティパッチをインストールします (2)
- 8 仮想ディレクトリから不要なファイルを削除します。 (1)
- 9 Atutor の最新バージョンにアップグレードします (1)
- 10 HTML コメントから秘密情報を削除します (3)
- 11 Microsoft ASP.NET 上でのデバッグを無効にします (2)
- 12 Web.Config ファイルを修正して、VIEWSTATE パラメータを暗号化します (3)
- 13 サーバからテストスクリプトを削除します (1)
- 14 パラメータ値が期待される範囲とタイプであることを確認します。デバッグエラーメッセージおよび例外を出力しないようにします。 (16)
- 15 ログインの試行に複数回失敗したアカウントをロックアウトします (1)
- 16 書く ASP.NET ページのプロパティを修正して、VIEWSTATE パラメータを署名します (3)
- 17 禁止されているリソースについて「404 - Not Found」レスポンスステータスコードを送出するか、完全に削除します (3)
- 18 秘密のセッション情報をパーマネント Cookie に保存しないようにします (1)
- 19 秘密情報を送信するときには、必ず HTTP POST メソッドを使用します (2)

ユーザーの入力から有害な文字をフィルタで取り除きます

この修復作業は、以下のようなセキュリティ問題を解決することを想定して設計されています:

- [1] クロスサイトスクリプティング
- [2] 格納されたクロスサイトスクリプティング
- [3] データベースエラーパターンを検出
- [4] SQLインジェクション
- [5] ブラインド SQLインジェクション
- [6] Cookie を汚染する SQLインジェクション
- [7] XPath インジェクション
- [8] HTTP レスポンス分割
- [9] 格納されたレスポンス分割
- [10] ログインページの SQLインジェクション

詳細

ユーザーの入力のサニタイズによって修復できる問題がいくつかあります。ユーザーの入力に危険性のある文字が含まれていないことを検証することによって、悪意のあるユーザーがアプリケーションに SQL クエリー、クライアント側で実行される Javascript コードの埋め込み、さまざまなオペレーティングシステムコマンドの実行など、意図しない動作を行わせるのを防止することができます。

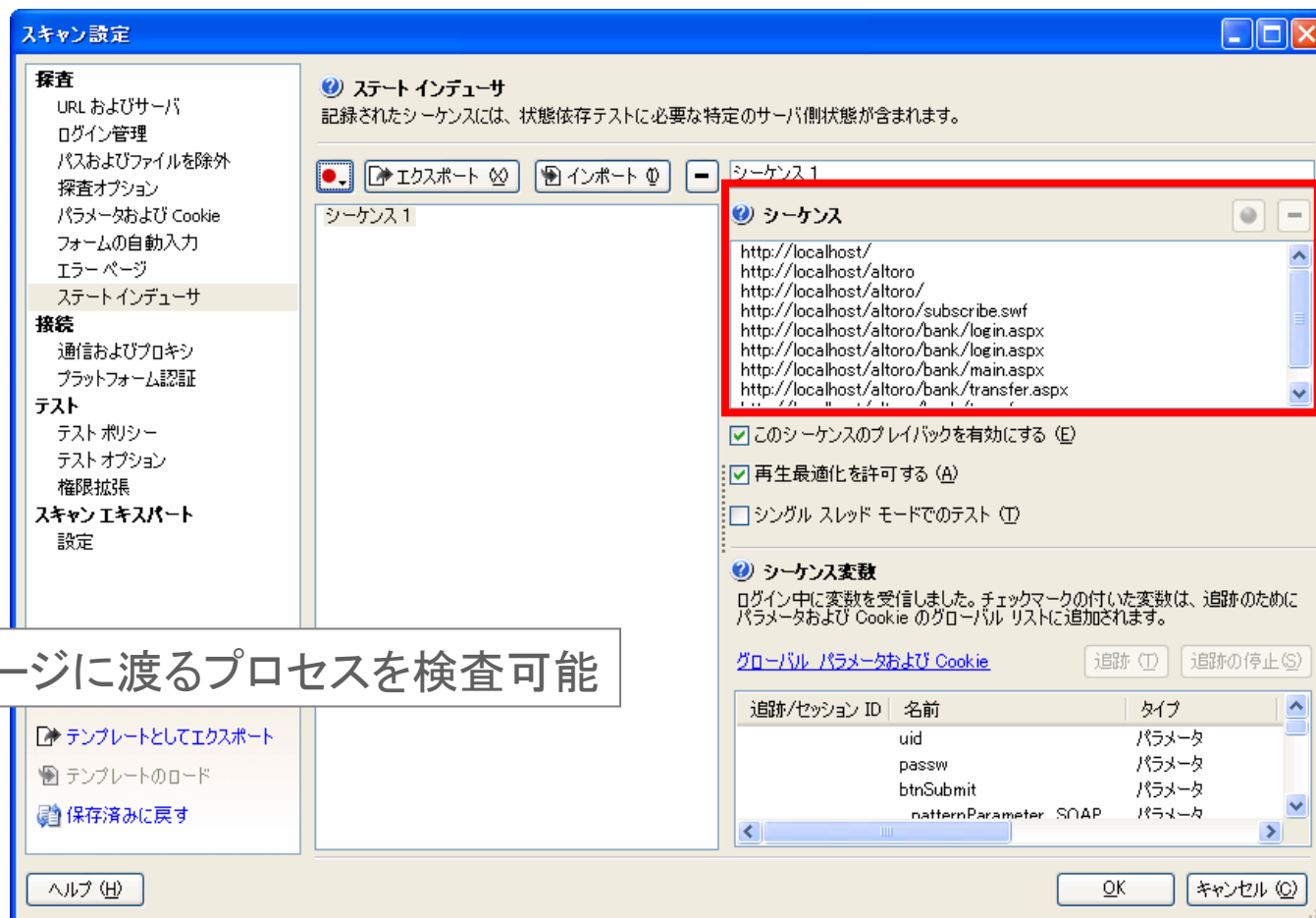
問題の総数: 93

重大度ゲージを発行

認識された URL 144/144 完了したテスト 14817/14817 93 セキュリティ問題 39 18 26 10

修正する人の観点に立った表示

AppScan 7.7 -ステート インデューサー(New)



複数ページに渡るプロセスを検査可能

スクリーンショットの添付



AppScan 7.7 - Word テンプレート(New)

Word テンプレートを使って自由にレイアウト

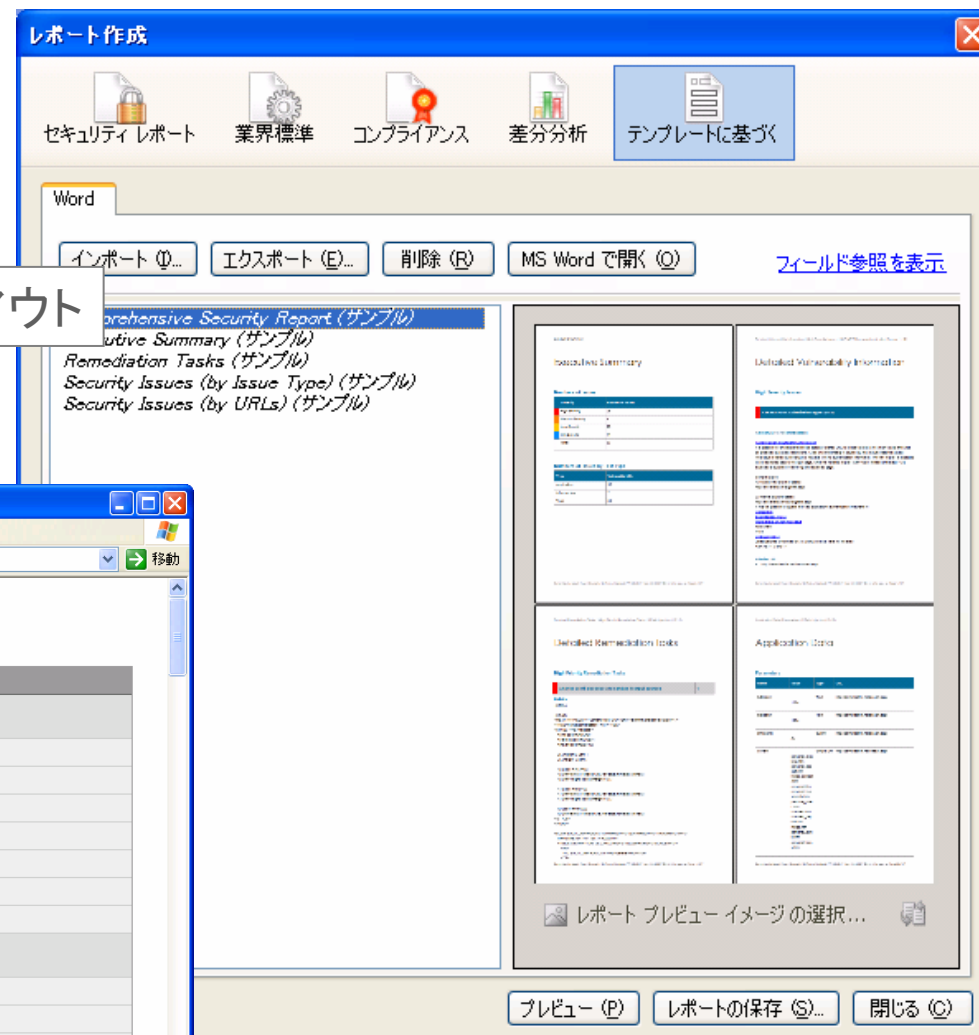
IBM Rational AppScan Custom Word Report Fields - Microsoft Internet Explorer

ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)

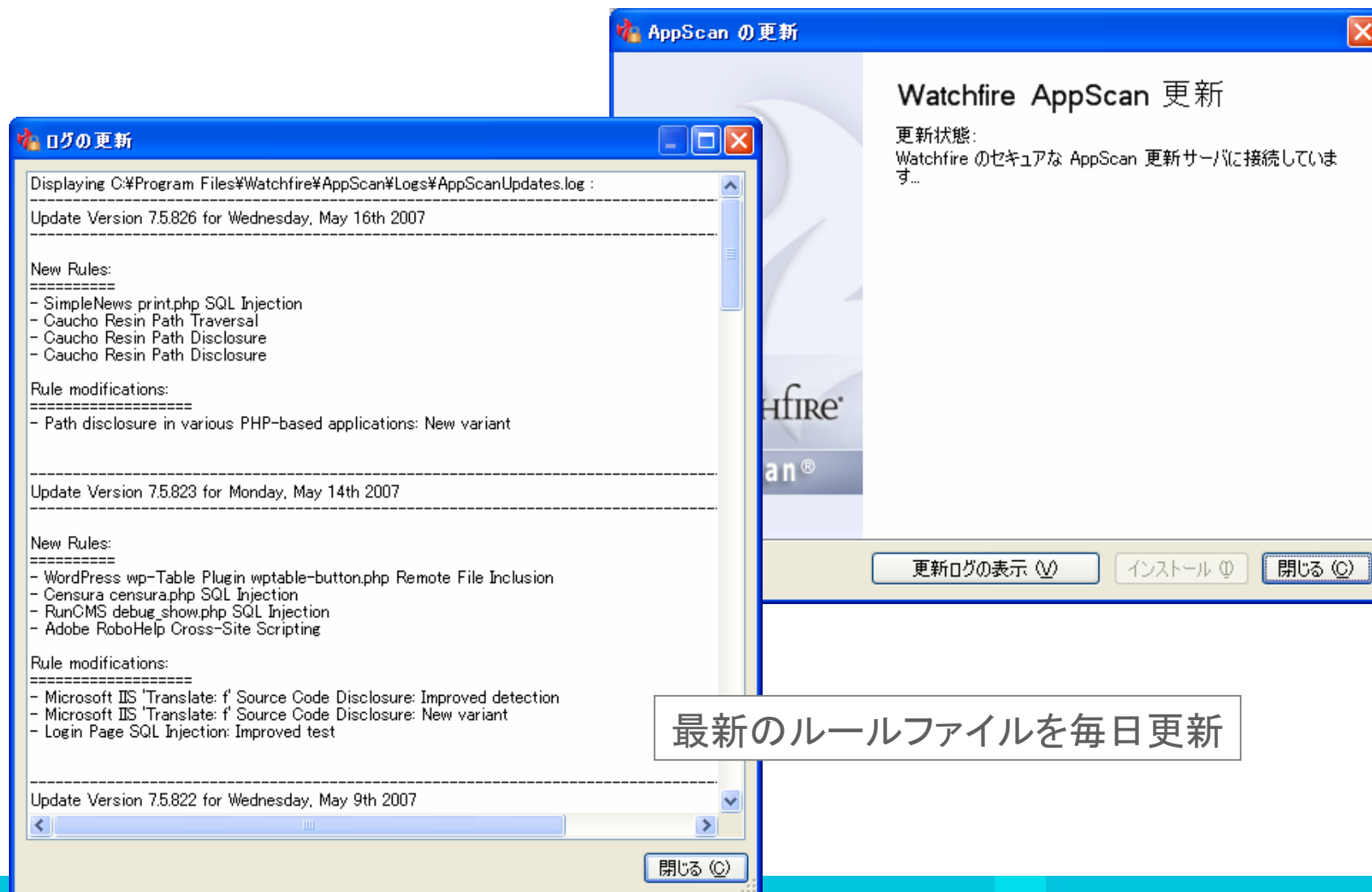
アドレス(AD) O:\Documents and Settings\Amemiya\Local Settings\Temp\WordCustomReportFields.html 移動

IBM Rational AppScan Custom Word Report Fields

Field Name	AppScan Data
Broken URLs	
AS:BrokenURLRepeaterStart	Start BrokenURL repeater
AS:BrokenURLRepeaterIndex	The index of the broken URL
AS:BrokenURLReason	The reason for the broken URL
AS:BrokenURLName	The broken URL name
AS:BrokenURLRequest	The request associated with the URL
AS:BrokenURLCount	The total number of broken URLs
AS:BrokenURLRepeaterEnd	End BrokenURL repeater
Comments	
AS:CommentRepeaterStart	Start Comment repeater
AS:CommentRepeaterIndex	The index of the comment
AS:CommentCount	The total number of comments



AppScan - デイリーアップデート



AppScan 7.7 - レポート一覧

業界標準

- OWASP Top 10 2007
- OWASP Top 10 2004
- SANS Top 20 V5
- SANS Top 20 V6
- WASC Threat Classification
- The Payment Card Industry Data Security Standard (PCI)
- NERC CIPC Electricity Sector Security Guidelines
- International Standard – ISO 17799
- International Standard – ISO 27001
- Visa's Payment Application Best Practices

コンプライアンス

- PIPED Act
- European Directive 1995/46/EC
- European Directive 2005/58/EC
- Japan's Personal Information Protection Act
- Data Protection Act
- California Assembly Bill No. 1950 and Senate Bill 1386
- Children Online Privacy Protection Act (COPPA)
- Electronic Funds and Transfer Act (EFTA)
- Freedom of Information and Protection of Privacy Act (FIPPA)
- Federal Information Security Mgmt. Act (FISMA)
- Financial Services (GLBA)
- Healthcare Services (HIPAA)
- Management of Information Security Technology (MITS)
- NERC Cyber Security Standards
- Privacy Act of 1974
- Safe Harbor
- Sarbanes-Oxley Act (SOX)
- The Securities Act
- Title 21 Code of Federal Regulations
- DCID 6/3 Availability Basic
- DCID 6/3 Availability Medium
- DCID 6/3 Availability High
- DCID 6/3 Confidentiality Reqs Protection Level 1
- DCID 6/3 Confidentiality Reqs Protection Level 2
- DCID 6/3 Confidentiality Reqs Protection Level 3
- DCID 6/3 Confidentiality Reqs Protection Level 4
- DCID 6/3 Confidentiality Reqs Protection Level 5
- DCID 6/3 Integrity Basic
- DCID 6/3 Integrity Medium
- DCID 6/3 Integrity High
- DCID 6/3 Securing Advanced Technology IS
- MasterCard SDP
- Visa CISP
- Basel II
- Family Education Rights and Privacy Act (FERPA)
- NIST Special Publication 800-53



AppScan - Pyscan エクステンション

The screenshot displays the AppScan interface with a scan of a local application at `http://localhost/`. The scan results show 80 security issues. A detailed view of the 'HTTP レスポンス分割' (HTTP Response Splitting) issue is shown, indicating a high severity and application-level test. The WASC (Web Application Security Classification) category is 'クライアント側攻撃' (Client-side attack).

The Pyscan extension window is open, showing a list of detected issues in Japanese, including HTTP Response Splitting, Microsoft ASP.NET Cross-Site Scripting, SQL Injection, and others. The Pyscan console shows the command `>>> printIssues()` and the resulting list of issues.

スクリプト言語から AppScan オブジェクトにアクセス可能 (Requires confirmation)

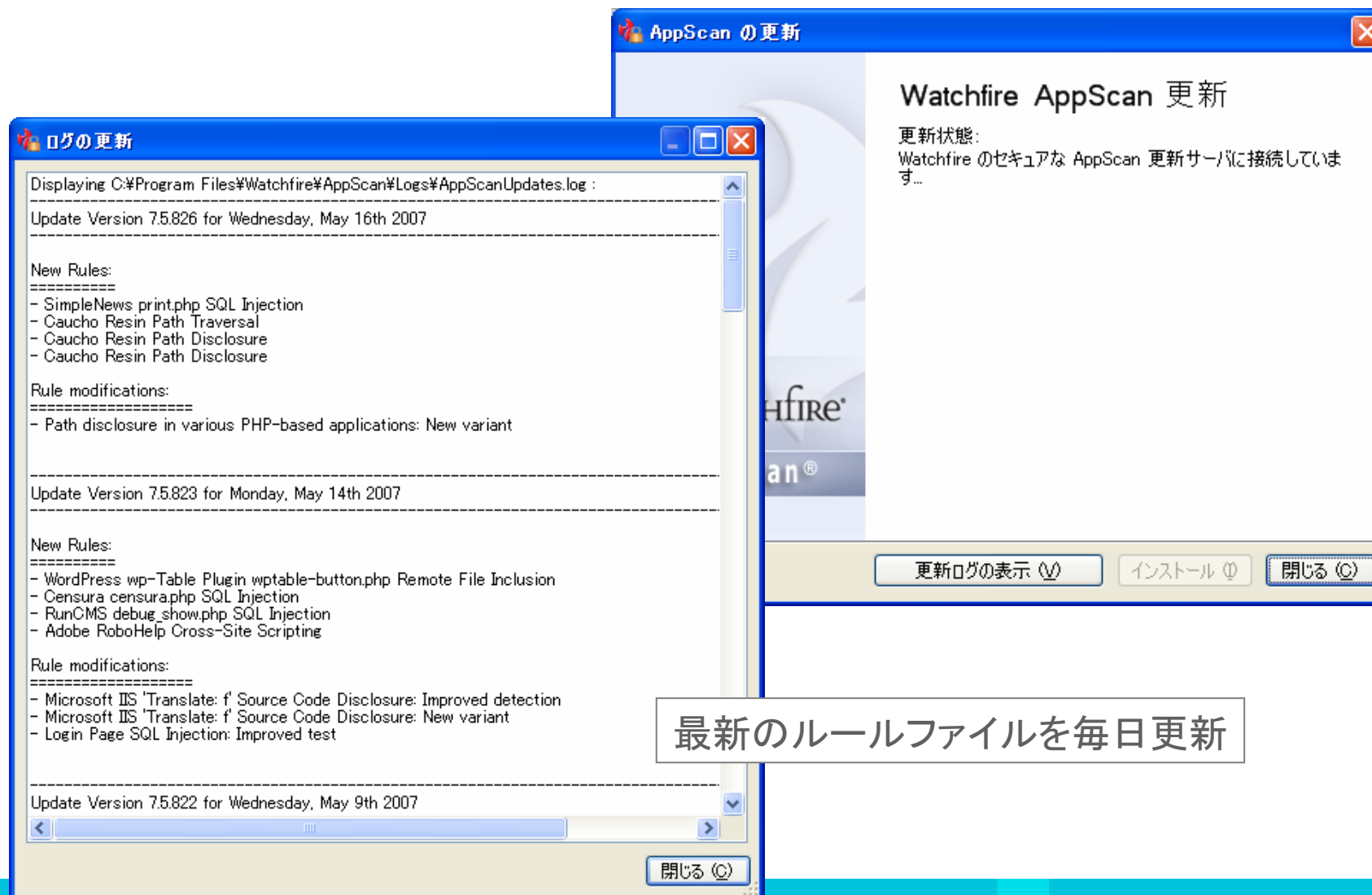
の外観を破壊する
ハッカーが正相の

セキュリティで保護されていない HTTP メソッドが有効
暗号化されていない VIEWSTATE パラメータ
非表示のディレクトリを検出

>>> |

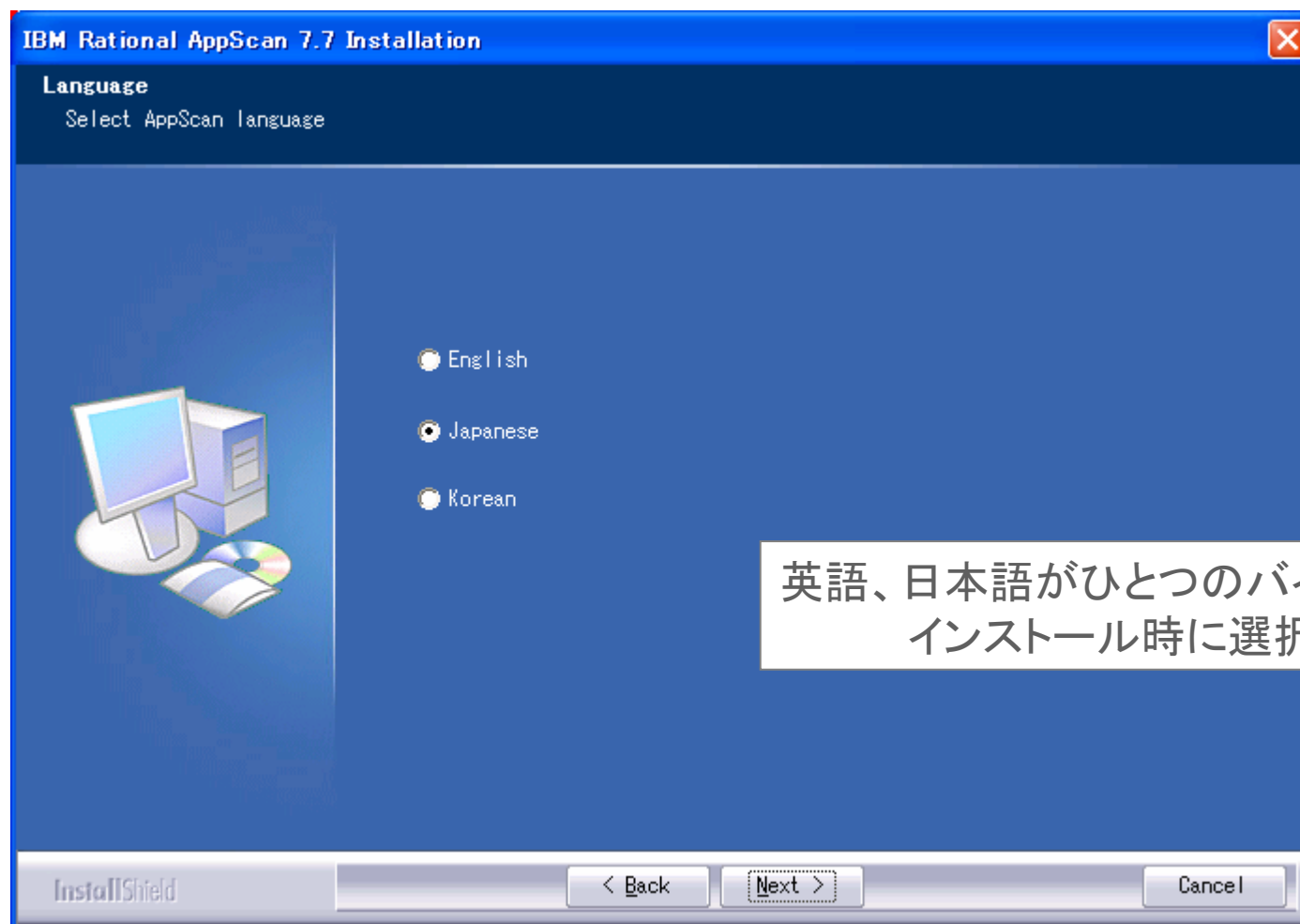
[Ln: 129] [Col: 4]

AppScan - デイリーアップデート



最新のルールファイルを毎日更新

日本語版



英語、日本語がひとつのバイナリに
インストール時に選択

AppScan のマーケット

- 侵入検査サービス
 - ▶ サービスのツールとして
- Sler、アプリケーション開発
 - ▶ 開発段階での検査
 - ▶ 納品時検査
- エンドユーザ
 - ▶ 検収時検査
 - ▶ 自社 Web の定期的な検査



AppScan の利点 - Sler、アプリケーション開発にとって

- セキュリティ検査にかかっていたコストの削減
- 高品質な製品の提供
- 人やプロジェクトごとに異なるセキュリティレベルの標準化
- 開発者から見た視点での問題の指摘
 - ▶ 修復作業ビュー
 - ▶ スクリーンショット付きの詳細なレポート
- セキュリティ専門家でなくても利用可能





IBM Software Group

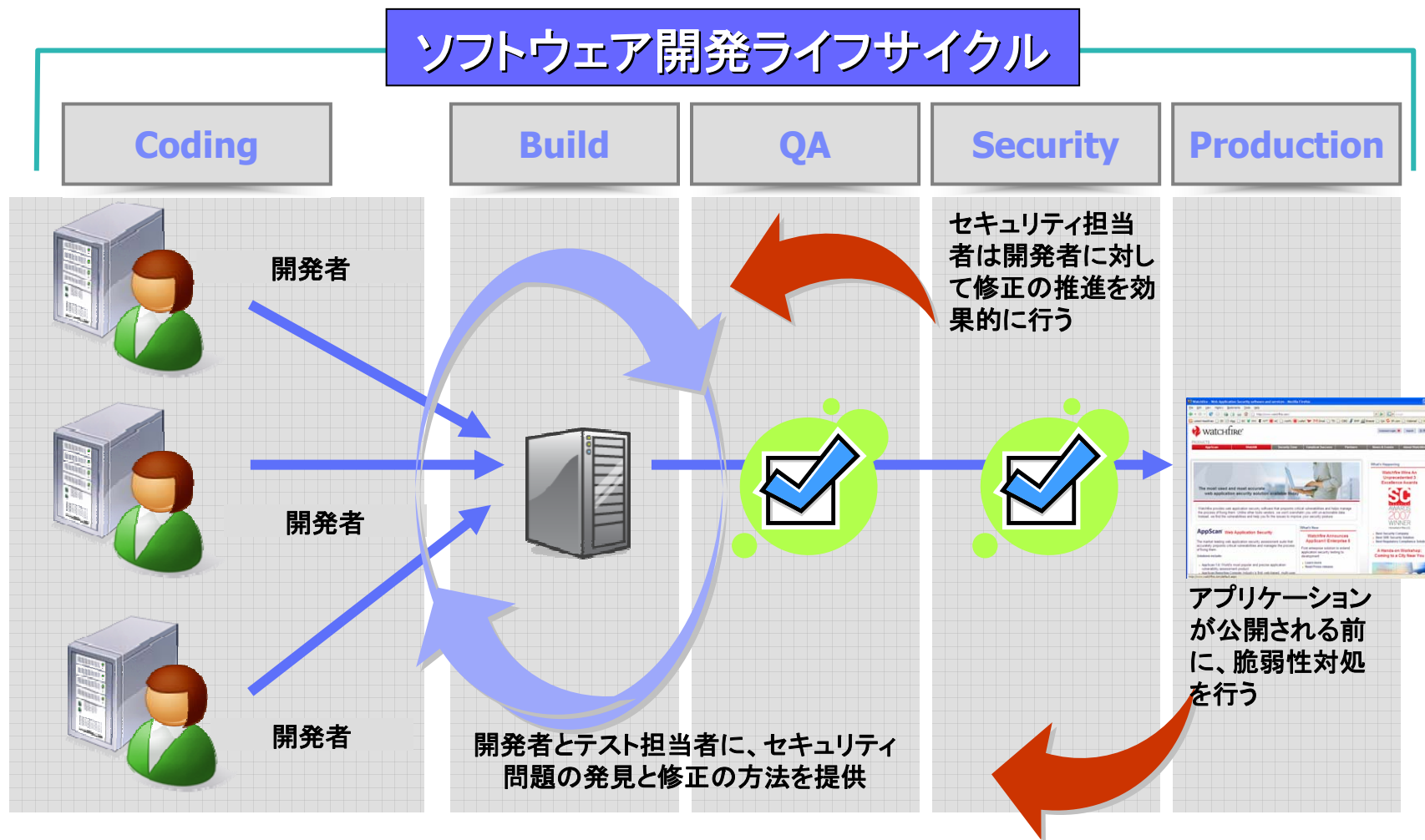
開発工程におけるセキュリティテスト

Rational software

→ Go to IBM

© 2007 IBM Corporation

SDLCにおける Web アプリケーション セキュリティ検査



アプリケーション セキュリティ確立のポイント

1. 人

- 教育
- コミュニケーション
- 責任の所在

2. プロセス

- プロセスの確立
- ゴールの設定

3. 技術

- 全社的に展開可能な技術を選定
- 導入のしやすさ、使いやすさ



ユーザーの声 - 開発者の意識改革

- 「導入当初は、テストの結果により大幅な修正を要求することもあったため、開発者からは嫌がられることもありましたが。しかし現在では、開発者自らがセキュアなコーディングを心がけるようになってきているのを感じます」(ヤマハ様)
- 「スキャンして出てきた問題はレポートでチェックし、システムを修正します。次のシステムの開発時には注意してコーディングするようになり、同じ間違いをすることが減りました」
「我々は多くのお客様の大切な情報を預かっています。システムのどこにも脆弱性があるってはいけません。安全性の高いシステムを構築することが我々の使命であり、仕事のモチベーションです。AppScan導入はセキュリティ品質の向上と同時に、開発者の意識向上のきっかけにもなりました」(東京海上日動システムズ様)

セミナーの御案内

- **2/8(金) : Webアプリケーション脆弱性対策セミナー**
 - ▶ @IBM SWCOC 渋谷
 - ▶ 各種の Web アプリケーション脆弱性対策を解説

- **2/19(火) : Webアプリケーションをハッカーからの攻撃から守る**
 - ▶ @IBM 箱崎事業所
 - ▶ AppScan を使った事例、効果を解説

<http://www.ibm.com/jp/software/rational/events/>





Thank You

