

# ***Understanding DevPartner SecurityChecker®***

Version 2.5



Technical support is available from our Technical Support Hotline or via our FrontLine Support Web site.

Technical Support Hotline:  
1-800-538-7822

FrontLine Support Web Site:  
<http://frontline.compuware.com>

This document and the product referenced in it are subject to the following legends:

Access is limited to authorized users. Use of this product is subject to the terms and conditions of the user's License Agreement with Compuware Corporation.

© 2006 Compuware Corporation. All rights reserved. Unpublished - rights reserved under the Copyright Laws of the United States.

#### U.S. GOVERNMENT RIGHTS

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in Compuware Corporation license agreement and as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii)(OCT 1988), FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as applicable. Compuware Corporation.

This product contains confidential information and trade secrets of Compuware Corporation. Use, disclosure, or reproduction is prohibited without the prior express written permission of Compuware Corporation.

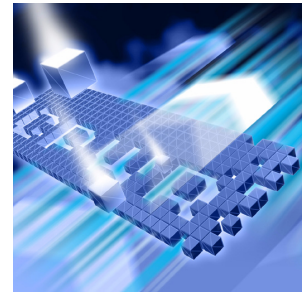
DevPartner and DevPartner SecurityChecker are registered trademarks of Compuware Corporation.

Acrobat<sup>®</sup> Reader copyright © 1987-2006 Adobe Systems Incorporated. All rights reserved. Adobe, Acrobat, and Acrobat Reader are trademarks of Adobe Systems Incorporated.

All other company or product names are trademarks of their respective owners.

US Patent Nos.: 5,987,249 and 6,332,213

June 27, 2006



# Table of Contents

## Preface

What's New In DevPartner SecurityChecker .....	vii
Who Should Read This Manual .....	ix
Conventions Used In this Manual .....	ix
Accessibility .....	ix
For More Information .....	x
Using the InfoCenter .....	x
Customer Assistance .....	x
For Non-technical Issues .....	x
North America .....	xi
International .....	xi
For Technical Issues .....	xi

---

## Chapter 1

### Installation and QuickStart

System Requirements .....	1
Hardware Requirements .....	1
Minimum Requirements - Operating Systems .....	2
Minimum Requirements - Software .....	2
Co-Existence Restrictions .....	3
Installation Considerations .....	3
Installing DevPartner SecurityChecker .....	3
Receiving Your License File .....	3
License Considerations .....	4
Installation .....	5
Installation Note: Proxy Information Dialog Box .....	5
Downloadable Updates .....	6
Quick Start .....	6

---

## Chapter 2

### SecurityChecker Overview

SecurityChecker Overview .....	13
When to Use SecurityChecker .....	15
A Typical Application Development Cycle .....	15
Analysis Types in the Development Cycle .....	15
Compile-time Analysis .....	16
Run-time Analysis .....	16
Integrity Analysis .....	16
Recommended Model for Using SecurityChecker .....	16

---

## Chapter 3

### Discovery and Discovery Maps

Understanding Discovery and Discovery Maps .....	17
Application Discovery .....	17
Manual Discovery - Check only the pages that I visit .....	18
Automatic Discovery - Check all pages in my application .....	18
Using an Existing Discovery Map .....	18
When to Use Manual Discovery .....	18
When to Use Automatic Discovery .....	19
Using Automatic Discovery .....	20
Automatically Creating a Discovery Map .....	21
Errors and Warnings During Discovery .....	23
Errors .....	23
Warnings .....	23
Adjusting Automatic Discovery Settings .....	24
Using the BlackList File .....	25
When to Use the BlackList .....	25
Using Saved Discovery Maps .....	26
Reusing a Discovery Map .....	26
Editing a Discovery Map .....	26
Renaming a Solution After Saving a Discovery Map .....	28
Troubleshooting the Limitations of Discovery .....	28
Limitations of Automatic Discovery .....	28
Multipart HTTP Requests .....	29
Executing Javascript .....	29
Activating Controls .....	29
Database Modification .....	29
Client Side Issues .....	29
Limitations of Manual Discovery .....	30
HTTP 1.0 Protocol .....	30

Multipart HTTP Requests . . . . .	30
Encryption Issues Using Manual Discovery - Secure Socket Layers and HTTPS Pages . . . . .	30
Integrated Windows Authentication . . . . .	31
File Types . . . . .	31
Client Side Issues . . . . .	31

---

## Chapter 4

### Analysis and Results

Analyzing an Application . . . . .	33
Summary Tab . . . . .	33
Classification of Vulnerabilities on the Summary Tab . . . . .	35
Severities . . . . .	35
Critical . . . . .	35
Important . . . . .	35
Moderate . . . . .	35
Informational . . . . .	35
Categories . . . . .	36
Security Context . . . . .	36
Insecure Coding Practices . . . . .	36
Execution Errors . . . . .	36
Application Integrity . . . . .	36
Deployment Issues . . . . .	37
Vulnerabilities Tab . . . . .	37
Details Pane . . . . .	38
Explanation Tab . . . . .	38
Details Tab . . . . .	38
Security Context Tab . . . . .	39
Call Stack Tab . . . . .	39
Interpreting Analysis Results . . . . .	40
Responding to the Analysis Results . . . . .	40
Evaluating the Results . . . . .	40
Triage the Results . . . . .	41
Using the Information in the Details pane . . . . .	41
Repair Repeated Vulnerabilities . . . . .	41
Reports . . . . .	42
Using Reports . . . . .	42
<b>Summary</b> Reports . . . . .	43
<b>Detailed</b> Reports . . . . .	43
Comprehensive Report . . . . .	43
Modifying Report Templates . . . . .	43
Example . . . . .	43
Using a Custom XSLT to Create a Report . . . . .	45

Troubleshooting .....	46
SQL Injection Attacks May Lower Integrity Vulnerability Counts .....	46
Using Reports in Netscape .....	47

---

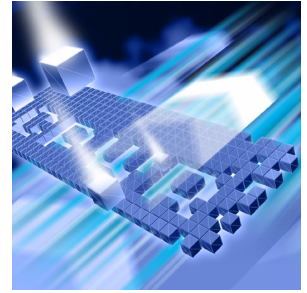
## Chapter 5

### Tailored Analysis

Structure of an ASP.NET Application .....	49
Logical Division of the Application .....	50
Tailoring Analysis to your Application .....	51
Compile-time Analysis .....	51
Run-time Analysis .....	53
Integrity Analysis .....	53
Rules and Vulnerabilities .....	54
Rule Selection .....	54
Suppressions .....	56
Suppression Types .....	56
Rule Level Suppressions .....	56
Location Level Suppressions .....	56
What is a “False Positive”? .....	57
Suppressing a Vulnerability .....	57
Quality Assurance and Automated Testing .....	57
What Quality Assurance Needs for Validation Testing .....	58
SecurityChecker Settings and Discovery Map Files .....	58
Naming Conventions .....	58
Using the Command Line Interface in an Automated Test Environment ...	59
Recording Benchmarks .....	59
Running a Session from the Command Line .....	60
Command File .....	60
Creating Multiple Settings Files .....	61
Creating Multiple Discovery Maps .....	62
Troubleshooting .....	62
Warnings in the Application Events Log .....	62
Inaccurate Module and File information .....	63

### Index

# Preface



- ◆ What's New In DevPartner SecurityChecker
- ◆ Who Should Read This Manual
- ◆ Conventions Used In This Manual
- ◆ Accessibility
- ◆ For More Information
- ◆ Customer Assistance

## What's New In DevPartner SecurityChecker

For Version 2.5, DevPartner SecurityChecker has added:

- ◆ **Downloadable Updates:** To keep you up to date on the latest security threats, SecurityChecker will regularly provide new rule updates. You can check for new updates at any time using the **Check for Updates** selection from the SecurityChecker menu. DevPartner SecurityChecker will also periodically initiate an automatic check for new rule information. When an update is available, SecurityChecker informs you and offers to install the update on your computer.
- ◆ **New Vulnerability Rules:** DevPartner SecurityChecker has added new Security Rules to all three analysis types – Compile-time, Run-time, and Integrity Analysis.
- ◆ **Enhanced Reporting Capabilities:**
  - ◇ Three Summary reports are available: Vulnerabilities organized according to the OWASP Top Ten, vulnerabilities organized according to Industry Classification, and vulnerabilities organized according to Category and Severity.

- ◇ Four types of Detailed reports are available: OWASP Top Ten, Industry Category, Severity, and Vulnerability Category. Reports contain a short description of each vulnerability discovered.
- ◇ A Comprehensive report is available and includes complete session data.
- ◇ Users can edit any of the default report templates to create a customized report.
- ◇ The New Reports dialog box provides an intuitive, easy to use interface.
- ◆ **Terminal Services Support:** All SecurityChecker features (running an analysis, reviewing session results, generating reports, etc.) can be used from a Terminal Services session.
- ◆ **Enhanced File Manipulation Capabilities:**
  - ◇ Settings files can be imported to a new session, and exported from a completed session. Once a session has completed, a link is available on the Summary Tab to save the current settings file.
  - ◇ Command file creation has been automated. Use the Create Command file selection from the SecurityChecker menu to automatically generate a sample command file based on the current session. Command files are used during a command line session, and were previously manually created.
- ◆ **Team System Integration:** DevPartner SecurityChecker integrates with Microsoft Team System. Defect reports are pre-populated with information about the specific security vulnerabilities detected by SecurityChecker.
- ◆ **Large Font Support:** The SecurityChecker user interface has been enhanced to support large fonts.
- ◆ **Rule Explanations in the Online Help:** DevPartner SecurityChecker now lists all the Vulnerability rules in the online help.



## Who Should Read This Manual

This manual is intended for software developers and quality assurance engineers who will be installing and using DevPartner SecurityChecker. This manual introduces the concepts and procedures basic to using SecurityChecker.

This manual assumes that you are familiar with the Microsoft Windows interface, Microsoft Visual Studio 2003 and 2005, and the installation of Windows software.

## Conventions Used In this Manual

This manual uses the following conventions to present information.

- ◆ Screen commands and menu names appear in a **bold typeface**. For example:

Choose **Start Analysis** from the **SecurityChecker** menu.

- ◆ File names and paths appear in a monospace bold typeface. For example:

The *Understanding DevPartner SecurityChecker* manual (Understanding DevPartner SecurityChecker.pdf) describes...

- ◆ Computer commands, variables within computer commands, and file names (for which you must supply values appropriate for your installation) appear in a monospace typeface. For example:

Enter `dpsc /help:option` at the command line prompt...

## Accessibility

Prompted by federal legislation introduced in 1998, and Section 508 of the U.S. Rehabilitation Act, enacted in 2001, Compuware launched an accessibility initiative to make its products accessible to all users, including people with disabilities. This initiative addresses the special needs of users with sight, hearing, cognitive, or mobility impairments.

Section 508 requires that all electronic and information technology developed, procured, maintained, or used by the U.S. Federal government be accessible to individuals with disabilities. To that end, the World Wide Web Consortium (W3C) Web Accessibility Initiative (WAI) has created a workable standard for online content.

Compuware supports this initiative by committing to make its applications and online help documentation comply with these standards. For more information, refer to:

- ◆ W3C Web Accessibility Initiative (WAI) at [www.W3.org/WAI](http://www.W3.org/WAI)
- ◆ Section 508 Standards at [www.section508.gov](http://www.section508.gov)
- ◆ Microsoft Accessibility Technology for Everyone at [www.microsoft.com/enable/](http://www.microsoft.com/enable/)

## For More Information

Use the following resources to learn more about SecurityChecker components.

- ◆ Your DevPartner SecurityChecker CD contains a copy of this manual, in Adobe Acrobat (.pdf) format.
- ◆ Read the Release Notes for late-breaking information, including details on any known anomalies in this release.
- ◆ Learn about licensing in the *Distributed License Management: Installation Guide*.
- ◆ Use the SecurityChecker integrated Visual Studio Help as you work with SecurityChecker within the Visual Studio environment. Access the SecurityChecker online help from the Help menu (**Help > Contents**). When the Contents pane appears, select DevPartner SecurityChecker.

## Using the InfoCenter

The **InfoCenter** provides direct links to various Compuware help and support sources, such as the SecurityChecker online help, PDF manuals, and Release Notes. **InfoCenter** is located under **Programs > Compuware DevPartner SecurityChecker** on the **Start** menu.

## Customer Assistance

The following provides information on how to access customer assistance for both non-technical and technical issues.

### For Non-technical Issues

Contact Customer Service for answers to your questions regarding upgrades and other order fulfillment needs. You can reach Customer Service from 8:30 AM to 5:30 PM EST, Monday through Friday.

## North America

- ◆ 603 578 8400
- ◆ Toll free: 800 468 6342

## International

- ◆ Europe: 00 800 6863 4248
- ◆ Finland: 990 800 6863 4248
- ◆ Israel: 014 800 6863 4248

All other international customers, check Compuware Worldwide Offices for the local phone number.

## For Technical Issues

Technical Support can assist you with all your technical problems related to SecurityChecker, from installation to troubleshooting.

Before contacting Technical Support, please read the relevant sections of the product documentation and the Release Notes.

You can contact Technical Support by:

E-Mail	Send as many details as possible to <a href="mailto:nashua.support@compuware.com">nashua.support@compuware.com</a>
World Wide Web	Submit issues and access our support knowledge base at <a href="http://frontline.compuware.com/nashua/">http://frontline.compuware.com/nashua/</a>
Telephone	Telephone support is available as a paid* Priority Support Service from 8:30 am to 5:30 pm EST, Monday through Friday. Have product version information ready. In the U.S. and Canada, call: 888 686 3427 International customers, call: +1 603 578 8100 * Technical Support handles installation and setup issues free of charge.
Fax	Send as many details as possible to 603 578 8401.

Before contacting Technical Support, please obtain and record the following information about your computer system:

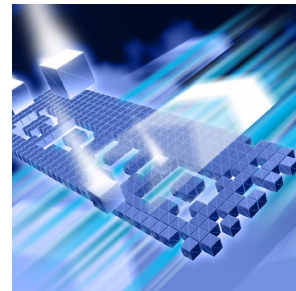
- ◆ Product name (or edition), and version
- ◆ Operating system: name, service pack, language variant
- ◆ Hardware configuration: processor speed, number of processors, amount of RAM, and display resolution
- ◆ Software environment: Visual Studio version, .NET Framework version, anti-virus software, and software firewall, if used

Technical Support will also need the following information about your problem:

- ◆ Any error messages displayed
- ◆ Stack dumps, exceptions, or other similar details
- ◆ SecurityChecker files, such as log or settings files
- ◆ Description of the application being analyzed
- ◆ Details about where in your application the problem occurred
- ◆ If possible, a small sample of the application

# Chapter 1

## Installation and QuickStart



- ◆ System Requirements
- ◆ Installing DevPartner SecurityChecker
- ◆ Downloadable Updates
- ◆ Quick Start

This chapter covers installation of SecurityChecker, and provides QuickStart information to get you up and running.

### System Requirements

This section covers installation requirements.

### Hardware Requirements

For best performance, run SecurityChecker on a system that meets or exceeds the recommended hardware configuration.

Table 1-1. Hardware Recommendations

Hardware	Minimum	Recommended
Pentium III	733 MHz	1.5+ GHz
Memory	512 MB	1 GB
Available Disk Space	500MB	500MB
Display	1024x768, 16-bit color	N/A
Other	CD Drive	N/A

**Note:** Although SecurityChecker can be installed anywhere, approximately 260 MB of space will be consumed on your system partition. If insufficient space is available, the installation will fail. Please make sure that there is at least 260 MB of storage available on the system partition prior to installation.

## Minimum Requirements - Operating Systems

The SecurityChecker software has been tested on, and supports the following operating systems.

Table 1-2. Minimum Requirements - Operating Systems

Operating System	Editions	Browser and IIS
Windows XP with SP2 (32-bit only)	Professional, Tablet PC	IE 5.5 IIS 5.1
Windows Server 2003 with SP1 and R2 (32-bit only)	Standard, Enterprise, and Web	IE 6.0 IIS 6.0

## Minimum Requirements - Software

The SecurityChecker software has been tested on, and integrates with the following Visual Studio versions:

Table 1-3. Minimum Requirements - Software

Version	Editions	.NET Framework	Languages
Visual Studio 2005	Professional Edition Team Edition for: <ul style="list-style-type: none"> <li>• Software Architects</li> <li>• Software Developers</li> <li>• Software Testers</li> </ul> Team Suite Tools for Office	.NET Framework 2.0	Visual C# .NET Visual Basic .NET
Visual Studio 2003 .NET	<ul style="list-style-type: none"> <li>• Developer</li> <li>• Professional</li> <li>• Enterprise Architect</li> </ul>	.NET Framework 1.1 (SP1) .NET Framework 2.0	Visual C# .NET Visual Basic .NET

## Co-Existence Restrictions

If you have a previous version of DevPartner SecurityChecker currently installed, it must be uninstalled prior to installing DevPartner SecurityChecker 2.5.

The SecurityChecker software supports co-existence with DevPartner Studio 8.0 and later, and DevPartner Fault Simulator 1.5 and later.

## Installation Considerations

Please note the following items when installing SecurityChecker:

- ◆ You must have administrative privileges on your machine to install SecurityChecker.
- ◆ The SecurityChecker software will not run on Microsoft .NET Framework 1.0. You must have at least Framework 1.1 installed.
- ◆ The SecurityChecker software will only analyze ASP.NET 1.1 and 2.0 applications.
- ◆ ASP.NET applications being analyzed must be on the local machine.
- ◆ IIS must be installed.
- ◆ If you will be using Visual Studio Team System to submit work items, Visual Studio Team Foundation Server and Team Explorer must be installed. The Work Item template must include “Bug” type.

## Installing DevPartner SecurityChecker

Installation is an automatic process. The SecurityChecker software detects your system configuration and installs the applicable software.

## Receiving Your License File

The SecurityChecker software requires a license file supplied by Compuware in order to execute beyond a single 14-day evaluation period. A license file is provided for each license that you purchase.

Your license file(s) will arrive by e-mail around the time that your product arrives, and is delivered to the person designated when the order was placed.

**Note:** If you purchased SecurityChecker from a reseller, you can request a license file from the Compuware Web site at [http://frontline.compuware.com/sw/license\\_default.asp](http://frontline.compuware.com/sw/license_default.asp), or by calling Worldwide License Management at 1-800-538-7822.

Install the license file you receive from Compuware by running the License Administration Utility (**Start | Programs | Compuware | License Administration | License Administration Utility**).

If the license file has not arrived when you are ready to install SecurityChecker, please contact Worldwide License Management: 1-800-538-7822. Outside the USA and Canada, please contact your local Compuware office or agent.

Complete information about installing and managing licenses can be found in *Distributed License Management License Installation Guide* which is accessible from the InfoCenter (**Start | Programs | Compuware DevPartner SecurityChecker | InfoCenter**), or on the product CD (... \Setup \Common \Compuware \LicInst4.pdf).

You may choose to store the SecurityChecker license in the same location as your other Compuware products. In this case, we recommend that you rename the SecurityChecker license file to prevent conflicts, accidental merging, or replacing of license files.

If you already have a licensed Compuware product and would like to merge the license files, contact Technical Support (1-888-686-3427).

## License Considerations

All DevPartner products use the Compuware Distributed License Management software to manage software licenses. You should be aware of the following considerations regarding your SecurityChecker software license:

- ◆ The SecurityChecker software can run as a 14-day evaluation, or one of two types of permanent license:
  - ◆ A **concurrent** license stored within a License Manager database on a network server
  - ◆ A **borrowed concurrent** license running on a computer disconnected from a network

The SecurityChecker software uses all license paths entered during installation to verify license information. If you experience lengthy SecurityChecker start-up delays, check your license path for invalid entries.

**Note:** Concurrent licenses require Compuware Distributed License Management (DLM 4.0). For more information on license installation, refer to the *Distributed License Management License Installation Guide*.



## Installation

To install DevPartner SecurityChecker:

- 1** Insert the product CD into your CD-ROM drive.  
If you have autorun enabled, the setup runs automatically. If not, open the **Add or Remove Programs** control panel, click **Add New Programs**, and then click **CD or Floppy**.
- Note:** If you receive a message from the virus protection software installed on your system, disable that software until after completion of the SecurityChecker installation.
- 2** When the setup screen appears, click **Install Compuware DevPartner SecurityChecker**.
- 3** Click **Install DevPartner SecurityChecker** and follow the on-screen instructions in the installation wizard.  
When installation is complete, you are returned to the **Options** screen.
- 4** Click **Configure a License** to enter your license file information. See [“Receiving Your License File”](#) on page 3 for more information.
- Note:** You can skip this step and default to a 14-day evaluation.
- 5** Click **Register Your Product** to complete the online product registration.

### **Installation Note: Proxy Information Dialog Box**

The SecurityChecker installation automatically detects the proxy server settings on your local machine. For applications running only on the local machine, this information will be sufficient.

If you are using a proxy server on your local machine that requires authentication information, you can choose to enter it here. The information can also be entered during an analysis session. If you enter the authentication information at install, you will not need to provide it for individual sessions.

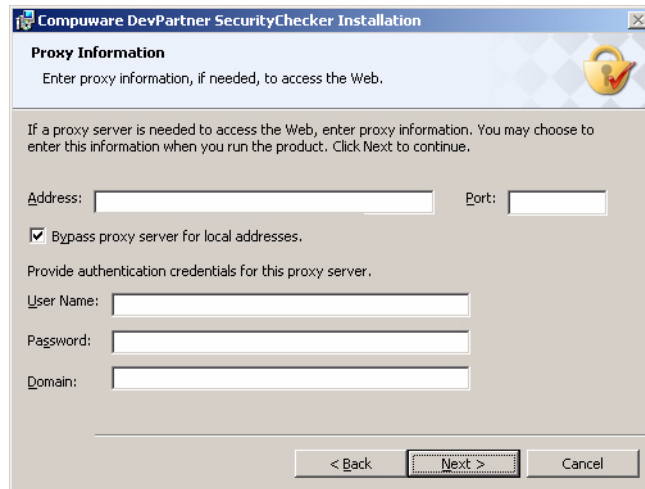


Figure 1-1. Proxy Information Dialog Box

## Downloadable Updates

To keep you up to date on the latest security threats, SecurityChecker will regularly provide new rule updates. You can check for new updates at any time using the **Check for Updates** selection from the SecurityChecker menu. SecurityChecker will also periodically initiate an automatic check for new rule information. When a rules update is available, SecurityChecker will display information regarding the update, as well as a link for additional details. You can choose to download and install the update at that time, or initiate a manual check later.

For additional information about configuring the Downloadable Updates service, refer to the online help.

## Quick Start

The SecurityChecker software automatically analyzes and identifies security vulnerabilities in your ASP.NET application. Through a combination of Run-time, Compile-time, and Integrity analysis, SecurityChecker pinpoints the location of security vulnerabilities, and then offers explanations and repair procedures to fortify your application.

This section explains how to run a default analysis of your ASP.NET application. In this section, you will lead SecurityChecker through your application, analyzing only the pages you choose. Then you will drill down into the analysis results, viewing detailed information about vulnerabilities and how to repair them.


- 1 Open Visual Studio.
- 2 Open the solution to be analyzed.
- 3 Click the **New SecurityChecker Session** icon  on the toolbar, or select **New Session** from the **SecurityChecker** menu. The SecurityChecker application opens in the IDE, and the **QuickStart** tab appears.



Figure 1-2. QuickStart Tab

The first-time default settings include all three types of analysis, and **Check only pages that I visit**.

- 4 In the **Specify start page** text box, you can specify a start page for your application. Enter the full URI (uniform resource indicator) to the page on your local system, for example, `http://localhost/BNTNET/home.aspx`.

If you have set a start page in the solution, you can leave the field blank and SecurityChecker will default to the solution setting.

The **Modify SecurityChecker Settings and Suppressions** link opens the Settings dialog box. The settings enable you to view and change parameters that SecurityChecker uses as you run an analysis session.

**Note:** For detailed information about each of the pages in the Settings dialog box, refer to the *Settings Dialog Box* topic under *User Interface* in the SecurityChecker online help.

**5** Click **Start Analysis**.

The Manual Discovery Information dialog box appears. Read this information carefully.

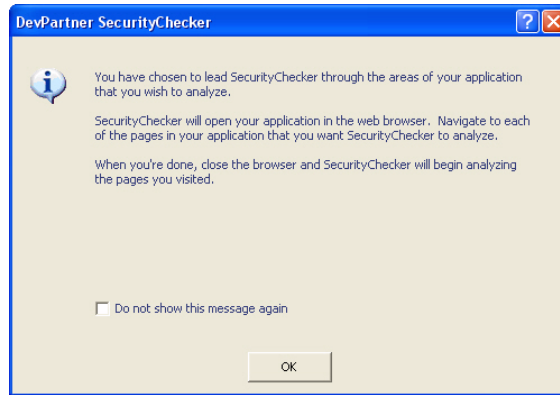


Figure 1-3. Manual Discovery Information

---

**Caution:** If your application will interact with a database during a SecurityChecker analysis session, create a reset script to return the database to its original state. Run this script immediately after an analysis session. Or, run SecurityChecker against a copy of the database.

During the analysis phase of a SecurityChecker session, SecurityChecker follows the map created in the discovery phase, repeating all of the mapped actions. If these actions touch your database, SecurityChecker will attempt attacks on those entry points, resulting in changes to your database. This behavior is normal for a product like SecurityChecker.

---

**6** Click **OK**.

The SecurityChecker software performs the following operations:

- ◇ Builds your application in Visual Studio if necessary
- ◇ Opens your application in a browser window

- 7 Navigate to each of the pages in your application that you want to analyze. To validate actions such as database access, or links to other applications, enter the necessary data in the required fields.

As you work, SecurityChecker lists each page you access in the **Actions** list on the **Discovery** tab. Pages with data fields appear in bold text.

*Tip: If this is your first time running SecurityChecker, you may want to only visit two or three pages of your application. Longer discovery maps increase analysis time, especially when Integrity analysis is selected.*

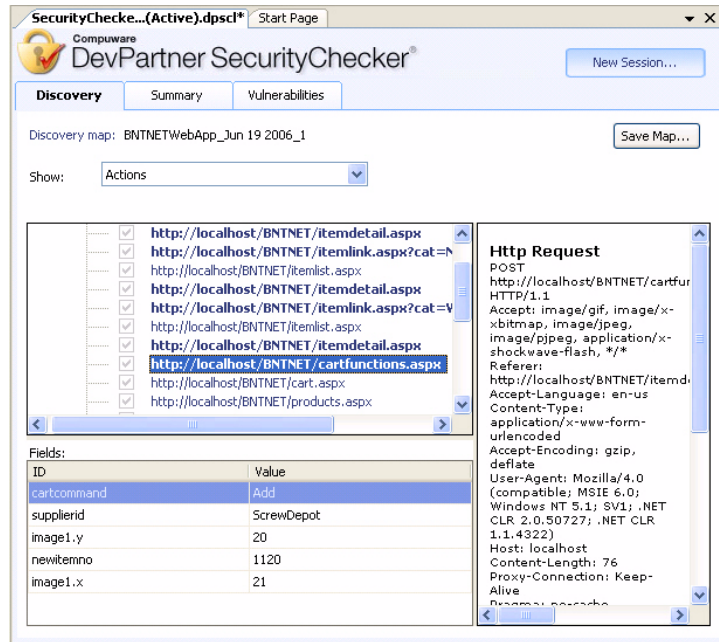


Figure 1-4. Discovery Tab

- 8 When you have completed navigation, close the browser.

The SecurityChecker software then begins analyzing the pages you visited. The **Summary** tab opens and displays a summary of the analysis results. You can watch the progress of the SecurityChecker analysis.

- ◇ A progress meter tracks activity.
- ◇ Analysis results are summarized in the pie chart and the graph.
- ◇ You can view the discovered vulnerabilities while the analysis is in progress.

- 9 Click on a slice of the pie chart to view the **Vulnerabilities** tab with items sorted by the selected Severity. Or, click on a specific severity in the list to see only items listed under that severity level.
- Click on the bar graph to view the **Vulnerabilities** tab showing items sorted by the selected category. Or, click on a specific category in the list to see only the items in that category.

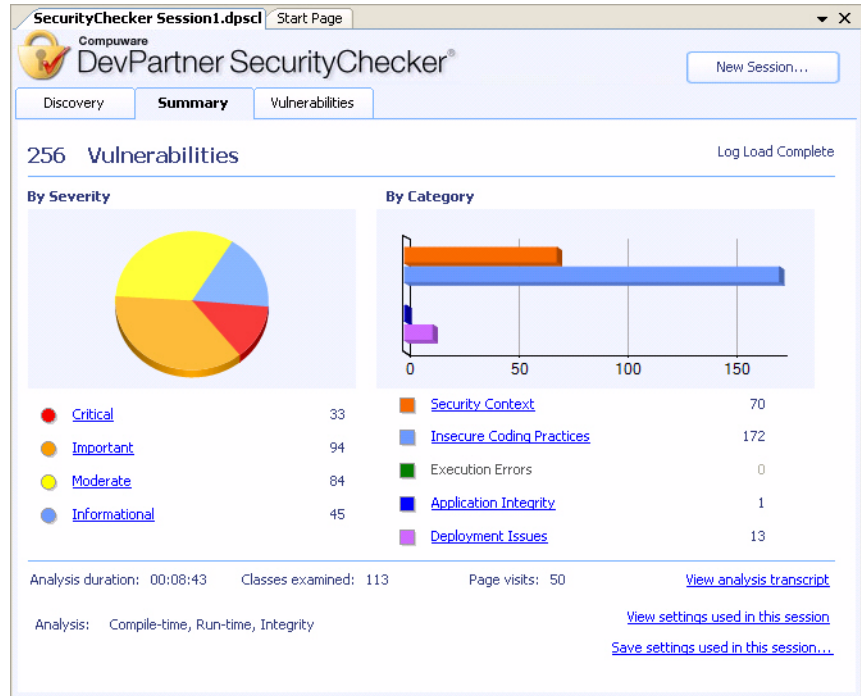


Figure 1-5. Summary Tab

The **Vulnerabilities** tab displays the results grouped by either severity or category. The **SecurityChecker Vulnerability Details** pane displays an explanation and details about each discovered vulnerability. If the **Details** pane is not visible, select **SecurityChecker Vulnerabilities Details** from the Visual Studio **View** menu.

For more information about a specific vulnerability, click on the item. The **Details** pane will show additional information about the vulnerability.

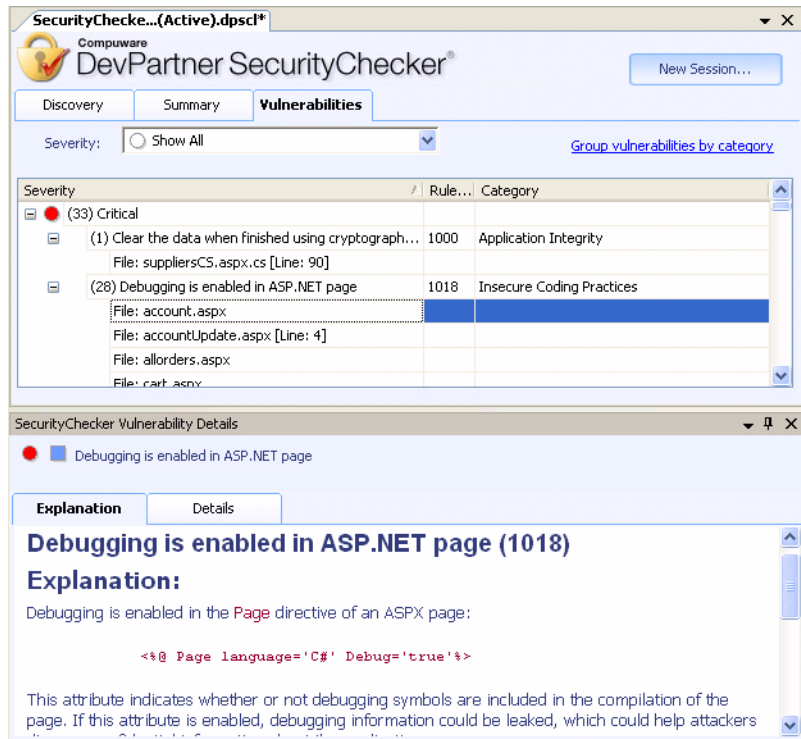


Figure 1-6. Vulnerabilities Tab with Vulnerability Explanation

When you have finished reviewing the results of your session, you can choose to either start a new session (click the **New Session** button to return to the **QuickStart** tab), or exit SecurityChecker. In both cases, you will be prompted to save or discard your session file.

As you become more familiar with SecurityChecker, you can tailor analysis settings to return specific information. From the SecurityChecker menu, select **Settings** to display the Settings dialog box.

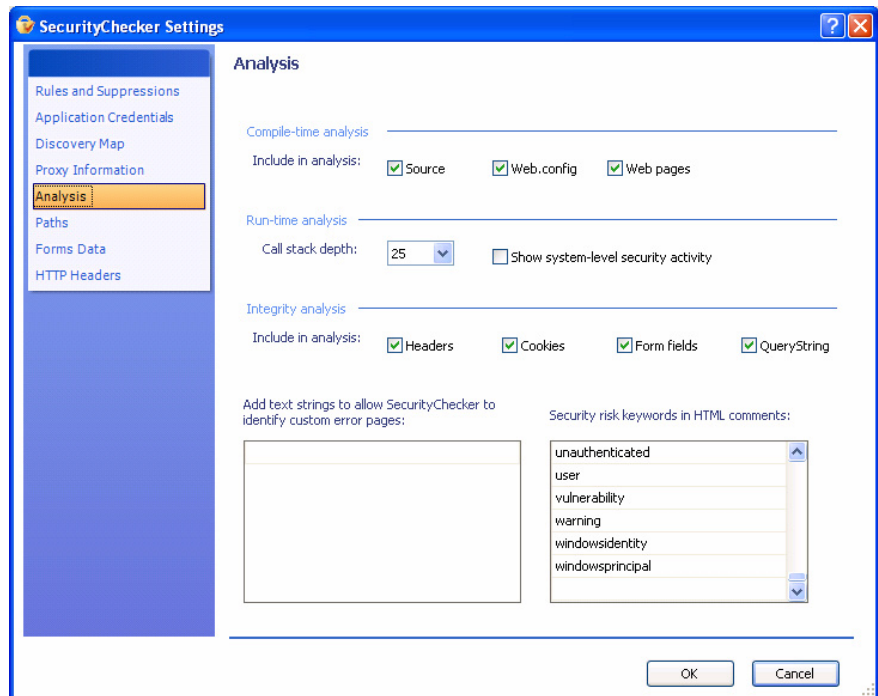


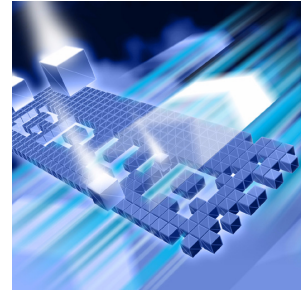
Figure 1-7. Analysis Settings Page - Settings Dialog Box

**Note:** For detailed information about each of the pages in the Settings dialog box, refer to the *Settings Dialog Box* topic under *User Interface* in the SecurityChecker online help.



# Chapter 2

## SecurityChecker Overview



- ◆ SecurityChecker Overview
- ◆ When to Use SecurityChecker

This chapter outlines some of the security vulnerabilities in Web-based applications, and provides conceptual information about how SecurityChecker helps you prevent successful attacks on your application.

### SecurityChecker Overview

The SecurityChecker software finds security vulnerabilities in your ASP.NET application. Designed to be used with Microsoft Visual Studio, SecurityChecker can find the following types of vulnerabilities in your application:

- ◆ Security context issues
- ◆ Insecure coding practices
- ◆ Execution errors
- ◆ Application integrity issues
- ◆ Deployment issues

Use SecurityChecker throughout the development lifecycle to locate security problems. With the increasing threat to Web-based applications, writing insecure code leaves your application, and often your database, vulnerable to attack. Security problems in Web applications expose the company to financial and legal liability. The SecurityChecker software helps solve this problem by arming the developer with three analysis modes to pinpoint security problems:

- ◆ **Compile-time Analysis**

The SecurityChecker software will scan your ASP.NET application source code looking for known security problems. Reviewing the actual source code can locate insecure coding practices down to the method or actual line of code. Analysis of the source code and meta data can also find problems that would be difficult or impossible to locate by only simulating attacks on the application as a whole.

- ◆ **Run-time Analysis**

Once your ASP.NET application executes, SecurityChecker can perform Run-time analysis to verify that your application is not using excessive privileges, accessing files in privileged directories, incorrectly using the registry, or performing other operations that could compromise your application's security.

- ◆ **Integrity Analysis**

Integrity analysis replays a series of known security attacks against the ASP.NET application. Each field, link, and page is tested for known problems such as Cross Site Scripting, SQL Injection attacks, Buffer Overflows, Parameter Tampering, etc.

The SecurityChecker software also analyzes the data returned from the application, looking for incorrect error handling, information leaks, diagnostic messages inadvertently left in the application, insecure comments, and many other vulnerabilities.

The SecurityChecker software gives you two options for analyzing your ASP.NET application:

- ◆ You can direct the analysis to specific pages, fields, and links on a page within your application.
- ◆ You can choose to let SecurityChecker automatically analyze each page in the application, beginning at the start page.

These options give you the ability to customize analysis to see specific pages, or to get "the big picture" of your application's security.

Once you have analyzed your application, you can generate different levels of reports and distribute the analysis results to your team. They can use the reports to correct security issues and validate that recent changes have not introduced new vulnerabilities.

Technical detail reports contain detailed information about the vulnerabilities discovered. Summary reports cover high level information such as the number of issues found, the category, and severity of those issues. The Comprehensive report includes all information gathered during an analysis session.

## When to Use SecurityChecker

Use SecurityChecker throughout the development process, and tailor it to your specific needs at the time. The SecurityChecker software provides three different types of analysis, each suited to a phase of the development cycle.

### A Typical Application Development Cycle

The standard development cycle consists of five phases:

- ◆ Coding
- ◆ Building
- ◆ Testing
- ◆ Debugging
- ◆ Deployment

Use SecurityChecker to perform different types of security analysis during each of the phases of application development.

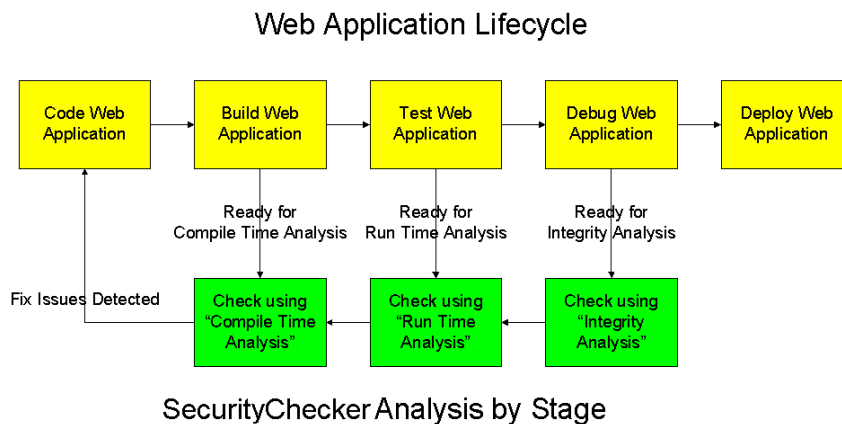


Figure 2-1. When to use SecurityChecker

### Analysis Types in the Development Cycle

The following are the three types of analysis SecurityChecker performs, and when they best fit into the development cycle.

## Compile-time Analysis

Compile-time Analysis searches for vulnerabilities in source code, HTML files, and `web.config` files. A compile-time analysis can be performed at any time throughout the development cycle. Because Compile-time analysis probes only static code, it runs quickly, fitting easily into the early development phase.

## Run-time Analysis

Run-time analysis locates vulnerabilities by examining the internal workings of your code as it executes. Use Run-time analysis early in the development cycle (as soon as you have a working application) to spot vulnerabilities while they are still easy to find and repair. As you add functionality to your application, use Run-time analysis to check for any new vulnerabilities that you may have introduced.

## Integrity Analysis

Integrity analysis simulates attacks on your application using known threats and exploits to reveal vulnerabilities. Integrity analysis can be used any time after the application has been debugged, and should be run before deployment of your ASP.NET application.

Integrity analysis uses a large battery of tests, and takes a significant amount of time to run.

### Changes in Database Records

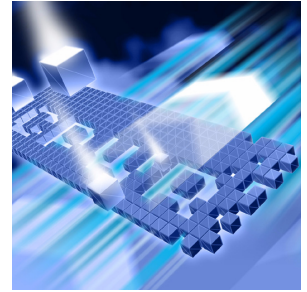
If your application will interact with a database during a SecurityChecker analysis session, create a reset script to return the database to its original state. Run this script immediately after an analysis session. For more information, see the note on [page 8](#).

## Recommended Model for Using SecurityChecker

- ◆ Begin by running Compile-time analysis at regular intervals during the coding and building phase.
- ◆ Add Run-time analysis as the project enters the testing phase.
- ◆ Perform Integrity analysis at the completion of any work unit, as well as in the debugging phase. Because Integrity analysis provides excellent field validation, it should be used often.
- ◆ As you make changes and repair vulnerabilities, run SecurityChecker to verify that no new vulnerabilities have been introduced.
- ◆ At Code Complete, run a full analysis on the application to verify readiness for production.

# Chapter 3

## Discovery and Discovery Maps



- ◆ Understanding Discovery and Discovery Maps
- ◆ Using Automatic Discovery
- ◆ Using Saved Discovery Maps
- ◆ Troubleshooting the Limitations of Discovery

This chapter describes the concept of discovery and the use of discovery maps. At the end of the chapter you will find a section covering the troubleshooting of discovery issues.

### Understanding Discovery and Discovery Maps

Discovery is the process during which SecurityChecker walks through your application structure and records the paths it takes. SecurityChecker then retraces the discovery map, analyzing your application and reporting the vulnerabilities it finds.

#### *Application Discovery*

Before SecurityChecker analyzes your compiled application, you must provide a map of the analysis path. You can choose to create a map, either manually or automatically, or use an existing map from a previous session.

**Note:** The SecurityChecker software does not create a discovery map for Compile-time analysis because it analyzes only the source code and meta data, not the operation or interaction of the application.

- ◆ Manual discovery is the method by which you direct SecurityChecker through your application. This allows you to choose specific pages and controls to be analyzed. Use the **Check only the pages that I visit** button on the QuickStart page to choose manual discovery.

- ◆ Automatic discovery gives you a broad view of your application's security. It provides a way to test pages, links, and fields without manually guiding SecurityChecker through the entire application. Use the **Check all pages in my application** button on the QuickStart page to choose automatic discovery.

The discovery map lists each uniform resource identifier (URI) visited during the discovery process. Both Run-time and Integrity analysis use discovery maps.

### Manual Discovery - Check only the pages that I visit

When you select **Check only the pages that I visit** and start the analysis, Internet Explorer will open. Navigate through your application, exercising the pages and components you want to analyze. The SecurityChecker software records the pages you visit, the links and controls you navigate through, and the data you enter. When you are finished, exit Internet Explorer to end the discovery process. The SecurityChecker software then uses the map you created to analyze your application.

### Automatic Discovery - Check all pages in my application

When you select **Check all pages in my application**, SecurityChecker uses a Web spider to automatically explore your project. Referred to as Automatic discovery, this process attempts to touch every part of your project. For pages that contain data fields, SecurityChecker has a Settings page that allows you to enter values so those pages can be exercised. However, when there is a form used on the start page (such as login and password), SecurityChecker will prompt you for the information in order to continue.

### Using an Existing Discovery Map

A third option on the QuickStart page, **Use a discovery map**, allows you to use a discovery map from a previous session. Use this option anytime you need to re-test an application; such as to validate code after making repairs, as part of a nightly regression suite, or if you have a large application that changes infrequently.

## When to Use Manual Discovery

Manual discovery allows you to analyze only the parts of the project that are of interest to you. In a complex Web application with extensive links, fields, and pages, limiting the analysis scope results in a more focused and manageable session.



Figure 3-1. Focusing the Scope of the Analysis

For example, use manual discovery when:

- ◆ You are part of a development team working on a large Web application, and only have responsibility for specific parts of the project.
- ◆ You have already made repairs to the code, and you want to test a specific page of your application.

The procedure for creating a manual discovery map using the **Check only the pages that I visit** option appears in [“Quick Start”](#) on page 6.

## When to Use Automatic Discovery

Automatic discovery gives you a broad view of your application’s security. It provides a way to test all pages, links, and fields without manually guiding SecurityChecker through the entire application. For example, use automatic discovery when:

- ◆ You are nearing deployment and want to check the entire application for security vulnerabilities.
- ◆ You are in the debugging phase and you want to create a large testing script. Use automatic discovery to ensure full coverage of your application. You can then use these large discovery maps to uncover bugs in your application, or as part of a regression testing suite run from the command line interface. For more information on using the command line, see [“Using the Command Line Interface in an Automated Test Environment”](#) on page 59.

## Using Automatic Discovery

Automatic discovery will explore your entire application, with some limitations. In applications where the first page requests login information, SecurityChecker will prompt you to enter valid data before continuing.

For data fields on subsequent pages, you can use the **Forms Data** page in the **Settings** dialog box to provide data that would be entered in text boxes within your application. When you specify an ID and Value on this page, you can enter data once for multiple fields with the same ID.

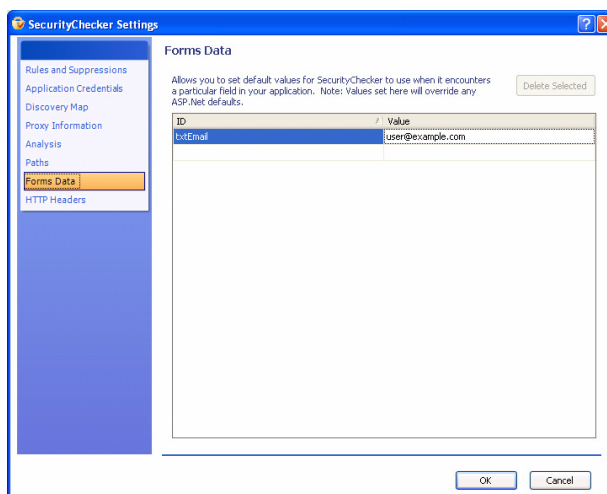


Figure 3-2. Settings Dialog Box - Forms Data Page

Consider the following example: Your application includes a text box control identified as `city`. If you enter `city` in the ID column and Chicago as the corresponding Value, every time you analyze your application with an existing discovery map SecurityChecker will supply Chicago as the value every time it encounters a blank `city` text box.

**Note:** For detailed information about each of the pages in the Settings dialog box, refer to the *Settings Dialog Box* topic under *User Interface* in the SecurityChecker online help.

You can also enter field data prior to analysis when using an existing discovery map. See [“Editing a Discovery Map”](#) on page 26 for more information on adding field data.



## Automatically Creating a Discovery Map

Use the following procedure to automatically create a discovery map. Once created, you can save the discovery map for future use. You can also choose to save the discovery map automatically at the end of the session. Selecting the **Automatically save discovery map with analysis** checkbox on the **Discovery Map** page in the **Settings** dialog box will save the map to your solution directory. See “[Adjusting Automatic Discovery Settings](#)” on page 24 for additional information.

- 1 From the **QuickStart** tab, select **Check all pages in my application**.

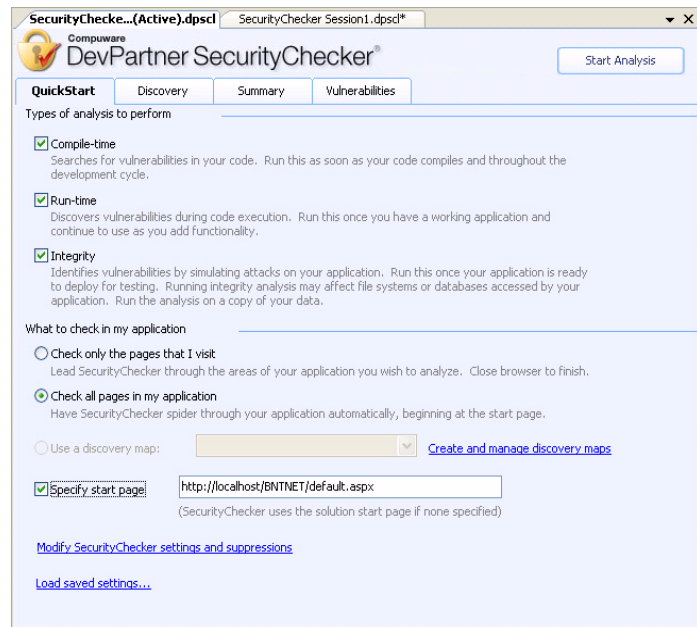


Figure 3-3. Automatically Creating a Discovery Map

- 2 Click **Start Analysis**.

If the start page requires information, such as user ID and a password, the **Start Page Field Data** form appears.

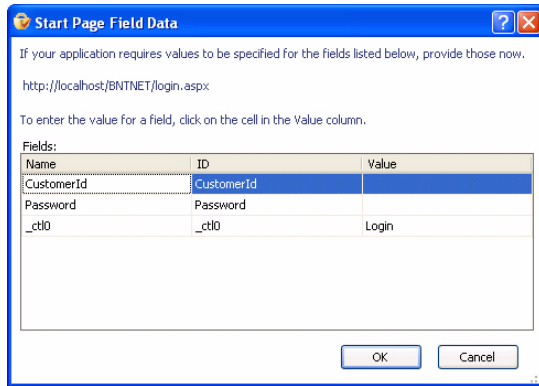


Figure 3-4. Start Page Field Data Entry

3 Enter the required information and click OK.

The **Discovery** tab appears. As SecurityChecker walks through your application, the **Actions** list on the **Discovery** tab lists each page it accesses. Any pages that contain data fields will appear in bold.

In addition, the pane to the right displays HTTP request and response information for a selected action.

Once the discovery is complete, you will begin to see session results. For information about analyzing the session results, see “[Interpreting Analysis Results](#)” on page 40.

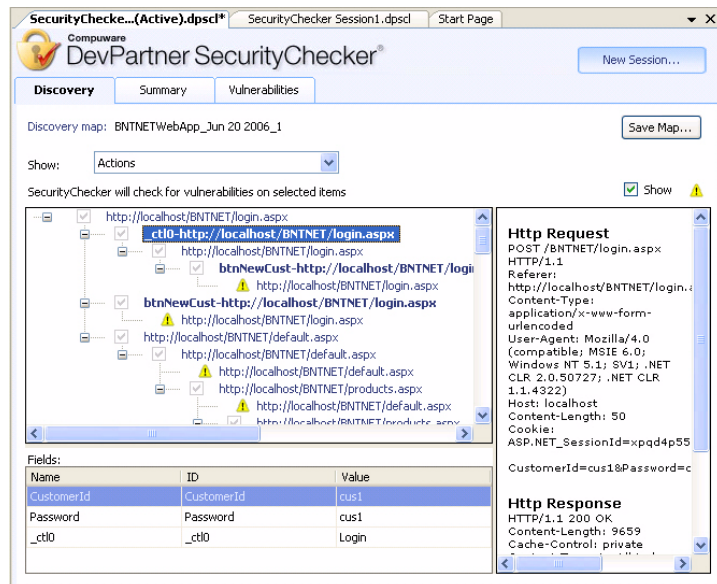



Figure 3-5. Discovery Tab

## Errors and Warnings During Discovery

Errors and warnings are displayed in the status pane to the right of the **Actions** list on the Discovery tab.

The **Individual Pages** view shows only one instance of each page visited, even if the page was visited 15 times. When **Individual Pages** is selected from the drop down, icons indicate that some or all instances of a page contain errors and warnings. If each visit did not generate a warning or error (for example, only 6 visits generated a warning), SecurityChecker will display a semi-warning (icon shows gray).

### Errors

During discovery, SecurityChecker may encounter a page that cannot be processed due to HTTP errors. These pages can be identified by the error icon . Selecting a page with an error will display specific details about the error in the status pane to the right of the **Actions** list.


Errors displayed in the Discovery map are not based on vulnerabilities discovered during analysis.

If you are looking at the pages accessed by SecurityChecker in the individual pages view, you can use the icon next to the page to find out if any or all instances of the page had warnings or errors.

### Warnings

During discovery, SecurityChecker may encounter a page that:

- ◆ Exceeds the limit for crawl depth set in the **Settings** dialog box
- ◆ Exceeds the link visitation limit set in the **Settings** dialog box
- ◆ Exceeds the limit for maximum links per page set in the **Settings** dialog box
- ◆ Is outside of the domain/solution set in the **Settings** dialog box
- ◆ Is part of the BlackList file (For information on the BlackList file, see [“Using the BlackList File”](#) on page 25)

These pages are flagged with a warning icon . The warning indicates that these pages will not be included in the analysis. The details of the warning are displayed in the status pane. After reviewing the details, you might decide to include the page. You can do so by modifying the setting indicated by the warning.

If you are looking at the pages accessed by SecurityChecker in the individual pages view, you can use the icon next to the page to determine if any or all instances of the page had warnings or errors.

*Tip:* For information on any of these settings see [“Adjusting Automatic Discovery Settings”](#) on page 24.

## Turning Warnings Off

If you find that viewing the warnings is unnecessary in your session, or you would like to focus only on the active pages, you can hide the warnings in a discovery map. Use the **Show** ⚠ check box to hide warnings. Re-selecting the check box will display the warnings.

## Adjusting Automatic Discovery Settings

The default configuration for automatic discovery provides enough information to accurately locate most security vulnerabilities. However, there may be circumstances when you want to adjust how deep SecurityChecker goes into your application. For example, you may have pages in your application that are highly cross-referenced. The default crawl depth of 10 may be returning too many page references. Adjust automatic discovery activity using the settings on the **Discovery Map** page of the **Settings** dialog box, described below.

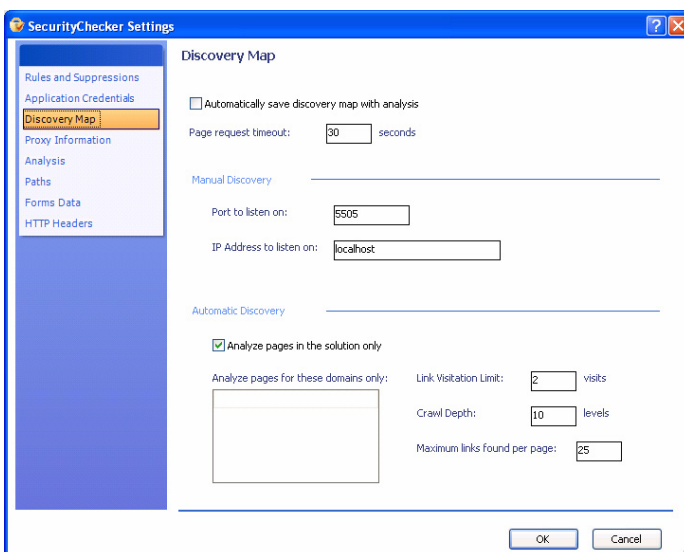


Figure 3-6. Settings Dialog Box - Discovery Map

- ◆ **Analyze pages in this solution only:** When selected, automatic discovery follows links to pages only in the current solution. Links to other pages on the local computer or the Internet are not followed.
- ◆ **Analyze pages for these domains only:** Limits the domains that SecurityChecker maps in automatic discovery mode. Click in this text field and type one or more domain names.
- ◆ **Link visitation limit:** This setting prevents the spider from looping back on itself infinitely. Limiting Link visitation helps if your application has a navigation bar on each page. The default value is 2.

- ◆ **Crawl depth:** This setting limits the maximum number of levels that SecurityChecker will crawl through while performing the discovery analysis. Use Crawl Depth to limit discovery if your application is highly cross-referenced. The default value is 10.
- ◆ **Maximum links per page:** This setting limits the number of links on a page that will be followed and mapped. Use the Maximum links per page setting if your application returns large data grids containing repeated information. The default value is 25.

## Using the BlackList File

The BlackList XML file (BlackList.dpscb) lists the pages that automatic discovery will ignore while analyzing your application. Using the BlackList file lets you narrow the focus of your analysis, similar to manual discovery. When you start a session, SecurityChecker scans the solution directory and uses any valid BlackList.dpscb file found.

A sample BlackList file:

```
<BlackList>
  <Page name="AccountCreationStep1.aspx" />
  <Page name="/BNTNET/header.aspx" />
  <Page name="http://localhost/BNTNET/
  AccountCreationStep3.aspx" />
</BlackList>
```

You can format an entry in the BlackList file in one of three ways:

- ◆ Provide the full path to the page to be avoided:

```
Http://localhost/testapp/Home.aspx
```

- ◆ Provide the local path to the page:

```
/testapp/home.aspx
```

- ◆ List only the page name:

```
Home.aspx
```

## When to Use the BlackList

Use the BlackList file with automatic discovery in the following situations:

- ◆ When you are analyzing a large application, but know that there are a few specific pages you will never need to analyze (that are not part of your responsibility).

- ◆ When you are analyzing specific pages in an application, but want to make sure you do not miss any part of those pages (exclude all other pages using the BlackList).

## Using Saved Discovery Maps

You can create discovery maps to analyze specific parts of the project, and save them for later use in validation and quality assurance testing.

You can use a saved discovery map in the following ways:

- ◆ Reused without changes
- ◆ Edited to provide a more focused approach
- ◆ Saved to be used in a later session or regression test, or as part of a batch file run from the command line.

## Reusing a Discovery Map

Reusing a discovery map expedites validation testing both for you and Quality Assurance testers. To reuse a discovery map:

- 1 On the **QuickStart** tab, select **Use a discovery map**.
- 2 Use the drop down box and choose the appropriate discovery map.
- 3 Select the type of analysis to be performed (Run-time or Integrity) and click **Start Analysis** to start the session.

## Editing a Discovery Map

Use the **Manage Discovery Maps** dialog box to select and edit an existing discovery map. Edit a discovery map by:

- ◆ Selecting or deselecting uniform resource identifiers (URI's) from the **Actions** list
- ◆ Altering field data associated with a given URI

From the **Actions** list in the map viewer, you can select or clear items to include or omit specific pages and controls. When you select a URI that contains a data field, the **Fields** pane displays the associated field ID and value. The SecurityChecker software will use the data during the analysis session when the field is encountered, further automating the analysis process. This level of detail can save time when validating fixes or running regression tests. To edit a discovery map:

- From the **QuickStart** tab, select **Create and manage discovery maps**, next to the discovery map drop down list. You can also choose **Manage Discovery Maps** from the SecurityChecker menu.

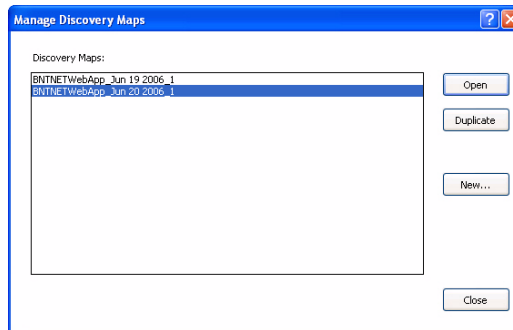


Figure 3-7. Manage Discovery Maps Dialog Box

- When the **Manage Discovery Map** dialog box appears, select an existing discovery map from the list, and click **Open**.

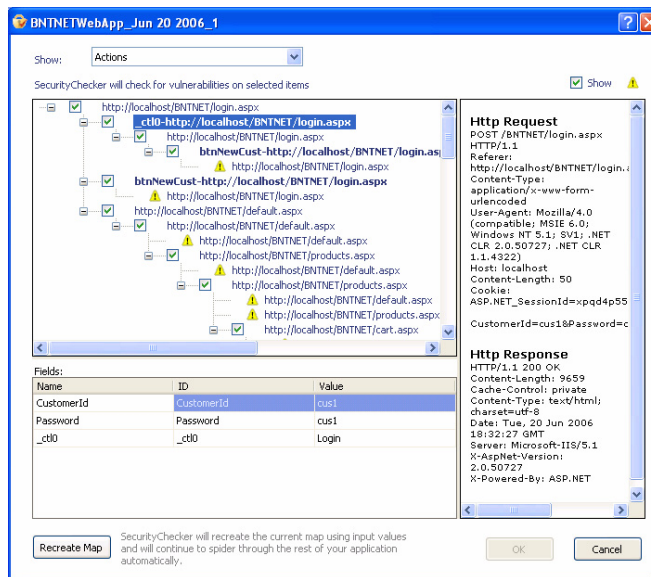


Figure 3-8. Map Viewer

- In the **Actions** list, select or clear checkboxes to include or omit specific pages and controls.

The de-selected pages remain in the discovery map and can be re-enabled at a later date.

The **Fields** area displays the values for the fields and controls that accept input on the page (indicated by URI's listed in bold). Data that must be present to verify the operation of the application, such as a valid customer name, or a part number, should be entered in the **Value** column.

- 4 Click **OK** to save the map, or select **Recreate Map** to create a new discovery map using the edited values.

## **Renaming a Solution After Saving a Discovery Map**

If you have saved a discovery map for a solution, renaming that solution renders the saved discovery map unusable. An error message displays, saying that the discovery map was created for a different solution and cannot be used.

Deleting the discovery map file from the solution directory, or the directory specified in the settings file, will resolve the problem.

Other circumstances that may invalidate a discovery map for a solution:

- ◆ Modifying the solution by deleting projects or .aspx pages within the various projects of the solution
- ◆ Modifying the pages in projects within the solution by renaming or deleting input controls on the pages in the solution
- ◆ Modifying the logic in the code behind the pages by changing the redirection of responses to new destination URI's

In cases where the repair of security vulnerabilities has resulted in significant changes to projects within a solution, create a new discovery map for the project or solution. Choose automatic discovery to assure that no part of the solution is overlooked.

## **Troubleshooting the Limitations of Discovery**

This section covers the limitations of automatic and manual discovery, and provides troubleshooting tips and workarounds when available.

### **Limitations of Automatic Discovery**

The following are issues encountered by automatic discovery. Workarounds are provided when available.



## Multipart HTTP Requests

Automatic discovery does not support multipart HTTP requests (i.e., messages that include the HTTP header field “Content-Type: multipart/...”). Requests of this type are often generated when uploading image files to a web server. They will appear in the discovery map, but will be grayed out. The page will not be analyzed by SecurityChecker Integrity or Run-time analysis.

## Executing Javascript

Automatic discovery can only parse standard javascript routines (i.e., `__doPostBack`). Currently there is no method for automatic discovery to execute javascript, and therefore those methods will be ignored. The result is that pages linked using javascript will not be found. For example, if a custom routine for an `onclick` handler redirects to another page, this page will not be found by automatic discovery. If your application uses javascript redirects, we recommend using manual discovery.

## Activating Controls

When automatic discovery encounters a page containing a “log off” option, it will activate the control and log off. The analysis will continue, but only at the access level of a “logged off” user. If you want analysis to continue as a full access user, you can use the Discovery Map dialog to remove the page from the analysis, or use Manual discovery.

*Tip: An additional option is to modify your application and temporarily disable the “logoff” control.*

## Database Modification

As SecurityChecker maps an application, it activates all buttons, fields, and links. If clicking these buttons or links causes the addition, modification, or deletion of records, the analysis phase will exercise the application and cause these modifications or deletions to occur. This is exercising the application as it was shown to SecurityChecker either through manual or automatic discovery.

If your application interacts with a database, consider creating a reset script to return the database to its original state. This script should be run immediately after an analysis session. Or, run SecurityChecker against a copy of your database.

## Client Side Issues

Automatic discovery does not emulate client side functions such as scripting.

## Limitations of Manual Discovery

The following are issues encountered by manual discovery. Workarounds are provided when available.

### HTTP 1.0 Protocol

Manual discovery does not support the HTTP 1.0 protocol. Ensure that Internet Explorer is configured to use the HTTP 1.1 protocol.

- 1 From the Internet Explorer window, Select **Tools > Options**.
- 2 Select the **Advanced** tab of the dialog box.
- 3 Under the HTTP 1.1 settings category, check both **Use HTTP 1.1** and **Use HTTP 1.1 through proxy connections**.

### Multipart HTTP Requests

Manual discovery does not support multipart HTTP requests (i.e., messages that include the HTTP header field “Content-Type: multipart/...”). Requests of this type are often generated when uploading image files to a web server. They will appear in the discovery map, but will be grayed out. The page will not be analyzed by SecurityChecker Integrity or Run-time analysis.

### Encryption Issues Using Manual Discovery - Secure Socket Layers and HTTPS Pages

The following issue is part of normal operation of encrypted communications. It represents a limitation of SecurityChecker.

Because communication over secure socket layers is encrypted, manual discovery cannot be used to monitor HTTPS communications. Since the client and server share a certificate used for the encryption of their conversation, a proxy sitting between them will not be able to make sense of the data transfer. Manual discovery will not work because it inserts a proxy between the client and server. **Automatic discovery does not insert a proxy between such communication, and will generate a discovery map.**

## Integrated Windows Authentication

Using Integrated Windows authentication for a solution will not allow manual discovery to take place. Integrated Windows Authentication (NTLM authentication) cannot be used through a proxy server. The SecurityChecker software will recognize this condition and display an error message. To work around this issue, configure the application to allow either anonymous login or basic authentication. Basic authentication requires you to enter the user name, password, and domain.

## File Types

Manual discovery only collects information for pages of the following file types: .ASP\*, .HTM\*, .AXD, and .ASMX.

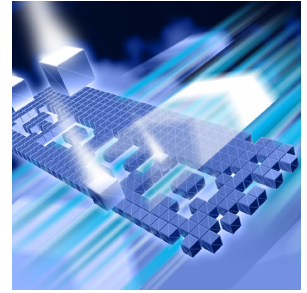
## Client Side Issues

During Integrity analysis, SecurityChecker will only attack cookies when used with manual discovery.



# Chapter 4

## Analysis and Results



- ◆ Analyzing an Application
- ◆ Interpreting Analysis Results
- ◆ Reports
- ◆ Using a Custom XSLT to Create a Report
- ◆ Troubleshooting

This chapter describes general use of the Summary and Vulnerabilities tabs, how to interpret the analysis results, and generating reports. At the end of the chapter you will find a section covering the troubleshooting of minor issues.

### Analyzing an Application

Application analysis is a straightforward process. The SecurityChecker software follows the direction you provide with the discovery map. The results are displayed on the Summary tab and Vulnerabilities tab.

#### *Summary Tab*

You can track progress and see the results of the session on the **Summary** tab. Clicking on an item in either the severity or category lists will open the **Vulnerabilities** tab displaying only the selected vulnerabilities. The pie chart and bar graph provide the same ability to drill down into the vulnerabilities.

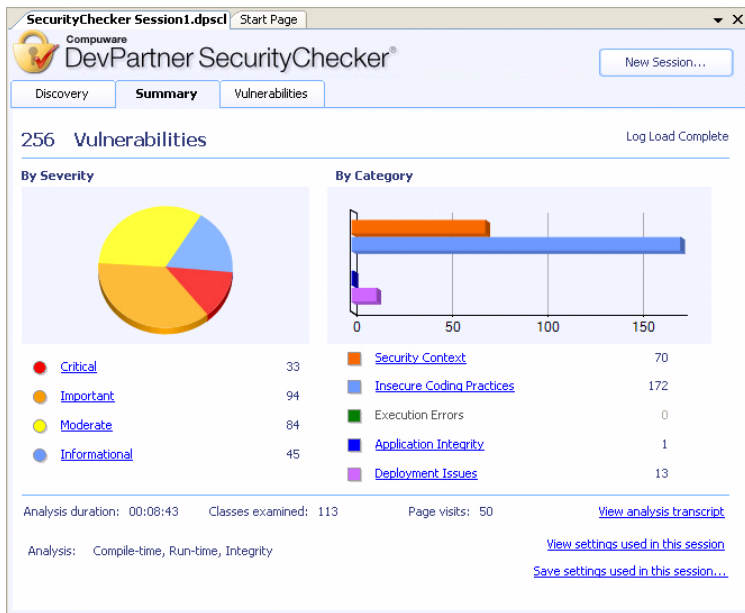


Figure 4-1. Summary Tab

You can view a transcript of the analysis and the settings used during the analysis session by selecting **View analysis transcript** on the bottom right corner of the **Summary** tab.

The Transcript log shows all events during the analysis, and can be viewed during the session. Use this window when you want more information about how the analysis proceeds, or to see details about any errors or inconsistencies encountered during analysis.

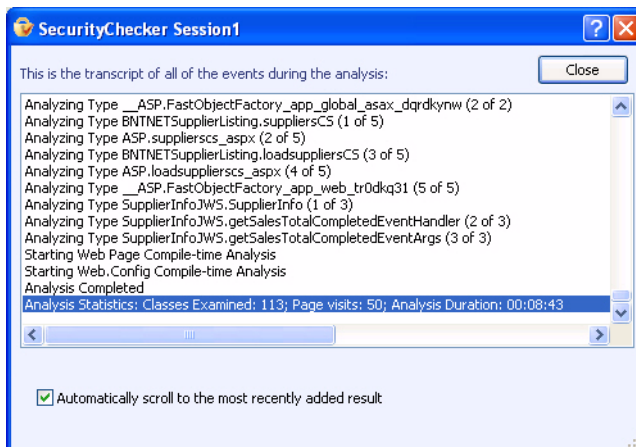


Figure 4-2. Analysis Transcript

## **Classification of Vulnerabilities on the Summary Tab**

The SecurityChecker software places the vulnerabilities reported during analysis into two initial classifications: Severity and Category. The following lists provide a brief explanation of each of the sub-classifications. This information will help you determine what information you may want to focus on during different analysis and development phases. For instance, you may not want to know about deployment issues during early development, but closer to deployment you may want to know about everything except informational vulnerabilities. For information on selecting and deselecting rules, see [“Rules and Vulnerabilities”](#) on page 54.

### **Severities**

There are four levels of severity reported by SecurityChecker: Critical, Important, Moderate, and Informational.

#### **Critical**

Vulnerabilities whose presence indicates the application is currently exposed to a known hacking exploit. The application is at high risk of attack if these issues are not fixed before deployment.

#### **Important**

Vulnerabilities detected whose exploitation could result in compromise of the application. The application may be at risk of attack if these issues are not fixed.

#### **Moderate**

Vulnerabilities that cannot by themselves result in a compromise of the application, but that might result in a compromise if an attacker combined them with other vulnerabilities. It is possible that the application may be attacked due to these issues dependent on the surrounding set of circumstances. It is advisable to fix these issues.

#### **Informational**

Informational vulnerabilities are classified as code that might affect the security of your application. The information provided should be weighed by the application developer for impact to the application before deployment.

## Categories

There are five categories of Vulnerabilities: Security Context, Insecure Coding Practices, Execution Errors, Application Integrity, and Deployment Issues.

### Security Context

The Security Context category examines the authentication mechanisms in your application. Authentication is defined as **“The process through which the identity of a computer or network user is verified.”** The Security Context category reports vulnerabilities in the code used to:

- ◆ Verify the rights required to access parts of the application
- ◆ Verify the rights to perform the action, the function, or the work in those parts of the application
- ◆ Affect the current state of the security context

Examples where security context vulnerabilities might occur include: NTLM, Forms Authentication, .NET Code Access Security, and Session Cookies.

### Insecure Coding Practices

Known patterns of coding that result in security vulnerabilities in both managed and unmanaged code.

Examples include: Using “unsafe” code in C#, having assemblies open to partially trusted callers, or code performing impersonation.

### Execution Errors

Security vulnerabilities and programming errors that occur at application runtime and expose the application to attack.

Examples of these errors would be unhandled exceptions, failed .NET demands for permissions, buffer overflows, and vulnerabilities open to cross-site scripting attacks.

### Application Integrity

Application Integrity refers to the ability of your application to protect valuable data, such as credit card information or personnel records. This data must be protected for privacy and defense reasons.



Issues with cryptography can compromise application integrity, and lead an attacker to your database. How errors are handled and seen by the user can affect the amount of knowledge an attacker can gain about the application structure. That knowledge widens the attack surface of the application.

## Deployment Issues

Deployment issues are various configuration details such as file system security, remote access capabilities, Web server configuration, and .NET runtime configuration that can affect the overall security of the application. These items all are concerns when an application is deployed.

## Vulnerabilities Tab

The **Vulnerabilities** tab lists all vulnerabilities found in the analyzed application. Use it to drill down and locate the specific issues. The vulnerabilities can be grouped either by **Severity** or **Category**. Use the drop-down box to display a list of specific categories or severities.

The screenshot shows the DevPartner SecurityChecker application window. The 'Vulnerabilities' tab is active, displaying a table of vulnerabilities. The table has columns for Severity, Rule ID, and Category. One vulnerability is selected, and its details are shown in a pane below.

Severity	Rule...	Category
(33) Critical		
(1) Clear the data when finished using cryptograph...	1000	Application Integrity
File: suppliersCS.aspx.cs [Line: 90]		
(28) Debugging is enabled in ASP.NET page	1018	Insecure Coding Practices
File: account.aspx		
File: accountUpdate.aspx [Line: 4]		
File: allorders.aspx		
File: cart.aspx		

**SecurityChecker Vulnerability Details**

Debugging is enabled in ASP.NET page

**Explanation** | Details

### Debugging is enabled in ASP.NET page (1018)

**Explanation:**

Debugging is enabled in the **Page** directive of an ASPX page:

```
<%@ Page language='C#' Debug='true'%>
```

This attribute indicates whether or not debugging symbols are included in the compilation of the page. If this attribute is enabled, debugging information could be leaked, which could help attackers

Figure 4-3. Vulnerabilities Tab and Details Pane

When the source file is available, double-clicking a **Vulnerability** will take you directly to the line or method in the source code where SecurityChecker discovered the vulnerability.

## Details Pane

When you select a vulnerability from the list, the **Details** pane opens, providing additional information. (You can also select **SecurityChecker Vulnerability Details** from the **View** menu to see this pane.)

The **SecurityChecker Vulnerability Details** pane displays four different tabs. The tabs that appear vary depending on what you select in the **Vulnerabilities** tab.

- ◆ When you select a category, severity, or vulnerability, only the **Explanation** tab appears. This is the high level information covering the general information about the selected item, including suggested repairs.
- ◆ When you select an individual vulnerability, the **Explanation** and **Details** tabs appear.
- ◆ The **Security Context** and **Call Stack** tabs display supporting information about vulnerabilities discovered during Run-time analysis.

The header of the **Details** pane also displays icons indicating the category and severity of the selected vulnerability, as well as the title.

### Explanation Tab

The **Explanation** tab describes the vulnerability. It also provides repair procedures and links to additional information. The information in this tab comes from many trusted sources, such as Microsoft Developer Network (MSDN), the Open Web Application Security Project ([www.OWASP.org](http://www.OWASP.org)), CERT Coordination Center ([www.cert.org](http://www.cert.org)), and various publications.

### Details Tab

The **Details** tab provides comprehensive information about the selected vulnerability. Where possible, the line of code containing the vulnerability is displayed. Information in the **Details** tab differs based on the type of analysis used to discover the vulnerability.

- ◆ **Compile-time:** Displays a **Vulnerability information** field.
- ◆ **Run-time:** Displays **Vulnerability information** and **Originating request** fields.

- ◆ **Integrity Analysis** displays **Support data for vulnerability**, including a description of why the rule fired and field information, **Page details**, and **Output captured** fields. A link is also provided that allows you to view the captured page in a Web browser.

## Security Context Tab

The Security Context represents the current security information for the application when SecurityChecker found the vulnerability. The fields are:

- ◆ **Application:** The identity under which the ASP.NET worker process is running.
- ◆ **User:** The browser context.
- ◆ **Details pane:** Contains additional security context information.

## Call Stack Tab

The Call Stack information enables you to interpret code interactions that you can use to identify both potential and actual problems.

Potential problems are issues that by themselves are not significant, but when coupled with other vulnerabilities, they might be considered critical. An actual problem is an issue that presents a clear vulnerability by itself.

While Security Context information will be available for all vulnerabilities discovered during Run-time analysis, Call Stack information is only available when the vulnerability is found as a result of a weakness in the code. Information includes:

- ◆ Unmanaged Thread ID.
- ◆ Module Name and Method/Function.
- ◆ File and Line Information: When PDB (program database) information is available for the item, SecurityChecker displays file and line information. When PDB information is not available, the IL (intermediate language) offset is displayed.

## Interpreting Analysis Results

This section explains different ways you can interpret and use the results of an analysis session.

### *Responding to the Analysis Results*

There are typically two ways to use a security analysis tool; the first is to use SecurityChecker from the beginning of the development cycle, testing early and often. In this case, you can review analysis results (Vulnerabilities) by category or severity. You can choose the type of vulnerabilities to focus on, and repair them over the development cycle.

The second scenario employs SecurityChecker to “fight fires.” Often the tool is not employed until late in the development cycle. Results will most likely be reviewed by severity, with the focus mainly on the Critical, Important, and Moderate groupings.

### *Evaluating the Results*

As you decide what vulnerabilities to fix, consider the following actions:

- ◆ **Analyze the vulnerabilities:** Consider the categories and severities of the detected vulnerabilities.  
Critical and Important vulnerabilities should be reviewed and addressed where appropriate.  
Moderate vulnerabilities should be reviewed and addressed if time permits.
- ◆ **Evaluate business objectives:** Do you have time to repair the detected vulnerabilities, or is there a critical need to deploy the application by a deadline?
- ◆ **Assess what could happen if this vulnerability does not get repaired:**
  - ◇ What is my level of exposure?
  - ◇ What is the probability that this vulnerability could be exploited?
  - ◇ What is the worst thing that can happen if my application is compromised?
- ◆ **Determine the scope of individual vulnerabilities:** Is the repair a simple code fix, or could it start a ripple affect, requiring sweeping changes to the architecture of the application?

## Triage the Results

Once you have a list of detected vulnerabilities, you need to decide how to attack them. In making these decisions, you might divide the vulnerabilities into two groups:

- ◆ **Simple code fixes:** Can the development team repair this vulnerability in a short time, with limited impact on the rest of the application? Typically, a team can repair these types of vulnerabilities during development.
- ◆ **Architectural issues:** Will the repair for this vulnerability require intense rework of the application's structure? You may need to defer repair of a large-scale vulnerability to a later release. In some cases, if you have a critical vulnerability, the feature may need to be removed from the application to avoid exposing the application to this class of attack.

## Using the Information in the Details Pane

Use the **Details** pane to garner more information about individual vulnerabilities when you assess the analysis results. The **Details** pane provides important information about where the error occurred, how SecurityChecker got to the error, and the environmental variables that affect the vulnerability of the application.

## Repair Repeated Vulnerabilities

In some cases, SecurityChecker will detect many vulnerabilities on the same page of your application. For example, Parameter Tampering may show as a vulnerability multiple times on the same page. The comprehensive parameter tampering tests attack fields in many different ways. If one attack succeeds, others may also succeed. Fixing one of the listed vulnerabilities may make the field impervious to attack.

Fixing a few of the reported vulnerabilities often clears up several others. Choose a few obvious issues, repair them, and then re-run the analysis.

# Reports

This section describes how to use reports and modify the SecurityChecker report templates.

## Using Reports

The SecurityChecker software provides HTML based reports that allow you to record the vulnerabilities discovered during analysis, organize the information based on the audience, and easily distribute the information to team members. These HTML reports display information in a format that can be read when SecurityChecker is not installed locally.

Some ways reports can be used:

- ◆ Presenting security status updates (summary level reports)
- ◆ Distributing specific vulnerability information to members of the development team who may not have SecurityChecker installed on their computer (reports can be configured to contain very specific vulnerability details)
- ◆ Archive validation testing results for manual comparison (generate reports while running in batch mode from the command line)

The **New Report** dialog box (Figure 4-4) is accessed from the SecurityChecker menu (**SecurityChecker > New Report**). There are three types of reports: **Summary**, **Detailed**, and **Comprehensive**. Summary and Detailed reports include several default report templates from which to choose. These templates allow you to organize the information by different criteria. When you select a report template, the right pane displays additional information about that template.

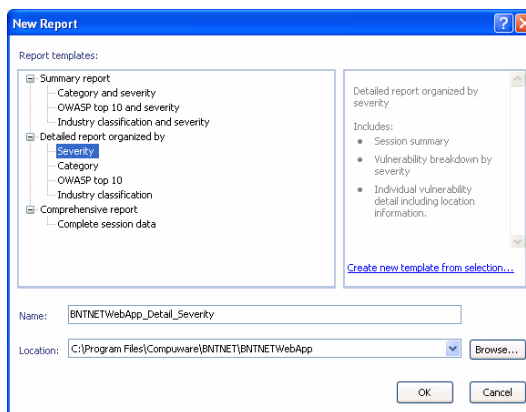


Figure 4-4. New Report Dialog Box

Each of the templates can be selected as a base for modification using [Create new template from selection](#).

## Summary Reports

Summary reports contain abridged information that can be assessed quickly. These reports provide excellent visual representations of the highest level information.

## Detailed Reports

Detailed reports provide vulnerability information based on the report template selected. Templates organize information by:

- ◆ Severity
- ◆ Category
- ◆ OWASP Top Ten
- ◆ Industry Classification

## Comprehensive Report

The Comprehensive report contains complete session data, including explanation and repair information for each vulnerability detected.

## Modifying Report Templates

In some instances you may want to change or limit the way a report presents security vulnerability information about your application. The SecurityChecker software allows you to modify a report template, creating a new report that provides only the data you specify.

## Example

As a development lead, you have performed a SecurityChecker analysis of your Web application. The results indicate several areas of vulnerability. Since any attack that cripples your Web presence has a large impact on revenue, one of your top priorities will be to repair vulnerabilities that could result in a Denial of Service.

The developer assigned to the task requires sufficient detail about the vulnerabilities detected and how to repair them. She does not have SecurityChecker installed locally, so providing her with a log file is not useful. Creating a detailed report focusing only on the critical **Denial of Service attacks** will provide enough information to repair the vulnerabilities, in a format independent of the SecurityChecker software.

To generate this custom report, SecurityChecker must be running and the session file active in the IDE. Make the following selections:

- 1 Open the **New Report** dialog box (**SecurityChecker menu > New Reports**), and select the **OWASP Top Ten** detailed report template.
- 2 Click **Create new template from the selection**.
- 3 In the **New Template** dialog box, give the report a logical name.

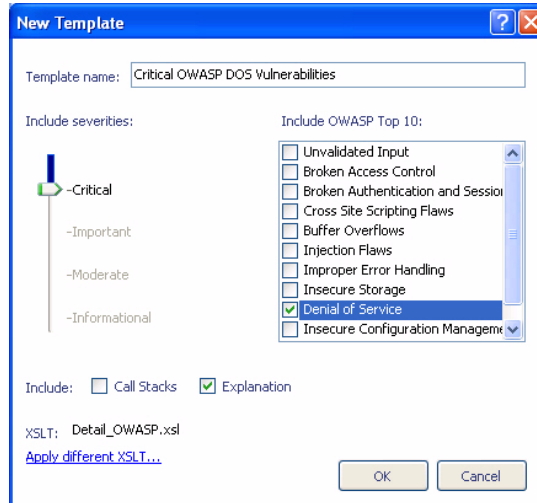


Figure 4-5. New Template Dialog Box - Creating a Custom Report Template

- 4 Use the slide control to select only the **Critical** vulnerabilities.
- 5 In the **OWASP Top Ten** list, clear all checkboxes except **Denial of Service**.
- 6 Under **Include:** select **Explanation**, and click **OK**.
- 7 When the **New Report** dialog box appears, select the new template.
- 8 Choose a directory in which to save the report, and click **OK** to generate the report.

The report containing only the relevant information can be provided to the developer as a printed document, or as a link to the HTML file.

For reports that require a customized HTML output, SecurityChecker provides the option to apply a different XSLT to the data.



## Using a Custom XSLT to Create a Report

The SecurityChecker software provides several default report templates for you to use and modify. In most cases, modifying an existing report template provides enough detail to produce a concise report. However, if you need to add a company logo to your report, or customize the HTML output, writing your own XSLT may be your best option. SecurityChecker allows you to apply your own XSLT to the session data to produce a custom report.

**Note:** XSLT creation is an advanced operation, and is outside the scope of this manual. If you are not familiar with creating an XSLT, we suggest you use one of the existing SecurityChecker XSL files as a template from which to work. If you need additional information about XSLT, perform an internet search or refer to XML or XSLT specific text books.

Using an XML editor, create an XSLT that specifies:

- ◆ The SecurityChecker data to include
- ◆ The format of the report (company logos, titles, page footers, etc.)
- ◆ The output file

When you are finished, save the XSLT.

**Note:** When applying a custom XSLT, be aware that the report template you choose determines the amount of information your XSLT uses as a starting point. A detailed report will provide more information to the transform than a summary report. However, any changes you make in the New Template dialog box will affect the data supplied to the XSLT.

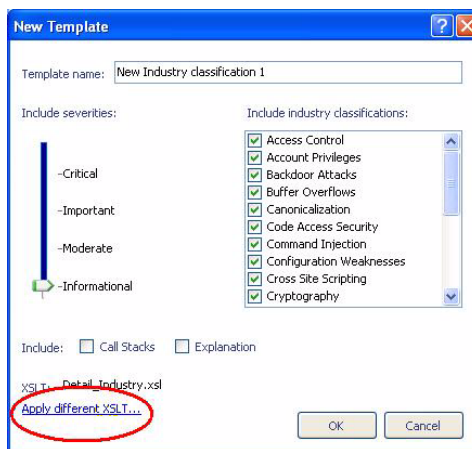


Figure 4-6. New Template Dialog Box - Apply Different XSLT

To use the XSLT, click [Apply Different XSLT...](#) on the **New Template / Modify Template** dialog box. Browse to and select the XSLT to be used.

The new report template will be listed in the **New Report** dialog box.

Report templates created in the **New Template / Modify Template** dialog box can also be applied to reports generated when running in batch mode using the command line interface. For more information on using the command line interface, see [“Using the Command Line Interface in an Automated Test Environment”](#) on page 59.

## Troubleshooting

This section provides information about issues you may encounter with results analysis and reports, and provides troubleshooting tips and workarounds when available.

### ***SQL Injection Attacks May Lower Integrity Vulnerability Counts***

During Integrity analysis, the page under attack has a specified amount of time to respond. If the page does not respond within the allotted time, an error is logged (e.g., Vulnerable to Denial of Service attacks), and processing of the page ceases. For pages vulnerable to denial of service attacks, certain SQL injection attacks will cause the page to exceed the time-out.

In some cases, SQL attacks are launched early in the analysis (for example, when all Integrity rules are enabled). When the page times out, no other attacks are run on this page, potentially resulting in fewer vulnerabilities being reported than if Integrity analysis had been run without the SQL injection attacks.

There are two ways to resolve this issue. The recommended resolution is to repair the SQL injection vulnerabilities and re-run the analysis.

The workaround is to turn off the SQL injection attacks (deselect rules 5500 through 5558) and re-run the analysis. Turning off these rules prevents the time out, allowing other Integrity attacks to be run.

**Note:** Note: You can adjust the Page request time out setting through the Discovery Map page in the Settings dialog box (**SecurityChecker Menu > Settings > Discovery Map**).

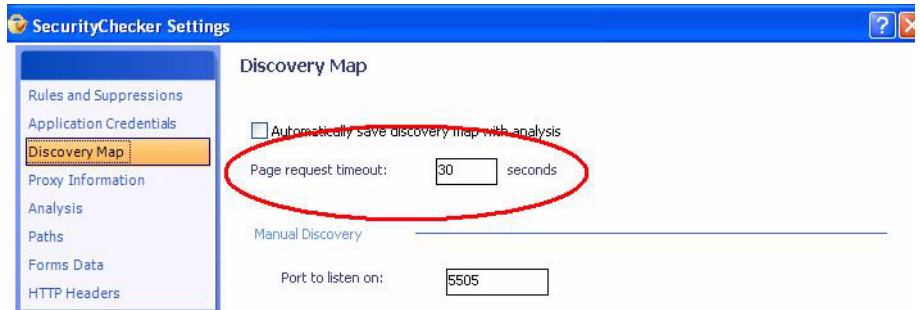


Figure 4-7. Settings Dialog Box - Adjust the Page Request Time out

## Using Reports in Netscape

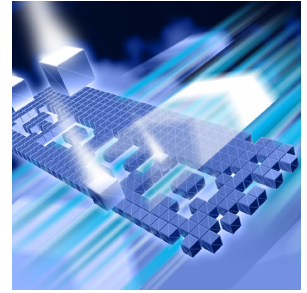
If your default browser is set to open Netscape, Summary reports will not display correctly. When you click **Generate** in the Reports dialog box, the Summary report that appears in Netscape does not show the pie chart and bar graphs. Instead, it shows broken link images.

Netscape is not supported by DevPartner SecurityChecker in this release. Associating HTML file types with Internet Explorer will resolve this issue.



# Chapter 5

## Tailored Analysis



- ◆ Structure of an ASP.NET Application
- ◆ Tailoring Analysis to your Application
- ◆ Rules and Vulnerabilities
- ◆ Suppressions
- ◆ Quality Assurance and Automated Testing
- ◆ Running a Session from the Command Line
- ◆ Troubleshooting

As you become more familiar with DevPartner SecurityChecker, you may want to further refine your analysis, clarify results, or automate analysis sessions. This chapter provides information about security concepts and the advanced use of SecurityChecker features. The information provided in this chapter will refine your security analysis sessions and add focus to the information you gather. At the end of the chapter you will find a section covering troubleshooting of minor issues.

### Structure of an ASP.NET Application

DevPartner SecurityChecker takes advantage of the structure of an ASP.NET application to efficiently analyze the security vulnerabilities of a Web application.

ASP.NET applications, such as online commerce Web sites, are typically very large. These sites serve many departments, each having their own set of pages, forms, and database access requirements. Because of the size and diversity of these sites, a team handles development. Within that team, individual developers work on several ASP.NET solutions. Some solutions are connected, and others are independent. Each solution can contain multiple projects.

On such a large application, security analysis can be broken down into logical, manageable sections, rather than applied to the entire application all at once.

## **Logical Division of the Application**

One or more projects often make up a solution within an ASP.NET application. Each project contains:

- ◆ Source code (.cs and .vb files)
- ◆ Web pages (ASPX, HTML, DHTML, ASAX, ASCX, etc.)
- ◆ Web.config files

A single solution usually uses many Web pages, and often one or more Web.config files.

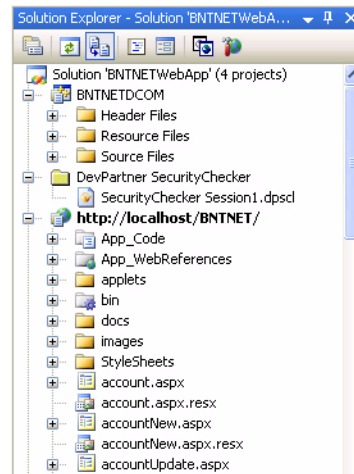


Figure 5-1. ASP.NET Solution File Layout

## Tailoring Analysis to your Application

The features on the **Analysis** page in the **Settings** dialog box enable you to tailor your analysis to collect only the information that you need.

To access the **Settings** dialog box, click **Modify SecurityChecker settings and suppressions** on the **QuickStart** tab, and choose **Analysis** in the **Settings** dialog box.

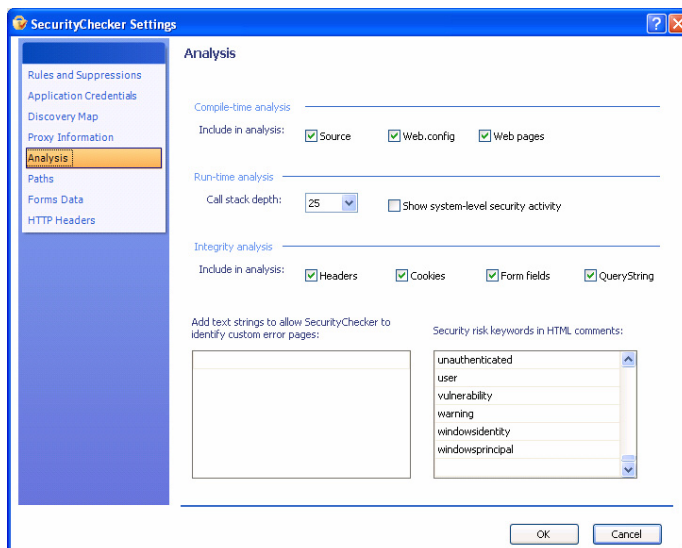


Figure 5-2. Settings Dialog Box - Analysis Page

### Compile-time Analysis

Compile-time analysis examines your source code and the associated meta data, the foundation of your application. An attack that penetrates the outer layer of a Web application will be less likely to extract information or disable the site if there are few vulnerabilities in the structure of the application.

You can refine your analysis with the **Compile-time analysis** selections on the **Analysis** page of the **Settings** dialog box (see [Figure 5-2](#) above).

- ◆ Select **Source** to analyze all source files, including .cs, and .vb files.
  - ◇ If you write the HTML code for the application and do not need to analyze the code-behind modules, clear the **Source** check box.
  - ◇ If you write the code-behind modules, select the **Source** check box to analyze all the source code.

- ◆ Select **Web.config** to detect vulnerabilities caused by authentication problems and debugging messages in your code.
  - ◇ To reduce the number of false vulnerabilities, clear the **Web.config** check box early in the development process, when debugging messages are enabled and authentications are set differently than for production.
  - ◇ Later in the development process, (during pre-deployment analysis, for instance), select the **Web.config** check box to analyze the contents of the **Web.config** file.
- ◆ Select **Web pages** (HTML) to analyze page directives and page-specific information for security issues.
  - ◇ If you know that the Web pages will not be changing, you can clear this option to reduce the analysis time.
  - ◇ If you are responsible for the HTML, enable **Web pages** for specific analyses.

By default, Compile-time analysis looks at all the source code in the solution. If you are responsible for only a portion of the code, you can limit the analysis to specific projects in the solution. In the **Rules and Suppressions** page of the **Settings** dialog box, use the **Projects** tab (at the bottom of the pane) to select the specific projects within the solution you want to analyze.

**Note:** For detailed information about each of the pages in the **Settings** dialog box, refer to the *Settings Dialog Box* topic under *User Interface* in the SecurityChecker online help.

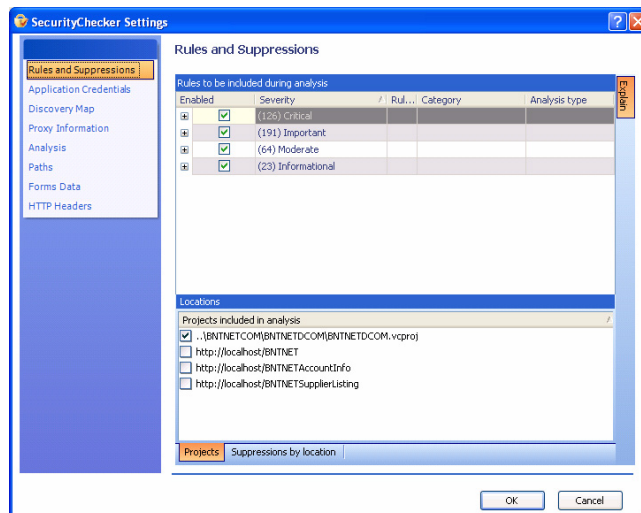


Figure 5-3. Projects Tab at the Bottom of the Rules and Suppressions Page



## Run-time Analysis

Run-time analysis discovers vulnerabilities by examining the internal workings of your code. Use Run-time analysis as soon as you have a working application and continue to use it as you add functionality to your application.

Run-time analysis requires the use of a discovery map to analyze your application. Using manual discovery (**Check only the pages that I visit**) to generate a discovery map can help focus your session. See [“Quick Start”](#) on page 6 for the steps to use manual discovery, or see [“When to Use Manual Discovery”](#) on page 18 for conceptual information about manual discovery.

The following selections appear under **Run-time analysis**:

- ◆ **Call Stack Depth:** Specifies how many call frames are recorded from the call stack for Run-time analysis. The default value of 25 should provide enough stack information to trace errors back from the framework into your ASP.NET application. If you do not see your code listed in the callstack, you may need to increase the number of call frames recorded.

Increasing this value will consume more memory in the ASP.NET worker process and may cause the analysis to take longer.

- ◆ **Show system level security activity:** Activity at this level indicates a change to a Code Access Security -based permission set from the application’s code. This setting can lead to a better understanding of how security settings are established for the executing application’s environment. It also reflects the changes to the initial security context as it evolves throughout the life of the application. Understanding your application code’s security context can be valuable while developing and testing the application, debugging security issues, or resolving deployment issues.

## Integrity Analysis

Integrity analysis simulates attacks on your application using known threats and exploits to reveal vulnerabilities. It exercises every possible link, field, or page in your application, subsequently returning a large amount of information. Using manual discovery during an Integrity analysis session allows you to limit the scope of the analysis. By selecting a subset of all the pages in the application, you can gather only the data that is relevant to your work.

Additionally, the **Analysis** page of the **Settings** dialog box provides optional settings to further refine the analysis results.

If you want SecurityChecker to manipulate various areas of an HTTP Request, enable the corresponding option as listed below.

- ◆ **Headers**
- ◆ **Cookies**
- ◆ **Form Fields**
- ◆ **Query String**

Users who want to speed up Integrity analysis or want to only focus on specific areas of the HTTP Request/Response may want to alter the following parameters:

- ◆ **Add text strings to allow SecurityChecker to identify custom error pages:** If your application uses custom error pages, add strings that appear on that page here. Listing these custom strings helps SecurityChecker identify error pages you have created, and reduces the number of false positive results. For additional information on false positives, see [“What is a “False Positive”?”](#) on page 57.

For example, when a login attempt fails, the page returned should display a message similar to: “Login Attempt Failed.” Displaying a message that says, “Login failed. Bad Password.” gives away too much information and tells the hacker that they are affecting the application by sending bad data.

**Note:** Adding custom error message strings can significantly reduce the number of false positives generated, especially for the parameter tampering rules.

- ◆ **Security risk keywords in HTML comments:** During development, comments are used extensively in code. Comments left in the HTML files when you deploy the application can pose a serious security risk. If your organization has words used in comments not typically used by other organizations, you can add them here. If particular words do not pertain to your application, remove them from the list.

## Rules and Vulnerabilities

The SecurityChecker software detects vulnerabilities based on a fixed list of rules. When vulnerabilities are detected, these rules fire, and SecurityChecker displays information on the **Vulnerabilities** tab.

### Rule Selection

By default, a SecurityChecker analysis session checks for all the rules associated with each of the analysis types. You can select a subset of the

rules to be used for each analysis session by using the **Rules and Suppressions** page in the **Settings** dialog box. A subset of rules can be based on any combination of severity, category, analysis type, or created by selecting individual rules.

**Note:** For detailed information about each of the pages in the Settings dialog box, refer to the *Settings Dialog Box* topic under *User Interface* in the SecurityChecker online help.

You may not want to include the full set of rules in an analysis session:

*Tip:* You can group the rules by category, severity, or analysis type by right clicking in the Rules list and selecting **Group by**.

- ◆ If you do not want to check for deployment issues during development, you can disable these rules by grouping them by **Category** and de-selecting the check box next to **Deployment Issues**.
- ◆ If you are only interested in fixing **Critical**, **Important**, and **Moderate** issues, you can group the rules by **Severity** and de-select the check box next to **Informational** rules.
- ◆ If you are a quality assurance lead, you may want to create a rule baseline that an application must pass before moving to production. You can choose only the rules you need to pass before accepting the application. Any exceptions to the rule set could be denoted using location level suppressions. See [“Location Level Suppressions”](#) on page 56 for more information.

If you need more specific divisions of rules, define individual rule sets that contain a custom mix of categories and severities. To define individual rule sets within a settings file:

**1** From the **Rules and Suppressions** page in the **Settings** dialog box, choose the rules for your rule set.

**2** Run your session.

After the session has completed, save the session settings. Click **Save Settings Used in this Session** on the Summary page. The default name for the settings file is <SolutionName>.dpscc. The **Settings** file contains the session rules set.

Alternative ways to save the settings file are to choose **Save Selected Items** from the **File** menu, or exit the session and wait for the **Save Session** dialog box to appear.

**3** To reuse these settings in a later session, copy the saved settings file from the SecurityChecker folder in the solution explorer.

# Suppressions

The SecurityChecker software provides suppressions as a mechanism to prevent errors from being recorded. You can create suppressions:

- ◆ To prevent SecurityChecker from reporting an error for a problem that you do not think needs to be corrected.
- ◆ To create rule level suppressions to reduce the number of rules SecurityChecker applies against your application.
- ◆ To prevent SecurityChecker from reporting a false positive.

Suppressions will help reduce the size of future session logs, simplifying the task of evaluating future results.

---

**Caution:** Suppressions will be applied to all future sessions *until you remove the suppression from the Rules and Suppressions page of the Settings dialog box.*

---

Suppressions will not be applied to an existing or past log.

## Suppression Types

There are two types of suppressions, *rule* level suppressions and *location* level suppressions.

### Rule Level Suppressions

Suppressing by category, severity, or vulnerability constitutes rule level suppressions. When you suppress at the rule level, SecurityChecker ignores all instances of these vulnerabilities.

Adding rule level suppressions in the **Vulnerabilities** tab is the same as disabling a category, severity level, or individual rule on the **Rules and Suppressions** page of the SecurityChecker **Settings** dialog box.

---

**Caution:** If you suppress a vulnerability at the Rule Level, SecurityChecker will no longer look for that vulnerability anywhere in your application. If possible, consider using Location Level Suppressions.

---

### Location Level Suppressions

Location level suppressions are vulnerabilities suppressed at the file or module level. When you suppress a vulnerability at the location level, SecurityChecker ignores instances of the vulnerability in specific locations, but records them in all others.

**Tip:** If your application successfully handled the vulnerability, consider adding a custom text string to identify the error page generated by your application. This identifies your error page to SecurityChecker, and reduces the number of false positives. For information on adding a custom text string see “Add text strings...” on page 54.

Many of the vulnerabilities will show the file or module name, and other information. When you right-click on file or module vulnerabilities and select **Add to suppression list**, you need to select one of the following:

- ◆ **All vulnerabilities in this location** prevents any vulnerability from being detected in this file or module.
- ◆ **This vulnerability in this location** prevents only this vulnerability from being detected in this file or module.

---

**Caution:** Suppressing all vulnerabilities at a location might hide serious problems that should be addressed. Use location level suppressions carefully.

---

For more detailed information on adding, removing, and the advanced use of suppressions, refer to the *Suppressions* topics under *Using SecurityChecker* in the online help.

## What is a “False Positive”?

The SecurityChecker software might interpret a piece of code as a vulnerability, but in that stage of development it is not a concern. Or it may be reporting an issue in your custom code that you know does not represent a problem. With any automated error detection tool, you will get some number of “false positives.” To reduce these false positives, suppress the rule to remove it from future SecurityChecker sessions.

**Note:** You can also prevent false positives by adding custom error text strings in the analysis settings. Doing so will tell SecurityChecker that your application successfully handled the invalid input.

## Suppressing a Vulnerability

To suppress a vulnerability:

- 1 At the end of an analysis session, click the **Vulnerability** tab.
- 2 Right-click the vulnerability you want to suppress and select **Add to suppression list**.

## Quality Assurance and Automated Testing

The SecurityChecker software is designed as a security vulnerability analysis tool to be used throughout the development cycle. However, this section explores using SecurityChecker as a testing tool to validate the repair of security vulnerabilities uncovered during development. It also explores the use of the command line for such test cases.

## What Quality Assurance Needs for Validation Testing

After the development team repairs the vulnerabilities uncovered with SecurityChecker, developers can provide the quality assurance team with settings files and discovery maps to make validation and regression testing more efficient.

### SecurityChecker Settings and Discovery Map Files

As developers find, fix, and perform validation testing on the vulnerabilities SecurityChecker uncovers, they should save the associated settings files (\*.dpscc) and discovery maps (\*.dpscs). The settings file contains a list of vulnerability rules used in the analysis, as well as other settings that control how the application is analyzed. The discovery map file contains the actions, or Web application requests that exercise specific portions of the application. When the quality assurance testers begin regression testing, these files are used together to duplicate testing the developers have performed, and to verify the operation of the application. The settings and discovery map files can also be used from a command line interface to automate testing. See [“Using the Command Line Interface in an Automated Test Environment”](#) on page 59 for more information.

Developers can also deliver a completed SecurityChecker session file to quality assurance testers. The session file contains the complete set of settings, the discovery map, and the results of the analysis session. Session files can be viewed from within the SecurityChecker interface, or used from the command line to generate reports. A quality assurance tester can also export the session information (from the user interface) to an XML file, and then use a customized tool to compare results from a developers analysis session to one of their own.

### Naming Conventions

The SecurityChecker software saves the settings files and discovery maps to the solution directory by default. The default naming convention for the settings file is the name of the solution (mysolution.dpscc).

The default naming convention for the discovery map is mysolution\_current\_date\_number.dpscs. A discovery map to be reused should be renamed with an intuitive naming convention, including an indication of the part of the application being tested. For example, LoginTest.dpscs, or ShoppingCartTest.dpscs.

## Using the Command Line Interface in an Automated Test Environment

Run SecurityChecker from the command line when you need to automate security analysis. When you run SecurityChecker from the command line, you can:

- ◆ **Integrate SecurityChecker into your automated quality assurance test suites.** The command line interface provides an easy way for any automated test product to launch a SecurityChecker analysis session. Using combinations of SecurityChecker settings and discovery maps, the application can be tested in many different ways. By using the “Export to XML” feature from the command line, automated tools can interpret and compare the results to baseline files.
- ◆ **Cover a large code base automatically.** If your ASP.NET application contains many pages or source code files, an analysis session may take longer than is practical to wait. In this case, the command line can be used to run tasks in a batch process.  
SecurityChecker can also generate a discovery map automatically by spidering the application links. For a large application, this can be done in batch mode as part of the analysis session.
- ◆ **Perform security regression testing.** Using session files or exported XML session data, an automated tool can compare results between sessions to verify that no new issues have been introduced.

## Recording Benchmarks

To track milestones and benchmarks, use SecurityChecker to generate summary and technical detail reports. You can also create custom XSLT style sheets to define your own reports.

Use a commercially available “differencing tool” to validate the analysis results against your current benchmark.

If you find differences, record them.

- ◆ In cases where the quality of the application has improved, a new benchmark will need to be established.
- ◆ If a false positive occurs, create a location based suppression to prevent future occurrences of the vulnerability from being recorded in the log.

If your application has been modified to output a custom error page, adding that information to the SecurityChecker settings will reduce the number of false positives generated during analysis. Use the **Analysis** page of the **Settings** dialog box to add a custom error text string to the session settings.

- ◆ Submit a defect to the defect-tracking tool when a failure is detected.
  - ◇ The SecurityChecker software allows you to save information to a separate directory and include it in your defect report. Access to settings files, reports, and configuration files creates a closed loop system, where you can deliver all the information the developer needs to correct the problem.
  - ◇ If you have Team System installed on your computer, and your templates support the Bug-type work item, you can directly submit defects that include detailed vulnerability information.

## Running a Session from the Command Line

This section provides general information about running automated testing from the command line.

For the specific steps to run a session, refer to the online help topic, *Running SecurityChecker from the Command Line*.

### Command File

To run a session from the command line, you must have a *command file*. Command files are XML files that name SecurityChecker settings files, log files, and report types. If no settings file is specified, SecurityChecker will look for a settings file with the default name based on the solution name. Typically, the format is `mysolution.dpscc`. If the file does not exist, SecurityChecker uses the default settings.

Use the **Create Command File** selection from the SecurityChecker menu to automatically generate a sample command file based on a completed session. Use this file as a template to customize your analysis commands.

You can also create a command file manually. SecurityChecker provides a sample command file for you to modify as necessary. The sample file is located in the SecurityChecker install directory:

```
(C:\Program Files\Compuware\DevPartner  
SecurityChecker\SampleCommandFile.xml).
```

The Command file functions as the user interface for command line operation. The following settings are specified by the command file when running from the command line:

- ◆ Path to the solution to be analyzed
- ◆ Path to the settings file if you are using a settings file other than the default



- ◆ Path and directory to save the session log file
- ◆ Path and directory to save the Command Line log file  
The command line log file is an XML file containing status and tracing information for events occurring during the analysis session. The verbosity of the output can be controlled through the use of command line switches:
  - ◇ /silent hides most analysis output
  - ◇ /verbose displays all analysis information

Use the command file to specify additional actions, such as saving a session log file to a different directory, and then generating and saving multiple reports from the log file.

For more information about the structure and use of the command file, refer to the online help topic, *Running SecurityChecker from the Command Line*.

## Creating Multiple Settings Files

If you run SecurityChecker from the command line to validate repairs or run automated test suites, you should create individual settings files tailored to your test cases.

Follow these steps to create a customized settings file for each test case.

- 1 From the SecurityChecker user interface, open the session file to be modified, or import an existing settings file using **Import Settings file** from the **Quickstart** tab.
- 2 Make the desired changes to the settings.
- 3 Run the analysis session. When the session is complete, verify the results.
- 4 Go to the **Summary tab** and select **Export current settings to a file** to save the SecurityChecker settings.

You may also select **Create a command file** from the SecurityChecker menu to create a command file that represents the session.

**Note:** Since the generated command file is a template, it contains only a sample setting file name. To use a settings file, you must modify the template before saving it.

To modify these settings in the future, import the settings file using the link on the QuickStart tab, and follow the same steps.

## Creating Multiple Discovery Maps

A SecurityChecker settings file references the discovery map used for an analysis session. However, you can modify the Command file to override the discovery map referenced in the settings file. This allows you to have a single settings file containing the rules and configuration information to run the analysis, and individual discovery maps for each test case.

To create a discovery map for each test case, you can edit an existing discovery map or create a new map using the following steps.

- 1 Click **Manage Discovery Maps** on the **SecurityChecker** toolbar.
- 2 Use the **Manage Discovery Maps** dialog box and the map viewer to create a new map or edit an existing map.
- 3 Tailor the new discovery map to cover only the pages needed for the specific test.
- 4 Click **OK** to save the map and close the map viewer.

**Note:** For more information on creating a discovery map, see [“Editing a Discovery Map”](#) on page 26.

## Troubleshooting

This section provides information about issues you may encounter with some of the advanced features of SecurityChecker, and provides troubleshooting tips and workarounds when available.

### Warnings in the Application Events Log

During Integrity analysis, SecurityChecker logs warnings to the Application Event Log. The warnings reflect the unhandled exceptions generated during the attacks executed by SecurityChecker to expose vulnerabilities in the application. Generating a high number of unhandled exceptions is normal operation for a security analysis tool. This is a known issue and will be addressed in a future release. Because the additional warnings may obscure other event information, you may choose to disable the events from being written to the event log using the following procedure.

- 1 Open the IIS Manager (**Start > Settings > Control Panel > Administrative Tools > Internet Information Services**).
- 2 Expand the treeview to expose the **Web Sites** node that appears under the root.

- 3 Right click on the **Web Sites** node and select the **Properties** option from the context menu. (For Windows 2000, select the root machine node and edit the **Master Properties**.)
- 4 Select the **Home Directory** tab.
- 5 Click the **Configuration** button.
- 6 Select the **Process Options** tab.
- 7 Uncheck the **Write unsuccessful client requests to event log** option as shown in [Figure 5-4](#) below.
- 8 Click **OK**.

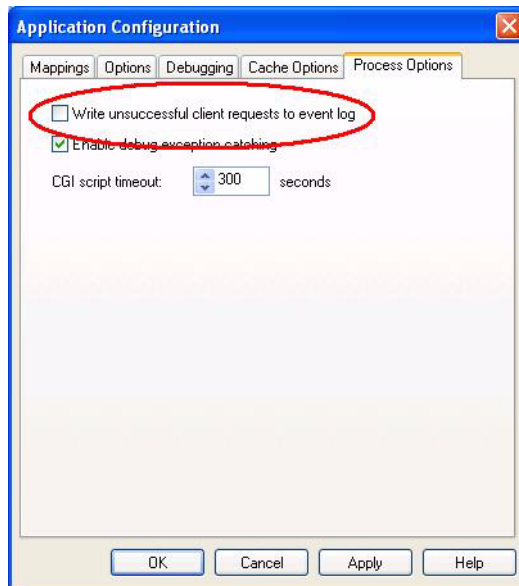


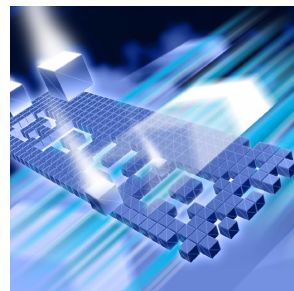
Figure 5-4. Application Configuration Dialog Box

### ***Inaccurate Module and File information***

If you are performing Run-time analysis and do not get accurate module or file information, recompile the application and re-run analysis. Sometimes PDB symbol information does not get copied into the Web server directory.



# Index



## A

- access 36
- actions list 22
- application
  - compromise 40
  - discovery 17
  - integrity 36
- architectural issues 41
- authentication 31, 36
  - basic 31
- automated testing 57
- automatic discovery
  - defined 18
  - limiting, customizing 24
  - using 20
  - when to use 19
- autorun 5

## B

- bar graph 33
- benchmarks 59
- blacklist 25
  - sample file 25
  - use 25
- buffer overflows 36

## C

- call stack 53

- call stack tab 39
- categories 36
  - application integrity 36
  - deployment issues 37
  - execution errors 36
  - insecure coding practices 36
  - security context 36
- CERT 38
- check all pages 21
- client side 29, 31
- code access security 36
- co-existence 3
- command file 60
  - create automatically 60
  - create manually 60
  - settings 60
- command line
  - log file 61
  - switches 61
- command line interface 59
- comments 54
- compile-time analysis 14, 16
  - customizing 51
- comprehensive reports 43
- concurrent license 4
- context, security 39
- controls
  - activation 29
- cookies 31, 54
- crawl depth 23, 25
- creating settings files 61
- critical (severity type) 35
- cross-site scripting 36
- cryptography 37

custom  
error pages 54  
customer service x

## D

data field 26  
data fields 20  
data grids 25  
database modification 29  
database records, changes to 16  
debugging 19  
denial of service 46  
deployment 19  
issues 37  
detailed reports 43  
details pane 38, 41  
details tab 38  
diagnostic messages 14  
discovery maps  
creating multiple 62  
discovery 17, 20  
automatic 18  
manual 17  
discovery map 17, 18, 22, 58  
creating 21  
editing 26  
reuse 26  
saving 28  
using saved 26  
discovery settings 24  
DLM 4.0 4  
domains 24  
downloadable updates 6

## E

encryption 30  
error pages 54  
errors 23  
events log 62  
exceptions 62  
execution errors 36  
explanation tab 38  
exposure 40

## F

false positive 54, 57  
field data, start page 21  
field ID 26  
file information 63  
file types 31  
format  
blacklist 25  
forms  
authentication 36  
data pane 20  
form fields 54

## H

headers 54  
HTML 16  
HTTP request 54  
HTTP requests  
multipart 29, 30  
HTTPS pages 30

## I

icon  
error 23  
show 24  
warning 23  
identity 36  
IL 39  
impersonation 36  
important (severity type) 35  
individual pages 23  
informational (severity type) 35  
insecure coding practices 36  
insecure comments 14  
installation 3  
wizard 5  
installing 5  
integrity analysis 14, 16, 53  
customizing 54  
intermediate language 39

## J

- javascript 29
  - redirects 29

## K

- keywords, security risk 54

## L

- license file
  - getting technical support 4
  - installing 4
  - obtaining 3
- link visitation 23, 24
- looping 24

## M

- manage discovery maps 26, 27
- manual discovery
  - defined 18
  - when to use 18
- maximum links 25
- meta data 51
- model, usage 16
- moderate (severity type) 35
- modify reports 42
- multiple settings files 61

## N

- naming conventions 58
- netscape 47
- new report 43
- NTLM 31, 36

## O

- output captured 39
- OWASP 38

## P

- page details 39
- page request timeout 47
- parameter tampering 41
- PDB 39, 63
- pie chart 33
- privileges 14
- program data base 39
- progress meter 9
- projects tab
  - selecting projects 52
- projects tab 52
- proxy
  - information dialog box 6

## Q

- quality assurance 57
  - regression testing 57
- query string 54
- quickstart
  - creating a discovery map 21
  - getting started 7

## R

- regression testing 19, 58, 59
- renaming
  - solution 28
- report templates 42, 43
  - modifying 43
- reports
  - comprehensive 42
  - detailed 42, 43
  - new template 44
  - standard 42
  - summary 42, 43

- use 42
- reset script 8, 29
- results
  - evaluating 40
- rule sets 55
- rules
  - base line 55
  - defined 54
  - saving 55
  - selection 54
- run-time analysis 14, 16, 53

## S

- sample command file 60
- scope
  - vulnerability 40
- scripting 29
- secure socket layers (SSL) 30
- security context 36, 39
- security context tab 39
- settings
  - analysis settings 51
  - analysis tab 51
  - discovery map 24
  - rules and suppressions 52, 55
  - settings files 58
- severities 35
  - critical 35
  - important 35
  - informational 35
  - moderate 35
- software requirements 2
- solution
  - renaming 28
- spider, Web 18
- SQL injection 46
- strings 54
- subset, rules 54
- summary 33
- summary reports 43
- summary tab 9
- support data 39
- suppressions 52, 56
  - add to suppression list 57
  - all vulnerabilities in this location 57

- list 57
- location level 56
- removing 56
- rule level 56
  - suppressing a vulnerability 57
  - this vulnerability in this location 57
- system level security 53
- system requirements 1

## T

- team system integration 60
- technical support xi
- templates 42
- test cases 57
- test suites
  - integrating 59
- testing 57
- thread ID 39
- transcript log 34
- triage results 41
- troubleshooting 62

## U

- unhandled exceptions 62
- uniform resource indicator 7
- unmanaged 36
- URI 26
- use, when to use 15

## V

- validation testing 58
- values 28
- vulnerabilities 54
  - classifications 35
  - exploit 40
  - severities 35
  - vulnerabilities tab 11, 37



## W

warning

  semi-warning [23](#)

warnings [23](#)

  events log [62](#)

web spider [18](#)

web.config [16](#)

worker process, ASP.NET [39](#)

