

## IDP Series

# Custom Attack Object Reference and Examples Guide

Release

5.1



Published: 2011-08-11

Revision 02

Juniper Networks, Inc.  
1194 North Mathilda Avenue  
Sunnyvale, California 94089  
USA  
408-745-2000  
www.juniper.net

This product includes the Envoy SNMP Engine, developed by Epilogue Technology, an Integrated Systems Company. Copyright © 1986-1997, Epilogue Technology Corporation. All rights reserved. This program and its documentation were developed at private expense, and no part of them is in the public domain.

This product includes memory allocation software developed by Mark Moraes, copyright © 1988, 1989, 1993, University of Toronto.

This product includes FreeBSD software developed by the University of California, Berkeley, and its contributors. All of the documentation and software included in the 4.4BSD and 4.4BSD-Lite Releases is copyrighted by the Regents of the University of California. Copyright © 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994. The Regents of the University of California. All rights reserved.

GateD software copyright © 1995, the Regents of the University. All rights reserved. Gate Daemon was originated and developed through release 3.0 by Cornell University and its collaborators. Gated is based on Kirton's EGP, UC Berkeley's routing daemon (routed), and DCN's HELLO routing protocol. Development of Gated has been supported in part by the National Science Foundation. Portions of the GateD software copyright © 1988, Regents of the University of California. All rights reserved. Portions of the GateD software copyright © 1991, D. L. S. Associates.

This product includes software developed by Maker Communications, Inc., copyright © 1996, 1997, Maker Communications, Inc.

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Products made or sold by Juniper Networks or components thereof might be covered by one or more of the following patents that are owned by or licensed to Juniper Networks: U.S. Patent Nos. 5,473,599, 5,905,725, 5,909,440, 6,192,051, 6,333,650, 6,359,479, 6,406,312, 6,429,706, 6,459,579, 6,493,347, 6,538,518, 6,538,899, 6,552,918, 6,567,902, 6,578,186, and 6,590,785.

*IDP Series Custom Attack Object Reference and Examples Guide*

Copyright © 2011, Juniper Networks, Inc.

All rights reserved.

Revision History

August 2011—Revision 02

The information in this document is current as of the date listed in the revision history.

#### YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. The Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

## END USER LICENSE AGREEMENT

**READ THIS END USER LICENSE AGREEMENT ("AGREEMENT") BEFORE DOWNLOADING, INSTALLING, OR USING THE SOFTWARE.** BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE OR OTHERWISE EXPRESSING YOUR AGREEMENT TO THE TERMS CONTAINED HEREIN, YOU (AS CUSTOMER OR IF YOU ARE NOT THE CUSTOMER, AS A REPRESENTATIVE/AGENT AUTHORIZED TO BIND THE CUSTOMER) CONSENT TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT OR CANNOT AGREE TO THE TERMS CONTAINED HEREIN, THEN (A) DO NOT DOWNLOAD, INSTALL, OR USE THE SOFTWARE, AND (B) YOU MAY CONTACT JUNIPER NETWORKS REGARDING LICENSE TERMS.

1. **The Parties.** The parties to this Agreement are (i) Juniper Networks, Inc. (if the Customer's principal office is located in the Americas) or Juniper Networks (Cayman) Limited (if the Customer's principal office is located outside the Americas) (such applicable entity being referred to herein as "Juniper"), and (ii) the person or organization that originally purchased from Juniper or an authorized Juniper reseller the applicable license(s) for use of the Software ("Customer") (collectively, the "Parties").

2. **The Software.** In this Agreement, "Software" means the program modules and features of the Juniper or Juniper-supplied software, for which Customer has paid the applicable license or support fees to Juniper or an authorized Juniper reseller, or which was embedded by Juniper in equipment which Customer purchased from Juniper or an authorized Juniper reseller. "Software" also includes updates, upgrades and new releases of such software. "Embedded Software" means Software which Juniper has embedded in or loaded onto the Juniper equipment and any updates, upgrades, additions or replacements which are subsequently embedded in or loaded onto the equipment.

3. **License Grant.** Subject to payment of the applicable fees and the limitations and restrictions set forth herein, Juniper grants to Customer a non-exclusive and non-transferable license, without right to sublicense, to use the Software, in executable form only, subject to the following use restrictions:

- a. Customer shall use Embedded Software solely as embedded in, and for execution on, Juniper equipment originally purchased by Customer from Juniper or an authorized Juniper reseller.
- b. Customer shall use the Software on a single hardware chassis having a single processing unit, or as many chassis or processing units for which Customer has paid the applicable license fees; provided, however, with respect to the Steel-Belted Radius or Odyssey Access Client software only, Customer shall use such Software on a single computer containing a single physical random access memory space and containing any number of processors. Use of the Steel-Belted Radius or IMS AAA software on multiple computers or virtual machines (e.g., Solaris zones) requires multiple licenses, regardless of whether such computers or virtualizations are physically contained on a single chassis.
- c. Product purchase documents, paper or electronic user documentation, and/or the particular licenses purchased by Customer may specify limits to Customer's use of the Software. Such limits may restrict use to a maximum number of seats, registered endpoints, concurrent users, sessions, calls, connections, subscribers, clusters, nodes, realms, devices, links, ports or transactions, or require the purchase of separate licenses to use particular features, functionalities, services, applications, operations, or capabilities, or provide throughput, performance, configuration, bandwidth, interface, processing, temporal, or geographical limits. In addition, such limits may restrict the use of the Software to managing certain kinds of networks or require the Software to be used only in conjunction with other specific Software. Customer's use of the Software shall be subject to all such limitations and purchase of all applicable licenses.
- d. For any trial copy of the Software, Customer's right to use the Software expires 30 days after download, installation or use of the Software. Customer may operate the Software after the 30-day trial period only if Customer pays for a license to do so. Customer may not extend or create an additional trial period by re-installing the Software after the 30-day trial period.
- e. The Global Enterprise Edition of the Steel-Belted Radius software may be used by Customer only to manage access to Customer's enterprise network. Specifically, service provider customers are expressly prohibited from using the Global Enterprise Edition of the Steel-Belted Radius software to support any commercial network access services.

The foregoing license is not transferable or assignable by Customer. No license is granted herein to any user who did not originally purchase the applicable license(s) for the Software from Juniper or an authorized Juniper reseller.

4. **Use Prohibitions.** Notwithstanding the foregoing, the license provided herein does not permit the Customer to, and Customer agrees not to and shall not: (a) modify, unbundle, reverse engineer, or create derivative works based on the Software; (b) make unauthorized copies of the Software (except as necessary for backup purposes); (c) rent, sell, transfer, or grant any rights in and to any copy of the Software, in any form, to any third party; (d) remove any proprietary notices, labels, or marks on or in any copy of the Software or any product in which the Software is embedded; (e) distribute any copy of the Software to any third party, including as may be embedded in Juniper equipment sold in the secondhand market; (f) use any 'locked' or key-restricted feature, function, service, application, operation, or capability without first purchasing the applicable license(s) and obtaining a valid key from Juniper, even if such feature, function, service, application, operation, or capability is enabled without a key; (g) distribute any key for the Software provided by Juniper to any third party; (h) use the

Software in any manner that extends or is broader than the uses purchased by Customer from Juniper or an authorized Juniper reseller; (i) use Embedded Software on non-Juniper equipment; (j) use Embedded Software (or make it available for use) on Juniper equipment that the Customer did not originally purchase from Juniper or an authorized Juniper reseller; (k) disclose the results of testing or benchmarking of the Software to any third party without the prior written consent of Juniper; or (l) use the Software in any manner other than as expressly provided herein.

5. **Audit.** Customer shall maintain accurate records as necessary to verify compliance with this Agreement. Upon request by Juniper, Customer shall furnish such records to Juniper and certify its compliance with this Agreement.

6. **Confidentiality.** The Parties agree that aspects of the Software and associated documentation are the confidential property of Juniper. As such, Customer shall exercise all reasonable commercial efforts to maintain the Software and associated documentation in confidence, which at a minimum includes restricting access to the Software to Customer employees and contractors having a need to use the Software for Customer's internal business purposes.

7. **Ownership.** Juniper and Juniper's licensors, respectively, retain ownership of all right, title, and interest (including copyright) in and to the Software, associated documentation, and all copies of the Software. Nothing in this Agreement constitutes a transfer or conveyance of any right, title, or interest in the Software or associated documentation, or a sale of the Software, associated documentation, or copies of the Software.

8. **Warranty, Limitation of Liability, Disclaimer of Warranty.** The warranty applicable to the Software shall be as set forth in the warranty statement that accompanies the Software (the "Warranty Statement"). Nothing in this Agreement shall give rise to any obligation to support the Software. Support services may be purchased separately. Any such support shall be governed by a separate, written support services agreement. TO THE MAXIMUM EXTENT PERMITTED BY LAW, JUNIPER SHALL NOT BE LIABLE FOR ANY LOST PROFITS, LOSS OF DATA, OR COSTS OR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS AGREEMENT, THE SOFTWARE, OR ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. IN NO EVENT SHALL JUNIPER BE LIABLE FOR DAMAGES ARISING FROM UNAUTHORIZED OR IMPROPER USE OF ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. EXCEPT AS EXPRESSLY PROVIDED IN THE WARRANTY STATEMENT TO THE EXTENT PERMITTED BY LAW, JUNIPER DISCLAIMS ANY AND ALL WARRANTIES IN AND TO THE SOFTWARE (WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE), INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT DOES JUNIPER WARRANT THAT THE SOFTWARE, OR ANY EQUIPMENT OR NETWORK RUNNING THE SOFTWARE, WILL OPERATE WITHOUT ERROR OR INTERRUPTION, OR WILL BE FREE OF VULNERABILITY TO INTRUSION OR ATTACK. In no event shall Juniper's or its suppliers' or licensors' liability to Customer, whether in contract, tort (including negligence), breach of warranty, or otherwise, exceed the price paid by Customer for the Software that gave rise to the claim, or if the Software is embedded in another Juniper product, the price paid by Customer for such other product. Customer acknowledges and agrees that Juniper has set its prices and entered into this Agreement in reliance upon the disclaimers of warranty and the limitations of liability set forth herein, that the same reflect an allocation of risk between the Parties (including the risk that a contract remedy may fail of its essential purpose and cause consequential loss), and that the same form an essential basis of the bargain between the Parties.

9. **Termination.** Any breach of this Agreement or failure by Customer to pay any applicable fees due shall result in automatic termination of the license granted herein. Upon such termination, Customer shall destroy or return to Juniper all copies of the Software and related documentation in Customer's possession or control.

10. **Taxes.** All license fees payable under this agreement are exclusive of tax. Customer shall be responsible for paying Taxes arising from the purchase of the license, or importation or use of the Software. If applicable, valid exemption documentation for each taxing jurisdiction shall be provided to Juniper prior to invoicing, and Customer shall promptly notify Juniper if their exemption is revoked or modified. All payments made by Customer shall be net of any applicable withholding tax. Customer will provide reasonable assistance to Juniper in connection with such withholding taxes by promptly: providing Juniper with valid tax receipts and other required documentation showing Customer's payment of any withholding taxes; completing appropriate applications that would reduce the amount of withholding tax to be paid; and notifying and assisting Juniper in any audit or tax proceeding related to transactions hereunder. Customer shall comply with all applicable tax laws and regulations, and Customer will promptly pay or reimburse Juniper for all costs and damages related to any liability incurred by Juniper as a result of Customer's non-compliance or delay with its responsibilities herein. Customer's obligations under this Section shall survive termination or expiration of this Agreement.

11. **Export.** Customer agrees to comply with all applicable export laws and restrictions and regulations of any United States and any applicable foreign agency or authority, and not to export or re-export the Software or any direct product thereof in violation of any such restrictions, laws or regulations, or without all necessary approvals. Customer shall be liable for any such violations. The version of the Software supplied to Customer may contain encryption or other capabilities restricting Customer's ability to export the Software without an export license.

12. **Commercial Computer Software.** The Software is "commercial computer software" and is provided with restricted rights. Use, duplication, or disclosure by the United States government is subject to restrictions set forth in this Agreement and as provided in DFARS 227.7201 through 227.7202-4, FAR 12.212, FAR 27.405(b)(2), FAR 52.227-19, or FAR 52.227-14 (ALT III) as applicable.

13. **Interface Information.** To the extent required by applicable law, and at Customer's written request, Juniper shall provide Customer with the interface information needed to achieve interoperability between the Software and another independently created program, on payment of applicable fee, if any. Customer shall observe strict obligations of confidentiality with respect to such information and shall use such information in compliance with any applicable terms and conditions upon which Juniper makes such information available.

14. **Third Party Software.** Any licensor of Juniper whose software is embedded in the Software and any supplier of Juniper whose products or technology are embedded in (or services are accessed by) the Software shall be a third party beneficiary with respect to this Agreement, and such licensor or vendor shall have the right to enforce this Agreement in its own name as if it were Juniper. In addition, certain third party software may be provided with the Software and is subject to the accompanying license(s), if any, of its respective owner(s). To the extent portions of the Software are distributed under and subject to open source licenses obligating Juniper to make the source code for such portions publicly available (such as the GNU General Public License ("GPL") or the GNU Library General Public License ("LGPL")), Juniper will make such source code portions (including Juniper modifications, as appropriate) available upon request for a period of up to three years from the date of distribution. Such request can be made in writing to Juniper Networks, Inc., 1194 N. Mathilda Ave., Sunnyvale, CA 94089, ATTN: General Counsel. You may obtain a copy of the GPL at <http://www.gnu.org/licenses/gpl.html>, and a copy of the LGPL at <http://www.gnu.org/licenses/lgpl.html>.

15. **Miscellaneous.** This Agreement shall be governed by the laws of the State of California without reference to its conflicts of laws principles. The provisions of the U.N. Convention for the International Sale of Goods shall not apply to this Agreement. For any disputes arising under this Agreement, the Parties hereby consent to the personal and exclusive jurisdiction of, and venue in, the state and federal courts within Santa Clara County, California. This Agreement constitutes the entire and sole agreement between Juniper and the Customer with respect to the Software, and supersedes all prior and contemporaneous agreements relating to the Software, whether oral or written (including any inconsistent terms contained in a purchase order), except that the terms of a separate written agreement executed by an authorized Juniper representative and Customer shall govern to the extent such terms are inconsistent or conflict with terms contained herein. No modification to this Agreement nor any waiver of any rights hereunder shall be effective unless expressly assented to in writing by the party to be charged. If any portion of this Agreement is held invalid, the Parties agree that such invalidity shall not affect the validity of the remainder of this Agreement. This Agreement and associated documentation has been written in the English language, and the Parties agree that the English version will govern. (For Canada: Les parties aux présentes confirment leur volonté que cette convention de même que tous les documents y compris tout avis qui s'y rattache, soient rédigés en langue anglaise. (Translation: The parties confirm that this Agreement and all related documentation is and will be in the English language)).



# Table of Contents

	<b>Preface</b> .....	<b>xiii</b>
	Objectives .....	xiii
	Audience .....	xiii
	Documentation Conventions .....	xiii
	Related Documentation .....	xv
	Documentation Feedback .....	xvi
	Requesting Technical Support .....	xvi
	Self-Help Online Tools and Resources .....	xvi
	Opening a Case with JTAC .....	xvii
<b>Chapter 1</b>	<b>Workflow: Creating a Custom Attack Object</b> .....	<b>1</b>
	Creating Custom Attack Objects Overview .....	1
	Understanding Our Approach to Addressing Known and Unknown	
	Vulnerabilities .....	2
	Known Vulnerabilities .....	2
	Unknown Vulnerabilities .....	3
	Setting Up a Test Lab for Custom Attack Objects .....	3
	Guidelines .....	3
	Topology .....	3
	Requirements .....	4
	Packet Capture and Replay Tools .....	4
	Reproducing an Attack .....	6
	Replaying a Packet Capture .....	7
	Running Attack Code .....	7
	Discovering the Attack Signature .....	8
	Creating a Signature Attack Object .....	9
	Creating a Compound Attack Object .....	29
	Testing a Custom Attack Object .....	33
<b>Chapter 2</b>	<b>Examples: Custom Attack Object Pattern Syntax</b> .....	<b>35</b>
	Example: Custom Attack Object DFA Expressions .....	35
	Example: Using Pattern Negation .....	37
	Example: Matching File Extensions .....	37
	Modifying Custom Attack Objects Due to Changes Introduced in Signature	
	Update #1972 .....	38
	Reference: Removed Contexts .....	38
	Example: Replacing the Context for Patterns Appearing in HTML Text .....	39
	Example: Replacing the Contexts for Patterns Appearing in URLs .....	39
	Signatures that Match Request Methods .....	39
	Signatures that Match URL Strings and URL Variables .....	40

<b>Chapter 3</b>	<b>Examples: Using Matching Constraints . . . . .</b>	<b>43</b>
	Example: Using Time Binding Parameters to Detect a Brute Force Attack . . . . .	43
	Example: Using the Within Bytes or Within Packets Constraint with Signature Attack Objects . . . . .	45
	Example: Using the Within Packets Constraint with Compound Attack Objects . . . . .	45
	Example: Using the Same Context Constraint with Compound Attack Objects . . . . .	46
	Example: Using Context Length Checking to Optimize Performance . . . . .	46
<b>Chapter 4</b>	<b>Case Studies: Creating Custom Attack Objects to Respond to Vulnerability Reports . . . . .</b>	<b>47</b>
	Example: Apache Tomcat Denial-of-Service Attacks . . . . .	47
	Example: UNIX CDE/dtlogin Vulnerability . . . . .	49
	Example: Detecting a Worm . . . . .	53
	Example: Compound Signature to Detect Exploitation of an HTTP Vulnerability . . . . .	55
<b>Chapter 5</b>	<b>Reference Tables . . . . .</b>	<b>61</b>
	Reference: Custom Attack Object Protocol Numbers . . . . .	61
	Reference: Custom Attack Object Service Properties . . . . .	67
	Reference: Custom Attack Object Service Contexts . . . . .	70
	Reference: Nonprintable and Printable ASCII Characters . . . . .	106
<b>Chapter 6</b>	<b>Index . . . . .</b>	<b>119</b>
	Index . . . . .	121



# List of Figures

<b>Chapter 1</b>	<b>Workflow: Creating a Custom Attack Object</b> . . . . .	<b>1</b>
	Figure 1: Custom Attack Object Test Lab . . . . .	4
	Figure 2: Custom Attack Object: General Tab . . . . .	10
	Figure 3: Custom Attack Object: Extended Tab . . . . .	11
	Figure 4: Custom Attack: Target Platform and Type Page . . . . .	13
	Figure 5: Custom Attack – General Properties Page . . . . .	15
	Figure 6: Custom Attack – Attack Pattern Page . . . . .	19
	Figure 7: Custom Attack – IP Settings and Header Matches Page . . . . .	23
	Figure 8: Custom Attack Object: TCP Packet Header Fields . . . . .	25
	Figure 9: Custom Attack Object: UDP Packet Header Fields . . . . .	27
	Figure 10: Custom Attack Object: ICMP Packet Header Fields . . . . .	28
	Figure 11: Custom Attack – Compound Members . . . . .	31
<b>Chapter 3</b>	<b>Examples: Using Matching Constraints</b> . . . . .	<b>43</b>
	Figure 12: Brute Force Attack Pattern . . . . .	44
	Figure 13: Brute Force Time Binding . . . . .	44
<b>Chapter 4</b>	<b>Case Studies: Creating Custom Attack Objects to Respond to Vulnerability Reports</b> . . . . .	<b>47</b>
	Figure 14: CAN2002 Packet Capture . . . . .	48
	Figure 15: CAN-2002-0682 Attack Pattern . . . . .	49
	Figure 16: XDMP Packet Capture . . . . .	51
	Figure 17: XDMP Attack Pattern . . . . .	52
	Figure 18: Blaster Worm Packet Capture . . . . .	54
	Figure 19: Blaster Worm Attack Pattern . . . . .	55
	Figure 20: HTTP Packet Capture . . . . .	57
	Figure 21: First Attack Pattern . . . . .	58
	Figure 22: Second Attack Pattern . . . . .	59



# List of Tables

	<b>Preface</b> .....	<b>xiii</b>
	Table 1: Notice Icons .....	xiv
	Table 2: Text Conventions .....	xiv
	Table 3: Syntax Conventions .....	xv
	Table 4: Related IDP Series Documentation .....	xv
<b>Chapter 1</b>	<b>Workflow: Creating a Custom Attack Object</b> .....	<b>1</b>
	Table 5: Test Lab Components .....	4
	Table 6: Packet Capture and Replay Tools .....	5
	Table 7: Attack Object Properties .....	8
	Table 8: Custom Attack Dialog Box: General Tab Settings .....	10
	Table 9: Custom Attack Dialog Box: Extended Tab Settings .....	12
	Table 10: Attack Object Types .....	14
	Table 11: Custom Attack – General Properties .....	15
	Table 12: Custom Attack – Attack Pattern .....	19
	Table 13: Custom Attack – IP Settings and Header Matches Page .....	24
	Table 14: Custom Attack Object: TCP Packet Header Fields .....	25
	Table 15: Custom Attack Object: UDP Header Fields .....	27
	Table 16: Custom Attack Object: ICMP Packet Header Fields .....	28
	Table 17: Custom Attack – General Properties .....	30
	Table 18: Compound Attack Parameters .....	31
<b>Chapter 2</b>	<b>Examples: Custom Attack Object Pattern Syntax</b> .....	<b>35</b>
	Table 19: Example: Custom Attack Object Regular Expressions .....	35
	Table 20: HTTP Service Contexts .....	38
	Table 21: HTTP Service Contexts: HTML Text .....	39
	Table 22: HTTP Service Contexts: Request Methods Before Update .....	40
	Table 23: HTTP Service Contexts: Request Methods After Update .....	40
	Table 24: HTTP Service Contexts: URL Strings and Variables Before Update ...	40
	Table 25: HTTP Service Contexts: URL Strings and Variables After Update ....	40
<b>Chapter 5</b>	<b>Reference Tables</b> .....	<b>61</b>
	Table 26: IDP Attack Objects: Protocol Numbers .....	61
	Table 27: IDP Attack Object: Service Properties .....	67
	Table 28: Service Contexts: AIM .....	70
	Table 29: Service Contexts: BGP .....	72
	Table 30: Service Contexts: DHCP .....	73
	Table 31: Service Contexts: DNS .....	73
	Table 32: Service Contexts: Finger .....	75
	Table 33: Service Contexts: First Data Packet .....	75
	Table 34: Service Contexts: FTP .....	75

Table 35: Service Contexts: Gnutella .....	77
Table 36: Service Contexts: Gopher .....	77
Table 37: Service Contexts: H225 .....	77
Table 38: Service Contexts: HTTP .....	78
Table 39: Service Contexts: IEC .....	81
Table 40: Service Contexts: IMAP .....	81
Table 41: Service Contexts: IRC .....	82
Table 42: Service Contexts: LDAP .....	83
Table 43: Service Contexts: Line .....	85
Table 44: Service Contexts: LPR .....	85
Table 45: Service Contexts: MGCP .....	85
Table 46: Service Contexts: Modbus .....	86
Table 47: Service Contexts: MSN .....	86
Table 48: Service Contexts: MSRPC .....	87
Table 49: Service Contexts: MS-SQL .....	87
Table 50: Service Contexts: MySQL .....	88
Table 51: Service Contexts: NetBIOS .....	89
Table 52: Service Contexts: NFS .....	89
Table 53: Service Contexts: NNTP .....	90
Table 54: Service Contexts: Normalized Stream .....	90
Table 55: Service Contexts: NTP .....	91
Table 56: Service Contexts: Packet .....	91
Table 57: Service Contexts: POP3 .....	91
Table 58: Service Contexts: RADIUS .....	92
Table 59: Service Contexts: REXEC .....	94
Table 60: Service Contexts: RLOGIN .....	94
Table 61: Service Contexts: RSH .....	95
Table 62: Service Contexts: RUSERS .....	95
Table 63: Service Contexts: SIP .....	95
Table 64: Service Contexts: SMB .....	96
Table 65: Service Contexts: SMTP .....	100
Table 66: Service Contexts: SNMP .....	101
Table 67: Service Contexts: SSH .....	102
Table 68: Service Contexts: SSL .....	102
Table 69: Service Contexts: Stream .....	103
Table 70: Service Contexts: Telnet .....	103
Table 71: Service Contexts: TFTP .....	103
Table 72: Service Contexts: TNS .....	103
Table 73: Service Contexts: VNC .....	104
Table 74: Service Contexts: YMSG .....	104
Table 75: ASCII Reference: Nonprintable Characters .....	106
Table 76: ASCII Reference: Printable Characters .....	108

# Preface

This preface includes the following topics:

- Objectives on page xiii
- Audience on page xiii
- Documentation Conventions on page xiii
- Related Documentation on page xv
- Documentation Feedback on page xvi
- Requesting Technical Support on page xvi

## Objectives

---

This guide provides reference tables and examples of configuration settings for user-defined attack objects.

## Audience

---

This guide is intended for network security experts. We expect most IDP Series customers to use the predefined attack objects provided by Juniper Networks Security Center (J-Security Center). We support user-defined attack objects for network security experts who have experience analyzing network packets and have requested the ability to configure attack object properties.

## Documentation Conventions

---

This section provides all the documentation conventions that are followed in this guide. Table 1 on page xiv defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.

Table 2 on page xiv defines text conventions used in this guide.

Table 2: Text Conventions

Convention	Description	Examples
<b>Bold typeface like this</b>	<ul style="list-style-type: none"> <li>Represents commands and keywords in text.</li> <li>Represents keywords</li> <li>Represents UI elements</li> </ul>	<ul style="list-style-type: none"> <li>Issue the <b>clock source</b> command.</li> <li>Specify the keyword <b>exp-msg</b>.</li> <li>Click <b>User Objects</b></li> </ul>
<b>Bold typeface like this</b>	Represents text that the user must type.	<b>user input</b>
fixed-width font	Represents information as displayed on the terminal screen.	<pre>host1# show ip ospf Routing Process OSPF 2 with Router ID 5.5.0.250 Router is an area Border Router (ABR)</pre>
Key names linked with a plus (+) sign	Indicates that you must press two or more keys simultaneously.	Ctrl + d
<i>Italics</i>	<ul style="list-style-type: none"> <li>Emphasizes words</li> <li>Identifies variables</li> </ul>	<ul style="list-style-type: none"> <li>The product supports two levels of access, <i>user</i> and <i>privileged</i>.</li> <li><i>clusterID</i>, <i>ipAddress</i>.</li> </ul>
The angle bracket (>)	Indicates navigation paths through the UI by clicking menu options and links.	<b>Object Manager &gt; User Objects &gt; Local Objects</b>

Table 3 on page xv defines syntax conventions used in this guide.

Table 3: Syntax Conventions

Convention	Description	Examples
Words in plain text	Represent keywords	terminal length
Words in italics	Represent variables	<i>mask</i> , <i>accessListName</i>
Words separated by the pipe (   ) symbol	Represent a choice to select one keyword or variable to the left or right of this symbol. The keyword or variable can be optional or required.	diagnostic   line
Words enclosed in brackets ( [ ] )	Represent optional keywords or variables.	[ internal   external ]
Words enclosed in brackets followed by and asterisk ( [ ]*)	Represent optional keywords or variables that can be entered more than once.	[ level1   level2   11 ]*
Words enclosed in braces ( { } )	Represent required keywords or variables.	{ permit   deny } { in   out } { clusterId   ipAddress }

## Related Documentation

Table 4 on page xv lists related IDP Series documentation.

Table 4: Related IDP Series Documentation

Document	Description
<a href="#">Release notes</a>	Contains information about what is included in a specific product release: supported features, unsupported features, changed features, known problems, and resolved problems. If the information in the release notes differs from the information found in the documentation set, follow the release notes.
<a href="#">IDP Detector Engine release notes</a>	Provides information about IDP Detector Engine releases, including new features, changed features, fixed problems, and known issues.
<a href="#">J-Security Center Attack Signatures</a>	Lists predefined attack signatures developed by J-Security Center.
<a href="#">J-Security Center Application Signatures</a>	Lists predefined application signatures developed by J-Security Center.
<a href="#">IDP Series installation guides</a>	Describes IDP Series hardware and provides instructions for installing, configuring, updating, and servicing the device.
<a href="#">IDP Series Feature Documentation</a>	A collection of topics from the <i>IDP Series Administration Guide</i> and <i>IDP Series Concepts and Examples Guide</i> , in HTML.
<a href="#">IDP Series Administration Guide</a>	Provides procedures for completing IDP Series administration tasks with the Network and Security Manager (NSM) central management program; with the IDP Series device Appliance Configuration Manager (ACM); and with the IDP Series device command-line interface (CLI).
<a href="#">IDP Series Concepts and Examples Guide</a>	Explains IDP Series features and provides examples of how to use the system.

Table 4: Related IDP Series Documentation (*continued*)

Document	Description
<i>IDP Series Custom Attack Objects Reference and Examples Guide</i>	Provides examples and reference information for creating custom attack objects.
<i>IDP Reporter User's Guide</i>	Describes how to use IDP Reporter, an on-box reporting platform that includes predefined reports on attack detection and application usage. You can also use IDP Reporter to schedule regular publication of reports that are of interest to you or your stakeholders.

## Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to [techpubs-comments@juniper.net](mailto:techpubs-comments@juniper.net), or fill out the documentation feedback form at <https://www.juniper.net/cgi-bin/docbugreport/>. If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version (if applicable)

## Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <http://www.juniper.net/support/warranty/>.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

## Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: <http://www.juniper.net/customers/support/>
- Search for known bugs: <http://www2.juniper.net/kb/>



- Find product documentation: <http://www.juniper.net/techpubs/>
- Find solutions and answer questions using our Knowledge Base: <http://kb.juniper.net/>
- Download the latest versions of software and review release notes:  
<http://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications:  
<https://www.juniper.net/alerts/>
- Join and participate in the Juniper Networks Community Forum:  
<http://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <http://www.juniper.net/cm/>

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: <https://tools.juniper.net/SerialNumberEntitlementSearch/>

## Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at <http://www.juniper.net/cm/> .
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see <http://www.juniper.net/support/requesting-support.html> .



## CHAPTER 1

# Workflow: Creating a Custom Attack Object

The following topics describe a suggested workflow for creating a custom attack object:

- [Creating Custom Attack Objects Overview on page 1](#)
- [Understanding Our Approach to Addressing Known and Unknown Vulnerabilities on page 2](#)
- [Setting Up a Test Lab for Custom Attack Objects on page 3](#)
- [Reproducing an Attack on page 6](#)
- [Discovering the Attack Signature on page 8](#)
- [Creating a Signature Attack Object on page 9](#)
- [Creating a Compound Attack Object on page 29](#)
- [Testing a Custom Attack Object on page 33](#)

## Creating Custom Attack Objects Overview

---

The task of creating a custom attack object involves the following basic steps:

1. Become familiar with our approach to threat prevention.
2. Set up a test lab.
3. Reproduce the attack in your lab.
4. Discover the signature.
5. Create a signature or compound attack object.
6. Test the attack object.

### Related Documentation

- [Understanding Our Approach to Addressing Known and Unknown Vulnerabilities on page 2](#)
- [Setting Up a Test Lab for Custom Attack Objects on page 3](#)
- [Reproducing an Attack on page 6](#)
- [Discovering the Attack Signature on page 8](#)

- Creating a Signature Attack Object on page 9
- Creating a Compound Attack Object on page 29
- Testing a Custom Attack Object on page 33

## Understanding Our Approach to Addressing Known and Unknown Vulnerabilities

---

This topic includes the following sections:

- Known Vulnerabilities on page 2
- Unknown Vulnerabilities on page 3

### Known Vulnerabilities

---

Known vulnerabilities are those documented within the Internet security community. The Internet security community comprises several security organizations, security analysts, and security forums. The security community continually discovers and analyzes new attacks and exchanges this information over the Internet. In this way, they can quickly locate, identify, and truly understand an attack.

Some security advisories include the actual attack code. You can use the attack information and the attack code to capture packet information and service contexts. You can use this information to create a custom signature attack object.

Unfortunately, most advisories do not post the attack code with the attack description. If you cannot obtain the attack code, read the advisory carefully and try to reconstruct the basics of the attack packet.



**CAUTION:** Remember to isolate code acquired from unknown sources.

---

The following organizations are active in the security community and are a good resource for locating attack information:

- NVD—National Vulnerability Database (<http://nvd.nist.gov>). The U.S. government repository of vulnerability management data represented using the Security Content Automation Protocol (SCAP).
- SANS—SysAdmin, Audit, Network, Security Institute ([www.sans.org](http://www.sans.org)). An information security research, certification, and education organization that provides security alerts. Also hosts the Internet Storm Center (ISC) at <http://www.incidents.org>.
- CVE—Common Vulnerabilities and Exposures (<http://cve.mitre.org>). A standardized list of vulnerabilities and other information security exposures.
- BugTraq (<http://securityfocus.com/archive/1>). A moderated mailing list hosted by Security Focus that discusses and announces computer security vulnerabilities.
- CERT coordination center (<http://www.cert.org>). A federally funded security alert organization that provides security advisories.

- Packet Storm Security (<http://packetstormsecurity.nl>). A nonprofit organization of security professionals that provides security information by way of security news, advisories, forums, and attack code.
- Metasploit (<http://www.metasploit.com>). Metasploit provides useful information for performing penetration testing, IDS signature development, and exploit research.
- FrSIRT—French Security Incident Response Team (<http://www.frsirt.com>). FrSIRT is an independent security research organization providing security advisories and real-time vulnerability alerting and notification services.
- ISS—Internet Security Systems (<http://www.iss.net>). An Internet security company that provides alerts and Internet threat levels.

---

### Unknown Vulnerabilities

Unknown vulnerabilities are those that have not been documented in Internet security community advisories. In these cases, the IDP Series Profiler, firewall, or IDP security event logs generated in your production environment alert you to suspicious activity and abnormal traffic. In your production environment, you will use packet logging tools to capture packets and service context information that you can later analyze and experiment with in your lab.

#### Related Documentation

- Creating Custom Attack Objects Overview on page 1

---

## Setting Up a Test Lab for Custom Attack Objects

This topic includes the following sections:

- Guidelines on page 3
- Topology on page 3
- Requirements on page 4
- Packet Capture and Replay Tools on page 4

---

### Guidelines

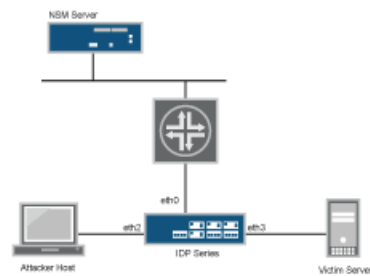
We recommend that you test and fine-tune custom attack objects in a laboratory environment that is not connected to your corporate network.

---

### Topology

Figure 1 on page 4 shows an example lab topology.

Figure 1: Custom Attack Object Test Lab



## Requirements

Table 5 on page 4 describes the purpose of test lab components.

Table 5: Test Lab Components

Component	Purpose
Attacker computer	You use packet generation tools to run exploit code from the attacker computer to the victim computer.
Victim computer	You use a packet capture utility, such as <a href="#">Wireshark</a> , to capture packets received by victim computer.
IDP device	<p>You use the <b>scio ccap</b> context capture utility (included with IDP OS software) to discover the attack context and pattern.</p> <p>We recommend you use IDP OS Release 5.1 or later.</p>
NSM server/client	<p>You use the NSM object editor to create and test the custom attack object.</p> <p>We recommend you use NSM 2010.4 or later.</p>
Hub or switch	Typically, you deploy an IDP device in transparent mode between a switch and a firewall. Alternatively, you can deploy the IDP device in sniffer mode, connected only to the SPAN port of a switch. We recommend you experiment with both deployment possibilities.

## Packet Capture and Replay Tools

Table 6 on page 5 describes packet capture and replay tools you use to reproduce the attack.

Table 6: Packet Capture and Replay Tools

Tool	Description
<b>scio ccap</b>	<p>An IDP OS <b>scio</b> utility. Use <b>scio ccap</b> to capture attack contexts.</p> <pre>[root@defaulthost ~]# scio ccap</pre> <p>Usage: <b>scio ccap</b> &lt;ccap-command&gt; [arguments...]</p> <p>ccap-commands:</p> <ul style="list-style-type: none"> <li><b>all</b> : Capture all contexts</li> <li><b>sip</b> &lt;ipaddr&gt;[/length]: Capture contexts for sessions from &lt;ipaddr&gt;[/length]</li> <li><b>dip</b> &lt;ipaddr&gt;[/length]: Capture contexts for sessions to &lt;ipaddr&gt;[/length]</li> <li><b>svc</b> &lt;service&gt;: Capture contexts for sessions that match &lt;service&gt;</li> </ul> <p>Keep the following notes in mind:</p> <ul style="list-style-type: none"> <li>• Issuing the <b>scio ccap</b> command starts context collection. Press <b>Ctrl-C</b> to end context collection.</li> <li>• Use standard Linux redirect-to-file syntax to direct screen output to a file (for example, <b>scio ccap &gt; file.txt</b>). Otherwise, results are sent to the screen.</li> </ul> <p>You can use the <b>all</b> option in lab environment, where you can control the network traffic. In a production environment, results using <b>all</b> are likely unmanageable, so we recommend you use the <b>sip</b> or <b>dip</b> option to capture the contexts for traffic that has a specific source or destination.</p>
<b>scio pcap</b>	<p>An IDP OS <b>scio</b> utility. Use <b>scio pcap</b> to replay a packet capture file.</p> <pre>[root@defaulthost ~]# scio pcap</pre> <p>Usage: <b>scio pcap</b> &lt;subs-name&gt; &lt;vc-name&gt; &lt;file&gt; [command [&lt;options&gt;]]</p> <p>commands:</p> <ul style="list-style-type: none"> <li><b>filter</b> &lt;options&gt;: specify tcpdump style filtering options</li> <li><b>start</b> &lt;options&gt;: specify packet number to start at and count of packets</li> <li><b>instrument</b> : instrument ids pattern matching</li> </ul> <p>For best results:</p> <ul style="list-style-type: none"> <li>• Stop the Profiler service before running the <b>scio pcap</b> utility. To do this, use the <b>profiler.sh stop</b> command. To restore the Profiler service, use the <b>profiler.sh restart</b> command.</li> <li>• When you replay packet captures, allow 30 seconds to elapse between <b>scio pcap</b> commands.</li> </ul>
Wireshark	<p>Wireshark is a popular program used to analyze network traffic. You can use Wireshark to create and view packet captures. For more information about Wireshark, see <a href="http://www.wireshark.org/">http://www.wireshark.org/</a>.</p>
Tcpreplay	<p>Tcpreplay is a set of BSD-licensed packet replay utilities. For more information about Tcpreplay, see <a href="http://tcpreplay.synfin.net/">http://tcpreplay.synfin.net/</a>.</p>

Table 6: Packet Capture and Replay Tools (*continued*)

Tool	Description
<b>tcpdump</b>	<p>Linux-based version of the commonly used packet capture utility. On IDP Series devices, you can use <b>tcpdump</b> to capture packets received (Rx packets) by the device.</p> <pre>[root@defaulthost ~]# man tcpdump Formatting page, please wait... TCPDUMP(8)</pre> <p>NAME</p> <p>tcpdump - dump traffic on a network</p> <p>SYNOPSIS</p> <pre>tcpdump [ -AdDeflLnNOpqRStuUvxX ] [ -c count ] [ -C file_size ] [ -F file ] [ -i interface ] [ -m module ] [ -M secret ] [ -r file ] [ -s snaplen ] [ -T type ] [ -w file ] [ -W filecount ] [ -E spi@ipaddr algo:secret,... ] [ -y datalinktype ] [ -Z user ] [ expression ]</pre>
<b>jnetTcpdump</b>	<p>An IDP OS Release 5.1 utility. Capable of capturing both Rx and Tx packets. The following example starts listening on eth4 for packets with destination IP address 4.0.0.4:</p> <pre>[root@localhost ~]# jnetTcpdump -i eth4 -f 4.0.0.4 dst jnetPassiveAttach done jnet tcpdump Started on eth4 for both Receive &amp; Transmit side Filter enabled - Host:4.0.0.4 as dst 0 50 56 a4 21 6c 0 50 56 a4 d 9 8 0 45 0 0 54 0 0 40 0 40 1 32 a3 4 0 0 3 4 0 0 4 8 0 55 8e 8e 4f 0 0 ba 9f 3e 4d 21 32 f 0 8 9 a b c d e f 10 11 12 13 14 15 0 50 56 a4 21 6c 0 50 56 a4 d 9 8 0 45 0 0 54 0 0 40 0 40 1 32 a3 4 0 0 3 4 0 0 4 8 0 97 88 8e 4f 0 1  bb 9f 3e 4d de 36 f 0 8 9 a b c d e f 10 11 12 13 14 15 Done...No of Packet Captured is 2 No of Packets filtered-out 2</pre>

**Related Documentation**

- [Creating Custom Attack Objects Overview on page 1](#)

## Reproducing an Attack

Reproducing the attack in your lab enables you to generate a context you can analyze to determine the signature elements described in “Discovering the Attack Signature” on page 8. A TCP session must include the TCP three-way handshake in order to generate contexts accurately. You can reproduce attack traffic in one of the following ways:

- [Replaying a Packet Capture on page 7](#)
- [Running Attack Code on page 7](#)



## Replaying a Packet Capture

---

In some cases, the security bulletin that prompts you to guard against a vulnerability includes a packet capture to illustrate traffic that can exploit a vulnerability.

To reproduce an attack when you have a packet capture:

1. Copy the *filename.pcap* file to the IDP Series device /tmp directory.
2. Open two separate SSH or virtual terminal connections to the IDP CLI (two separate windows).
3. In one window, enter the following command to start the context capture:

```
[user@host]% scio ccap all
```

4. In another window, enter the following command to replay the *filename.pcap* file:

```
[user@host]% scio pcap s0 eth0 filename.pcap
```

5. Switch to the context capture window. When you are ready to terminate the context capture, press **Ctrl-C**.
6. Review the results:
  - If the attack uses service contexts known to the IDP system, the contexts are displayed on the screen. For a reference of known contexts, see “Reference: Custom Attack Object Service Contexts” on page 70.
  - If the attack does not use service contexts known to the IDP system, the results are empty.

## Running Attack Code

---

In some cases, the security bulletin that prompts you to guard against a vulnerability includes details of the attack code used to exploit the vulnerability.

To reproduce an attack when you do not have a packet capture but you do have the attack code:

1. Copy the attack code to your test lab attacker computer. If necessary, compile the code.
2. Log in to the IDP Series device CLI and enter the following command to start the context capture:

```
[user@host]% scio ccap all
```
3. On the test lab victim computer, start a Wireshark capture. Be sure you to specify the interface that receives traffic from the IDP Series device.
4. On the test lab attacker computer, run the attack code.
5. Switch to the context capture window. When you are ready to terminate the context capture, press **Ctrl-C**.

## 6. Review the results:

- If the attack uses service contexts known to the IDP system, the contexts are displayed on the screen. For a reference of known contexts, see “Reference: Custom Attack Object Service Contexts” on page 70.
  - If the attack does not use service contexts known to the IDP system, the results are empty.
7. On the test lab victim computer, review the Wireshark Captured Frames progress box. The progress box displays a list of services and the number of frame captures for each service. When the capture limits have been met, the progress box closes and Wireshark displays the packet capture. Use Wireshark utilities to analyze and save the packet capture.

**Related  
Documentation**

- Creating Custom Attack Objects Overview on page 1

## Discovering the Attack Signature

The purpose of your investigation is to identify the attack pattern so you define attack object properties to match it. Table 7 on page 8 describes the properties you want to identify.

**Table 7: Attack Object Properties**

Property	Description
Service	<p>The protocol where the attack is found. Use a context capture utility, such as <b>scio ccap</b>, to identify the service. Research the protocol well enough to be able to compare a normal session with an anomalous session. Be knowledgeable of start and end procedures, service contexts, and payload transfer.</p> <p>For a list of services we can match, see “Reference: Custom Attack Object Service Properties” on page 67.</p>
Context	<p>The segment of the communication where the attack signature is always found. Use a context capture utility to determine the context. The more precise you can be, the less likely you are to encounter false positives.</p> <p>For a list of service contexts we can match, see “Reference: Custom Attack Object Service Contexts” on page 70. If you are unable to determine the context with <b>scio ccap</b>, you can configure a context based on packet, stream, or line.</p>
Direction	<p>Client-to-server, server-to-client, or both. Use a packet capture utility, such as Wireshark, to examine the session flow.</p> <p>The session initiator is considered the client, even if that source IP is a server.</p> <p>Most attacks are found in client-to-server traffic. You can use the Wireshark Follow TCP Stream or Hex Dump feature to investigate session details and determine direction with certainty.</p> <p>For information about the typical direction for a given context, see “Reference: Custom Attack Object Service Contexts” on page 70.</p>

Table 7: Attack Object Properties (*continued*)

Property	Description
Pattern or signature	<p>A DFA expression. Use a packet capture utility to examine the payload. Identify a pattern that always exists within the context of an attack. Consult a regular expression site for information about DFA.</p> <p>For a quick reference of hexadecimal and octal numbers for printable and nonprintable characters, see "Reference: Nonprintable and Printable ASCII Characters" on page 106.</p>

**Related Documentation**

- [Creating Custom Attack Objects Overview on page 1](#)

## Creating a Signature Attack Object

A signature attack object is a pattern you want the system to detect. You use a DFA expression to represent the pattern. All of the other signature properties you can set (such as service or protocol context, direction, and other constraints) are provided so you can optimize performance of the system in detecting the pattern and eliminate false positives. In general, you want to tune settings of a signature attack object so that the system looks for it in every context where it might occur and in no other context.

To configure a signature attack object:

1. In the Object Manager, select **Attack Objects > IDP Objects**.
2. Click the **Custom Attacks** tab.
3. Click the + icon to display the Custom Attack dialog box.
4. Configure attack object settings. Figure 2 on page 10 shows the General tab. Table 8 on page 10 provides guidelines for completing the settings.

Figure 2: Custom Attack Object: General Tab

**New - Custom Attack**

General Extended

Name

Description

Severity Info

Category

Keywords

Recommended ☐

Attack Versions

Platform	SignaturePack	Type	Details
----------	---------------	------	---------

Detection Performance Not defined

OK Cancel

Table 8: Custom Attack Dialog Box: General Tab Settings

Setting	Description
Name	The name displayed in the UI.  <i>TIP:</i> Include the protocol the attack uses as part of the attack name.
Description	(Optional) Information about the attack. Although a description is optional when you create a new attack object, it can help you remember important information about the attack. For examples, view the attack descriptions for predefined attacks.
Severity	Info, Warning, Minor, Major, or Critical. Critical attacks are attempts to crash your server or gain control of your network. Informational attacks are the least dangerous and typically are used by network administrators to discover holes in their own security system.
Category	A predefined or new category.

Table 8: Custom Attack Dialog Box: General Tab Settings (*continued*)

Setting	Description
Keywords	Unique identifiers that can be used to search and sort log records.
Recommended	Indicates that this attack object is among your highest-risk set of attack objects. Later, when you add this attack object to dynamic groups, you can specify whether to include only recommended attack objects.
Attack Versions	Skip this for now.
Detection Performance	Select <b>High</b> , <b>Medium</b> , <b>Low</b> , or <b>Not Defined</b> .

5. Configure additional attack details on the Extended tab. Figure 3 on page 11 shows the Extended tab. Table 9 on page 12 provides guidelines for completing the settings.

Figure 3: Custom Attack Object: Extended Tab

The screenshot shows the 'New - Custom Attack' dialog box with the 'Extended' tab selected. The dialog is titled 'New - Custom Attack' and has a close button (X) in the top right corner. The 'General' tab is also visible but not selected. The 'Extended' tab contains the following sections and fields:

- Specify web sites (using URLs) that provide details about this attack.**
  - Primary URL:
  - Secondary URL:
  - Tertiary URL:
- Standard References**
  - CVE:
  - BugTraq:  (dropdown menu)
- Provide more detailed information about this attack.**
  - Impact:
  - Description:
  - Tech Info:
  - Patches:
- Hint:** HTML tags can be used to include links etc.

At the bottom right of the dialog are 'OK' and 'Cancel' buttons.

Table 9: Custom Attack Dialog Box: Extended Tab Settings

Setting	Description
Primary URL	Up to three URLs (primary, secondary, tertiary) to external references you used to research the attack.
Secondary URL	
Tertiary URL	
CVE	The Common Vulnerabilities and Exposures (CVE) ID that the attack object addresses. CVE is a standardized list of vulnerabilities and other information security exposures. The CVE number is an alphanumeric code, such as CVE-2209.
BugTraq	The BugTraq ID number that the attack object addresses. BugTraq is a moderated mailing list that discusses and announces computer security vulnerabilities. The BugTraq ID number is a three-digit code, such as 831 or 120.
Impact	Information about the impact of a successful attack, including information about system crashes and access granted to the attacker.
Description	Additional information.
Tech Info	Information about the vulnerability, the commands used to execute the attack, which files are attacked, registry edits, and other low-level information.
Patches	Any patches available from the product vendor, as well as information about how to prevent the attack.

6. Click the **General** tab.
7. Under Attack Versions, click the + icon to display the New Attack wizard.
8. On the Target Platform and Type page, select a device platform and attack type. Figure 4 on page 13 shows the Target Platform and Type page. Table 10 on page 14 describes the attack types.

Figure 4: Custom Attack: Target Platform and Type Page

**Custom Attack -- Target Platform and Type**  
Please select one or more target platforms for this version.

☐ mx 9.4 and above  
☐ j-series 9.5 and above  
☐ srx-branch 9.4 and above  
☐ isg-3.0.0  
☐ isg-3.1.134269  
☐ isg-3.1.135801  
☐ isg-3.4.0  
☐ isg-3.4.125129  
☐ isg-3.4.135816  
☐ isg-3.5.0  
☐ isg-3.5.134268  
☐ isg-3.5.135816  
☐ srx 9.2 and above  
☐ idp-4.0.0  
☐ idp-4.0.110090709  
☐ idp-4.0.110090831  
☐ idp-4.1.0  
☐ idp-4.1.110100209  
☐ idp-4.1.110100525  
☐ idp-4.2.0  
☐ idp-4.2.110100209  
☐ idp-4.2.110100525  
☐ idp-5.0.0  
☐ idp-5.0.110100209  
☐ idp-5.0.110100525  
☒ idp-5.1.110100525  
☐ 5.3.0-default  
☐ 5.3.0-ro-bo  
☐ 5.3.0-smb-server  
☐ 5.3.0-worm

Type

Next Cancel Help

Table 10: Attack Object Types

Type	Description
Signature	<p>Uses a stateful attack signature (a pattern that always exists within a specific section of the attack) to detect known attacks.</p> <p>Stateful signature attack objects also include the protocol or service used to perpetrate the attack and the context in which the attack occurs.</p> <p>If you know the exact attack signature, the protocol, and the attack context used for a known attack, select this option.</p>
Compound Attack	<p>Detects attacks that use multiple methods to exploit a vulnerability. This object combines multiple signatures or protocol anomalies into a single attack object, forcing traffic to match all combined signatures or anomalies within the compound attack object before traffic is identified as an attack.</p> <p>By combining and even specifying the order in which signatures or anomalies must match, you can be very specific about the events that must place before the IDP engine identifies traffic as an attack.</p> <p>If you need to detect an attack that uses several benign activities to attack your network, or if you want to enforce a specific sequence of events to occur before the attack is considered malicious, select this option.</p>

9. Select **Signature** and click **Next**.

10. On the Custom Attack – General Properties page, configure constraints and other settings. Figure 5 on page 15 shows the Custom Attack – General Properties page. Table 11 on page 15 provides guidelines for completing the settings.



Figure 5: Custom Attack - General Properties Page

**Custom Attack -- General Properties**  
Specify Signature Information

**Info**

Platform(s) idp5.1.0  
Type Signature  
False-Positives Unknown

**Service Binding**

Protocol Type Any

**Time Binding**

Enabled ☐  
Scope Peer  
Count/Min. 2

**With-in Bytes Constraint**

Lower Limit /	Upper Limit	Start Point

**With-in Packets Constraint**

Lower Limit /	Upper Limit

**Context Check**

Constraint /	Comparison Operator	Operand

Back Next Cancel Help

Table 11: Custom Attack – General Properties

Property	Description
Info	

Table 11: Custom Attack – General Properties (*continued*)

Property	Description
False Positives	<p>Select the frequency that the attack object produces a false positive on your network: <b>Unknown, Rarely, Occasionally, Frequently</b>.</p> <p>Typically, you do not initially know the frequency of false positives. You can update this setting as your observations change.</p>
<b>Service Binding</b>	
Protocol Type	<p><b>Service</b>—If you were able to determine the service through your research, select <b>Service</b>. Later in the wizard, you can specify a service context.</p> <hr/> <p><b>IP</b>—If you are not sure of the service but you know IP details, select <b>IP</b> and specify a protocol type number.</p> <hr/> <p><b>TCP, UDP, or ICMP</b>—If you do not know the service context but you know protocol details, select the protocol.</p> <p>For TCP and UDP protocol types, specify the port ranges.</p> <hr/> <p><b>RPC</b>—If you are detecting threats over remote procedure call (RPC) protocol, select this option and specify the program ID.</p> <p>RPC is used by distributed processing applications to handle interaction between processes remotely. When a client makes a remote procedure call to an RPC server, the server replies with a remote program. Each remote program uses a different program number.</p> <hr/> <p><b>IPv6 or ICMPv6</b>—Do not select these options. IDP Series devices do not support inspection of IPv6.</p> <hr/> <p><b>Any</b>—If you are unsure of the correct service, select <b>Any</b> to match the signature in all services. Matching any service essentially turns off service binding and has a significant performance impact. Specify <b>Any</b> when you know that attacks are using multiple services to attack your network.</p> <p><b>NOTE:</b> You must select a service binding other than <b>Any</b> if you want to select a context for the attack.</p>

Table 11: Custom Attack – General Properties (*continued*)

Time Binding	
Enable	Time binding attributes track how many times a signature is repeated. By configuring the scope and count of an attack, you can detect a sequence of the same attacks over a period of time (one minute) across sessions. This method is useful for detecting brute force attacks that attempt to guess authentication credentials or overwhelm system capacity to handle data.
Scope	<p>Select the scope within which the count occurs:</p> <ul style="list-style-type: none"> <li>• <b>Source</b>—Detects the signature in traffic from the source IP address for the specified number of times, regardless of the destination IP address.</li> <li>• <b>Destination</b>—Detects the signature in traffic from the destination IP address for the specified number of times, regardless of the source IP address.</li> <li>• <b>Peer</b>—Detects the signature in traffic between source and destination IP addresses of the sessions for the specified number of times.</li> </ul>
Count/Min	<p>Enter the number of times per minute that the signature must be detected within the specified scope before the device identifies the traffic as a match.</p> <p>The minute timer starts when the signature first matches the event. If the signature matches the same event for the specific count or higher within the next 60 seconds, the traffic is a match.</p> <p>The system increments the count each time it detects the signature, either regardless of port (application identification) or according to your port binding settings. For example, when the system detects the signature on TCP/80 and then on TCP/8080, the count is 2.</p>

Table 11: Custom Attack – General Properties (*continued*)

Constraints	
Within Bytes Constraint	<p>Use this constraint to require that the pattern be found within a byte range:</p> <ul style="list-style-type: none"> <li>• Lower limit—Specify the beginning of the range.</li> <li>• Upper limit—Specify the end of the range.</li> <li>• Start point—Your selection must be consistent with your pattern context setting. For example, if you configured one of the service contexts, select <b>Context</b>. If you configured one of the packet contexts, select <b>Packet</b>. If you configured one of the stream contexts, select <b>Stream</b>.</li> </ul> <p>In NSM, it is possible to select a start point that is inconsistent with the pattern context setting. For example, the NSM object editor allows you to configure a pattern context http-variable and then set a within bytes start point that is start-of-packet. However, the within bytes match logic will be resolved to the start point you should have selected: context.</p> <p>Inspection for this object terminates when the range limit is reached.</p> <p>Example: If you know a threat can be identified either completely within the first 20 bytes of the http-variable context or not identified at all, you set the context to http-variable and use the within-bytes constraint to terminate inspection after bytes 1-20 of the generated http-variable context are processed.</p> <p>You can set multiple constraints. The constraints are evaluated as a Boolean OR.</p> <p>Example: You configure two start-of-stream constraints with byte ranges of 20-40 and 80-100. The constraint rules out matches unless found within either byte range.</p>
Within Packets Constraint	<p>Use this constraint to require that the pattern be found completely within a packet range:</p> <ul style="list-style-type: none"> <li>• Lower limit—Specify the beginning of the range.</li> <li>• Upper limit—Specify the end of the range.</li> </ul> <p>Inspection (for this object) terminates when the range limit is reached.</p> <p>Example: If you know a threat can be identified either in the first 2 packets or not identified at all, you set a stream context and use the within packets constraint to terminate inspection after 2 packets.</p>
Context Check	<p>Use this constraint to require the matching context be of a specified byte length to be a hit:</p> <ul style="list-style-type: none"> <li>• Constraint—Select <b>length</b>.</li> <li>• Comparison operator—Select =, !, &gt;, or &lt;.</li> <li>• Operand—Select a byte length.</li> </ul> <p>Example: You can use the context check constraint as a tuning device to limit processing for harmless traffic. For example, if you know that a certain class of attack, like a buffer overflow attack, always has an unusually large byte length in a given context, you can use this constraint to ignore contexts of normal length. If you set the FTP username context length requirement to be &gt; 18, you would only see signature hits if the FTP username context is longer than 18 bytes.</p> <p>You can specify multiple constraints. For example, if you add a &lt; 25 constraint to the previous example, you would only see hits if the username context is between 18 and 25 bytes.</p>
Click <b>Next</b> .	
11. On the Custom Attack – Attack Pattern page, configure pattern settings. Figure 6 on page 19 shows the Custom Attack – Attack Pattern page. Table 12 on page 19 provides guidelines for completing the settings.	

Figure 6: Custom Attack – Attack Pattern Page

**Custom Attack -- Attack Pattern**  
Specify Signature Information

Detection Pattern

Negate ☐

Context

Direction

Flow

Back Next Cancel Help

Table 12: Custom Attack – Attack Pattern

Setting	Description
Pattern	A DFA expression. The following rows summarize DFA syntax conventions. For detailed information, consult a standard source on programming with regular expressions.

Table 12: Custom Attack – Attack Pattern (*continued*)

Setting	Description
\B.0.1..00\B	<p>Bit-level matching for binary protocols. The length of the bitmask must be in multiples of 8.</p> <p>The first \B denotes the start of the bitmask. The last \B denotes the end of the bitmask.</p> <p>The decimal (.) indicates the bit can be either 0 or 1.</p> <p>A 0 or 1 indicates the bit at that position must be 0, or must be 1.</p>
\0 <octal_number>	For a direct binary match.
\X<hexadecimal-number>\X	For a direct binary match.
\[<character-set>\]	For case-insensitive matches.
.	To match any symbol.
*	To match 0 or more symbols.
+	To match 1 or more symbols.
?	To match 0 or 1 symbol.
()	Grouping of expressions.
	<p>Alternation. Typically used with ().</p> <p>Example: The following expression matches dog or cat: (dog   cat).</p>
[]	<p>Character class. Any explicit value within the bracket at the position matches.</p> <p>Example: [Dd]ay matches Day and day.</p>
[<start>--<end>]	<p>Character range. Any value within the range (denoted with a hyphen). You can mix character class and a hexadecimal range.</p> <p>Example: [AaBbCcDdEeFf0-9].</p>
[^<start>--<end>]	<p>Negation of character range.</p> <p>Example: [^Dd]ay matches Hay and ray, but not Day or day.</p> <p><b>NOTE:</b> To negate an entire signature pattern, select the Negate option under the pattern text box.</p>
\u<string>\u	Unicode insensitive matches.
\s	Whitespace.

Table 12: Custom Attack – Attack Pattern (*continued*)

Setting	Description																		
\	<p>Use a backslash to escape special characters so that they are matched and not processed as regular expression operators.</p> <table> <tr> <th>Character</th><th>Escaped</th></tr> <tr> <td>*</td><td>\*</td></tr> <tr> <td>(</td><td>\(</td></tr> <tr> <td>)</td><td>\)</td></tr> <tr> <td>.</td><td>\.</td></tr> <tr> <td>+</td><td>\+</td></tr> <tr> <td>\</td><td>\\</td></tr> <tr> <td>[</td><td>\0133</td></tr> <tr> <td>]</td><td>\0135</td></tr> </table> <p><b>NOTE:</b> Because the combination of the backslash and the open and close square brackets are used in the case-insensitive expression, you must use the backslash with the octal code for the bracket characters.</p>	Character	Escaped	*	\*	(	\(	)	\)	.	\.	+	\+	\	\\	[	\0133	]	\0135
Character	Escaped																		
*	\*																		
(	\(																		
)	\)																		
.	\.																		
+	\+																		
\	\\																		
[	\0133																		
]	\0135																		
Negate	Negates the attack pattern.																		

Table 12: Custom Attack – Attack Pattern (*continued*)

Setting	Description
Context	<p>Binds pattern matching to a context.</p> <p>For known services, such as HTTP, select the service in the first box, and select the HTTP context you discovered with <b>scio ccap</b>, such as HTTP POST Parsed Param, in the second box.</p> <p>If you were unable to discover the context, select <b>Other</b> in the first box, and select one of the following contexts in the second box:</p> <ul style="list-style-type: none"> <li>• <b>Packet</b>—Detects the pattern in any packet.</li> <li>• <b>First Packet</b>—Inspects only the first packet of a stream. When the flow direction is set to any, the detector engine checks the first packet of both the server-to-client (STC) and client-to-server (CTS) flows. Less processing means greater performance. If you know that the pattern appears in the first packet of a session, select <b>First Packet</b>.</li> <li>• <b>First Data Packet</b>—Inspection ends after the first packet of a stream. Select this option to detect the attack in only the first data packet of a stream. If you know that the pattern appears in the first data packet of a stream, select <b>First Data Packet</b>.</li> <li>• <b>Stream 256</b>—Reassembles packets and searches for a pattern match within the first 256 bytes of a traffic stream. Stream 256 is often the best choice for non-UDP attacks. When the flow direction is set to <b>any</b>, the detector engine checks the first 256 bytes of both the STC and CTS flows. If you know that the pattern will appear in the first 256 bytes of a session, select <b>Stream 256</b>.</li> <li>• <b>Stream 8K</b>—Like Stream 256 except reassembles packets and searches for a pattern match within the first 8192 bytes of a traffic stream.</li> <li>• <b>Stream 1K</b>—Like Stream 256 except reassembles packets and searches for a pattern match within the first 1024 bytes of a traffic stream.</li> <li>• <b>Line</b>—Detects a pattern within a specific line. Use this context for line-oriented applications or protocols (such as FTP).</li> <li>• <b>Stream</b>—Reassembles packets and extracts the data to search for a pattern match. However, the IDP engine does not recognize packet boundaries for stream contexts, so data for multiple packets is combined. Select this option only when no other context option contains the attack.</li> </ul> <p><b>NOTE:</b> If you select a line, stream, or service context, you do not configure match criteria for IP settings and protocol header fields.</p>
Direction	<p>Select the direction in which to detect the pattern:</p> <ul style="list-style-type: none"> <li>• <b>Client to Server</b>—Detects the pattern only in client-to-server traffic.</li> <li>• <b>Server to Client</b>—Detects the pattern only in server-to-client traffic.</li> <li>• <b>Any</b>—Detects the pattern in either direction.</li> </ul> <p>The session initiator is considered the client, even if that source IP is a server.</p>
Flow	<p>Select the flow in which to detect the attack:</p> <ul style="list-style-type: none"> <li>• <b>Control</b>—Detects the pattern in the initial connection that is established to issue commands, requests, and so on. Ninety-nine percent of signatures use control.</li> <li>• <b>Auxiliary</b>—Detects the pattern in the response connection that is established intermittently to transfer requested data. This option supports a small number of protocols, such as PTP.</li> <li>• <b>Both</b>—Detects the pattern in the initial and response connections.</li> </ul> <p><b>TIP:</b> Using a single flow (instead of Both) improves performance and increases detection accuracy.</p>



Click **Next** to display the Custom Attack – IP Settings and Header Matches page.

Figure 7 on page 23 shows the Custom Attack – IP Settings and Header Matches page.

Table 13 on page 24 provides guidelines for completing the settings.

**Figure 7: Custom Attack – IP Settings and Header Matches Page**

**Custom Attack -- IP Settings and Header Matches**  
Specify Signature Information

IP Protocols

IP Version  
☒ IPv4 ☐ IPv6

Type-of-service

Packet length

Id

Time-to-live

Protocol

Source

Destination

RB

MF

DF

Back Finish Cancel Help

12. If you have selected a line, stream, stream 256, or service context, do not configure match criteria for IP settings and protocol header fields. Click **Finish**.

If you are using a packet context, you can refine matching by adding criteria for IP flags and packet headers, as described in the following tables.



**TIP:** If you are unsure of the IP flags and IP fields you want to match, leave all fields blank. If no values are set, the IDP engine attempts to match the signature for all header contents.

**Table 13: Custom Attack – IP Settings and Header Matches Page**

Setting	Description
IP Version	Select <b>IPv4</b> . IDP Series devices do not support inspection of IPv6.
Type of Service	Service type. Common service types are: <ul style="list-style-type: none"> <li>• 0000 Default</li> <li>• 0001 Minimize Cost</li> <li>• 0002 Maximize Reliability</li> <li>• 0003 Maximize Throughput</li> <li>• 0004 Minimize Delay</li> <li>• 0005 Maximize Security</li> </ul>
Packet Length	Number of bytes in the packet, including all header fields and the data payload.
ID	Unique value used by the destination system to reassemble a fragmented packet.
Time-to-live	Time-to-live (TTL) value of the packet. This value represents the number of routers the packet can pass through. Each router that processes the packet decrements the TTL by 1; when the TTL reaches 0, the packet is discarded.
Protocol	Protocol used in the attack.
Source	IP address of the attacking device.
Destination	P address of the attack target.
RB	Reserved bit. This bit is not used.
MF	More fragments. When set (1), this option indicates that the packet contains more fragments. When unset (0), it indicates that no more fragments remain.
DF	Don't fragment. When set (1), this option indicates that the packet cannot be fragmented for transmission.

Figure 8 on page 25 shows the Custom Attack – IP Settings and Header Matches page. Table 14 on page 25 provides guidelines for completing the settings.

Figure 8: Custom Attack Object: TCP Packet Header Fields

**Custom Attack -- IP Settings and Header Matches**  
Specify Signature Information

IP Protocols

TCP/UDP/ICMP Header Matches TCP Packet Header Fields

Source Port	none	none
Dest Port	none	none
Seq. Number	none	none
Ack. Number	none	none
Header Length	none	none
Data Length	none	none
Window Size	none	none
UrgPtr	none	none
Urgent bit	none	
ACK bit	none	
PSH bit	none	
RST bit	none	
SYN bit	none	
FIN bit	none	
R1 bit	none	
R2 bit	none	

Back Finish Cancel Help

Table 14: Custom Attack Object: TCP Packet Header Fields

Setting	Description
Source Port	Port number on the attacking device.
Destination Port	Port number of the attack target.
Sequence Number	Sequence number of the packet. This number identifies the location of the data in relation to the entire data sequence.

Table 14: Custom Attack Object: TCP Packet Header Fields (*continued*)

Setting	Description
ACK Number	ACK number of the packet. This number identifies the next sequence number; the ACK flag must be set to activate this field.
Header Length	Number of bytes in the TCP header.
Window Size	Number of bytes in the TCP window size.
Data Length	Number of bytes in the data payload. For SYN, ACK, and FIN packets, this field should be empty.
Urgent Pointer	Data in the packet is urgent; the URG flag must be set to activate this field.
URG Bit	When set, the urgent flag indicates that the packet data is urgent.
ACK Bit	Acknowledgment flag. When set, acknowledges receipt of a packet.
PSH Bit	Push flag. When set, indicates that the receiver should push all data in the current sequence to the destination application (identified by the port number) without waiting for the remaining packets in the sequence.
RST Bit	Reset flag. When set, resets the TCP connection, discarding all packets in an existing sequence.
FIN Bit	Final flag. When set, indicates that the packet transfer is complete and the connection can be closed.
R1 Bit, R2 Bit	Reserved bit. Unused.

Figure 9 on page 27 shows the Custom Attack – IP Settings and Header Matches page. Table 15 on page 27 provides guidelines for completing the settings.

Figure 9: Custom Attack Object: UDP Packet Header Fields

**Custom Attack -- IP Settings and Header Matches**  
Specify Signature Information

IP Protocols

TCP/UDP/ICMP Header Matches UDP Packet Header Fields

Source Port none

Dest. Port none

Data Length none

Back Finish Cancel Help

Table 15: Custom Attack Object: UDP Header Fields

Setting	Description
Source Port	Port number on the attacking device.
Destination Port	Port number of the attack target.
Data Length	Number of bytes in the data payload.

Figure 10 on page 28 shows the Custom Attack -- IP Settings and Header Matches page. Table 16 on page 28 provides guidelines for completing the settings.

Figure 10: Custom Attack Object: ICMP Packet Header Fields

Table 16: Custom Attack Object: ICMP Packet Header Fields

Setting	Description
<b>ICMP</b>	
ICMP Type	Primary code that identifies the function of the request or reply.

Table 16: Custom Attack Object: ICMP Packet Header Fields (*continued*)

Setting	Description
ICMP Code	Secondary code that identifies the function of the request or reply within a given type.
Sequence Number	Sequence number of the packet. This number identifies the location of the request/reply in relation to the entire sequence.
ICMP ID	Identification number, which is a unique value used by the destination system to associate requests and replies.
Data length	Number of bytes in the data payload.



**NOTE:** ICMPv6 header fields are not applicable. IDP Series devices do not support inspection of IPv6.

13. Click **Finish**.

#### Related Documentation

The following related topics are included in the *IDP Series Custom Attack Object Reference and Examples Guide*:

- Creating a Compound Attack Object on page 29
- Creating Custom Attack Objects Overview on page 1
- Reference: Custom Attack Object Protocol Numbers on page 61
- Reference: Custom Attack Object Service Properties on page 67
- Reference: Custom Attack Object Service Contexts on page 70
- Example: Custom Attack Object DFA Expressions on page 35

The following related topic is included in the *IDP Series Administration Guide*:

- Attack Objects Task Summary

The following related topic is included in the *IDP Series Concepts and Examples Guide*:

- Using Attack Objects

## Creating a Compound Attack Object

Use compound attack objects in cases where:

- Attacks use multiple methods to exploit a vulnerability and, inspected independently, the individual contexts appear benign.
- Matching multiple contexts reduces false positives.
- Coupling a signature with a protocol anomaly reduces false positives.

You select signature attack objects or predefined anomalies as “members” of the compound object, and you use Boolean expressions to specify matching logic.

To configure a compound attack object:

1. Configure general attack object properties and reference information as described for signature attack objects.

On the Target Platform and Type page, select a target platform, select **Compound Attack**, and click **Next**.

2. On the Custom Attack – General Properties page, configure the settings described in Table 17 on page 30.

**Table 17: Custom Attack – General Properties**

Property	Description
False Positives	Same guidelines as for signature attack objects.
Service Binding	
Time Binding	

---

Click **Next**.

3. On the Compound Members page, specify compound attack parameters and add members. Figure 11 on page 31 shows the Custom Attack – Compound Members page. Table 18 on page 31 provides guidelines for completing the settings.



Figure 11: Custom Attack – Compound Members

**Custom Attack -- Compound Members**  
Specify Signature Information

Compound Attack Parameters

Scope: Session Reset

Boolean Expression:

Member Name	Member Type
Compound Attack Member	

Context-Check Constraint

Constraint /	Comparis...	Operand

Match within same context: (empty)

With-in Bytes Constraint

Lower Limit /	Upper Limit	Member N...

With-in Packet Constraint

Lower Limit /	Upper Limit	Member N...

Back Finish Cancel Help

Table 18: Compound Attack Parameters

Setting	Description
Scope	<p>Select one of the following:</p> <ul style="list-style-type: none"> <li><b>Session</b>—Allows multiple matches for the object within the same session.</li> <li><b>Transaction</b>—Matches the object across multiple transactions that occur within the same session.</li> </ul>
Reset	Enable to detect multiple occurrences of the attack object in the same session. Disable to log multiple occurrences as one.

Table 18: Compound Attack Parameters (*continued*)

Setting	Description
Boolean Expression	<p>Type a Boolean expression using the following Boolean operators:</p> <ul style="list-style-type: none"> <li>• OR—If either of the member name patterns match, the expression matches.</li> <li>• AND—If both of the member name patterns match, the expression matches. It does not matter which order the members appear in.</li> <li>• OAND—If both member name patterns match, and if they appear in the same order as in the Boolean expression, the expression matches.</li> </ul> <p>For example, the Boolean expression ((s1 OAND s2) OR (s1 OAND s3)) AND (s4 AND s5) would match an attack that contains s1 followed by either s2 or s3, and that also contains s4 and s5 in any location.</p>
Add member	<p>Click the + icon, select <b>Signature</b> or <b>Protocol Anomaly</b>, and complete the configuration details.</p> <p>For signature members, specify the same contextual information as you do for a signature attack object.</p> <p>For protocol anomaly members, select from a list of predefined protocol anomalies.</p> <p><b>BEST PRACTICE:</b> Our signature team uses the following naming convention for members: m01, m02, m03, and so on. We recommend you use this same naming convention.</p>
Context Check	<p>Use this constraint to require the matching context be of a specified byte length to be a hit:</p> <ul style="list-style-type: none"> <li>• Constraint—Select <b>length</b>.</li> <li>• Comparison operator—Select =, !, &gt;, or &lt;.</li> <li>• Operand—Select a byte length.</li> </ul> <p>Example: You can use the context check constraint as a tuning device to limit processing for harmless traffic. For example, if you know that a certain class of attack, like a buffer overflow attack, always has an unusually large byte length in a given context, you can use this constraint to ignore contexts of normal length. If you set the FTP username context length requirement to be &gt; 18, you see signature hits only when the FTP username context is longer than 18 bytes.</p> <p>You can specify multiple constraints. For example, if you add a &lt; 25 constraint to the previous example, you see hits only when the username context is between 18 and 25 bytes.</p>
Match within same context	<p>Use this constraint to require selected signature members to be found in the same context instance (in any order). You can select up to 32 signature members.</p> <p>Protocol anomaly members are not selectable and are not a component of this constraint.</p> <p>Example: You design a compound attack with service context ftp-filename, and you enable this restraint. The pattern for member 1 is <b>test</b>; the pattern for member 2 is <b>hello</b>. A user opens an FTP session and requests files test.txt and hello.txt. Each file transfer occurs in its own context, not within the same context instance, so the FTP session does not trigger this attack object. Instead, consider what happens when the user requests a file named test-hello.txt. In this case, both members are found in a single context instance, so the FTP session is a match.</p>
Within Bytes Constraint	<p><b>NOTE:</b> IDP OS Release 5.1 does not support the within bytes constraint for compound attack objects.</p>

Table 18: Compound Attack Parameters (*continued*)

Setting	Description
Within Packets Constraint	<p>Use this constraint to require that the pattern be found completely within a packet range of a stream:</p> <ul style="list-style-type: none"> <li>Lower limit—Specify the beginning of the range.</li> <li>Upper limit—Specify the end of the range.</li> <li>Member—Select one or two members. You cannot configure a relationship for more than two members.</li> </ul> <p>If you set a packet constraint for one member, the program logic counts packets beginning with the start-of-stream. The member must be found completely within the packet range indicated.</p> <p>If you select two members and apply a packet constraint to them, the program logic counts the first match as packet 0. If you specify a range of 1-2 with member 1 and member 2, the second pattern must occur within one or two packets from the packet containing the first match. Specifying 0-1 requires the pattern to appear in the same packet or within one packet from the first match. Order does not matter unless you use an Boolean ordered AND to specify the order in which the patterns must appear.</p> <p>Inspection for this object terminates when the range limit has been reached.</p>

4. Click **Finish**.

**Related Documentation** The following related topics are included in the *IDP Series Custom Attack Object Reference and Examples Guide*:

- Creating a Signature Attack Object on page 9
- Creating Custom Attack Objects Overview on page 1

The following related topic is included in the *IDP Series Administration Guide*:

- Attack Objects Task Summary

The following related topic is included in the *IDP Series Concepts and Examples Guide*:

- Using Attack Objects

## Testing a Custom Attack Object

We recommend the following workflow to test a custom attack object. Note that the following procedure consists of general steps and is intended for expert users who are familiar with these tasks.

To test a custom attack object:

1. Create a new security policy and new IDP rulebase rule that includes only the custom attack object to be tested. Enable logging and packet logging.
2. Push the policy to the IDP Series lab device.

3. From the attacker computer, reproduce the attack that targets the victim computer.
4. Use the NSM Log Viewer to see whether the traffic generated logs as expected.

If your test fails, review the attack advisory, the protocol RFC, and the attack code or packet captures to identify additional information that can help you fine-tune your settings. The most frequent issue that requires tuning is the syntax of the DFA expression.

**Related  
Documentation**

The following related topic is included in the *IDP Series Custom Attack Object Reference and Examples Guide*:

- Reproducing an Attack on page 6

The following related topics are included in the *IDP Series Administration Guide*:

- Modifying IDP Rulebase Rules (NSM Procedure)
- Pushing Security Policy Updates to an IDP Series Device (NSM Procedure)
- IDP Series Logs and Reports in NSM Task Summary
- Attack Objects Task Summary

CHAPTER 2

# Examples: Custom Attack Object Pattern Syntax

The following examples in this chapter illustrate syntax for matching an attack pattern:

- Example: Custom Attack Object DFA Expressions on page 35
- Example: Using Pattern Negation on page 37
- Example: Matching File Extensions on page 37
- Modifying Custom Attack Objects Due to Changes Introduced in Signature Update #1972 on page 38

## Example: Custom Attack Object DFA Expressions

Table 19 on page 35 provides examples of syntax for matching an attack pattern.

Table 19: Example: Custom Attack Object Regular Expressions

Example Syntax	Description	Example Matches
Hello.\B.0.1..00\B...world	<p>There are two aspects to matching:</p> <p>Must match the bitmask pattern: \B.0.0.1..00\B</p> <p>Must match the number of bytes (signified by .) before and after the bitmask pattern.</p>	<p>Matches:</p> <p>Hello.\B.0.11100\B...world Hello.\B.0.10000\B...world</p> <p>Does not match:</p> <p>Hello.\B.0.1..00\B.world Hello.\B.0.1..1\B...world</p>
\X01 86 A5 00 00\X	Pattern with the five specified bytes verbatim.	01 86 A5 00 00
(hello world)	Pattern with hello or world occurring once.	hello world
(hello world)+	Pattern with hello or world occurring one or more times.	helloworld worldhello hellohello

Table 19: Example: Custom Attack Object Regular Expressions (*continued*)

Example Syntax	Description	Example Matches
\[hello\]	Pattern hello, case insensitive.	hElLo HELLO heLLO
\uHello\u	Pattern hello, Unicode insensitive.	hello 68656c6c6f
hello\sworld	Pattern hello world, the two words separated by a whitespace.	hello world
[c-e]a(d t)	Pattern with the first letter of c, d, or e; the middle letter a; and ending in d or t.	cat dad eat
[^c-d]a(d t)	Pattern that begins a letter other than c, d, or e; have the second letter a; and end in d or t.	fad zad
a*b+c	Pattern with any number of a characters (including zero); followed by one or more b characters; followed by a c character.	bc abc aaaabbbbc
T[Kk]	Pattern that begins with an uppercase T, followed by a case-insensitive k.	TK Tk
([Tt])k	Pattern that begins with a case-insensitive t, followed by a lowercase k.	Tk Tk
Sea[lm]	Pattern that begins with Sea, followed by a lowercase l, m, or n.	Seal Seam Sean
([B-D])at	Pattern that begins with an uppercase B, C, or D, followed by a lowercase at.	Bat Cat Dat

Table 19: Example: Custom Attack Object Regular Expressions (*continued*)

Example Syntax	Description	Example Matches
\0133\[hello\]\0135	Pattern that begins with an opening bracket, followed by case-insensitive hello, ending with a closing bracket. This expression uses the \0 expression to signify that the following expression is an octal code, then the octal code for the opening bracket (133) or the closing bracket (135) follows.	[hello] [HeLLo]

**Related Documentation**

- Reference: Nonprintable and Printable ASCII Characters on page 106

### Example: Using Pattern Negation

You can use pattern negation to exclude a pattern known to be safe and to match all else.

For example, suppose you are designing an attack object to inspect traffic to an FTP server. You know that account username and passwords are well maintained to ensure that only authorized users can access internal resources. However, as networks grow and new components are added, user accounts can proliferate, thereby increasing network access to specific components. In this example, you have an FTP server on your internal network that has multiple user accounts enabled. To improve security, you want to restrict access to the FTP administrator.

You create an attack object for the FTP service, ftp-username context, and pattern **admin**; and you select the **Negate** check box. The result is an attack object that can flag login attempts by users other than **admin**. You can use this attack object in a rule that logs or drops matching traffic.

**Related Documentation**

- Creating a Signature Attack Object on page 9
- Creating a Compound Attack Object on page 29

### Example: Matching File Extensions

In this example, you want to detect Microsoft Windows metafiles, which use the extensions .emf (Windows Enhanced Metafiles) and .wmf (Microsoft Windows Metafile).

To match either of these file types, use a simple DFA expression:

```
.*\.[w|emf\]
```

In this expression:

- The period combined with the asterisk (.\* ) indicates that one or more characters must appear (wildcard match).
- The backslash combined with the period character (\. ) indicates that the period character is escaped (the period appears in the pattern).
- The parentheses at the beginning and end of the expression ( ) indicate a group. The pipe character between the e and the w (e|w) indicates an OR relationship between the characters. For this expression, e or w must appear in the pattern to match this expression; only one must be present.
- The opening bracket ([ ) indicates the beginning of a case-insensitive match for all characters until the closing bracket (] ) appears.
- The closing bracket (] ) indicates the ending of a case-insensitive match.

#### Related Documentation

- [Creating a Signature Attack Object on page 9](#)
- [Creating a Compound Attack Object on page 29](#)

## Modifying Custom Attack Objects Due to Changes Introduced in Signature Update #1972

This topic describes changes to some service contexts generated by the HTTP protocol decoder. Beginning with [Signature Update #1972](#), the HTTP protocol decoder no longer generates some contexts. If your IDP security policy includes custom signatures that use the contexts that have been removed, you must modify your attack object definitions as described below to avoid policy compilation errors. This topic includes the following information:

- [Reference: Removed Contexts on page 38](#)
- [Example: Replacing the Context for Patterns Appearing in HTML Text on page 39](#)
- [Example: Replacing the Contexts for Patterns Appearing in URLs on page 39](#)

### Reference: Removed Contexts

To improve performance, the HTTP protocol decoder no longer generates the contexts listed in the first column of Table 20 on page 38. Review this table for guidelines on replacing the contexts in custom attack objects.

**Table 20: HTTP Service Contexts**

Removed	Replace With	Guideline
http-text-html-body	http-text-html	Change signatures that use context http-text-html-body to http-text-html. You do not need to make changes to the signature pattern or other properties.



Table 20: HTTP Service Contexts (*continued*)

Removed	Replace With	Guideline
<ul style="list-style-type: none"> <li>http-get-url-parsed-param</li> <li>http-post-url-parsed-param</li> <li>http-head-url-parsed-param</li> <li>http-get-url-parsed-param-parsed</li> <li>http-post-url-parsed-param-parsed</li> <li>http-head-url-parsed-param-parsed</li> </ul>	Use a combination of the following contexts: <ul style="list-style-type: none"> <li>http-request-method</li> <li>http-url-parsed</li> <li>http-variable-parsed</li> </ul>	<p>Use a compound signature with a Boolean AND to break the signature pattern into multiple pieces. Ensure the Scope field is set to Transaction.</p> <p>Using the http-request-method context is optional. You use the http-request-method context to bind detection to http GET or POST or HEAD transactions. For GET method, we use the pattern <code>\[GET\]</code> (case insensitive GET). Use http-request-method only if the results you logged previously matching on Request Method are worth preserving. If not, omit it to improve performance. If you use http-request-method, order it first in the compound chain.</p> <p>Use the http-url-parsed context to match an attack signature identifiable in the URL. Use this context to match a pattern in the URL that appears before variable parameters—the part of the URL before the question mark (?).</p> <p>Use one or more http-variable-parsed contexts to match the URL variable parameters—the part of the URL after the question mark (?), normally separated by ampersands (&amp;).</p>

#### Example: Replacing the Context for Patterns Appearing in HTML Text

Each context generated by the HTTP detector engine has a performance cost. Contexts http-text-html and http-text-html-body serve the same purpose. Reducing the number of contexts improves performance.

Table 21 on page 39 shows the properties of a signature before [Update #1972](#) and the signature after. This is a simple change. You change only the context. You do not need to change the pattern or other properties.

Table 21: HTTP Service Contexts: HTML Text

	Before Update	After Update
Context	http-text-html-body	http-text-html
Pattern	<code>.*&lt;span&gt;&lt;/span&gt;.*</code>	<code>.*&lt;span&gt;&lt;/span&gt;.*</code>

#### Example: Replacing the Contexts for Patterns Appearing in URLs

This section has two parts:

- Signatures that Match Request Methods on page 39
- Signatures that Match URL Strings and URL Variables on page 40

##### ***Signatures that Match Request Methods***

When modifying custom attack objects that previously matched request methods GET, POST, or HEAD, consider whether matches against these request method patterns were effective for you. Keep in mind, each context generated has a performance cost. If request

method is not essential to your results, take this opportunity to recast your signature without it.

Table 22 on page 40 and Table 23 on page 40 show the properties of a signature before [Update #1972](#) and the compound signature after. This example preserves an interest in request method.

**Table 22: HTTP Service Contexts: Request Methods Before Update**

	Signature Before Update
Scope	–
Context	http-get-url-parsed-param
Pattern	\[/viper/vegaspalms/\].*

**Table 23: HTTP Service Contexts: Request Methods After Update**

	Compound Signature After Update	
	m01	m02
Scope	Transaction	
Context	http-request-method	http-url-parsed
Pattern	\[GET\]	\[/viper/vegaspalms/\].*

***Signatures that Match URL Strings and URL Variables***

In general, breaking a single pattern into multiple contexts could positively or negatively impact performance. You need to test your changes to understand performance impact before deploying the attack objects in a production network. The example shown in Table 24 on page 40 and Table 25 on page 40 breaks URL matching into multiple contexts. Our security team has tested performance for the recommendations described here.

**Table 24: HTTP Service Contexts: URL Strings and Variables Before Update**

	Signature Before Update
Scope	–
Context	http-get-url-param-parsed-param
Pattern	\[/cvs/index[0-9]?\.php\?option=com_content&do_pdf=1&id=1\]

**Table 25: HTTP Service Contexts: URL Strings and Variables After Update**

	Compound Signature After Update			
	m01	m02	m03	m04

Table 25: HTTP Service Contexts: URL Strings and Variables After Update (*continued*)

	Compound Signature After Update			
Scope	Transaction			
Context	http-url-parsed	http-variable-parsed	http-variable-parsed	http-variable-parsed
Pattern	\[/cvs/index[0-9]?\.php\]	\[option=com_content\]	\[do_pdf=1\]	\[id=1\]

- Related Documentation**
- Reference: Custom Attack Object Service Contexts on page 70
  - Reproducing an Attack on page 6
  - Discovering the Attack Signature on page 8
  - Creating a Compound Attack Object on page 29
  - Testing a Custom Attack Object on page 33



## CHAPTER 3

# Examples: Using Matching Constraints

The following examples in this chapter show how to use matching constraints:

- Example: Using Time Binding Parameters to Detect a Brute Force Attack on page 43
- Example: Using the Within Bytes or Within Packets Constraint with Signature Attack Objects on page 45
- Example: Using the Within Packets Constraint with Compound Attack Objects on page 45
- Example: Using the Same Context Constraint with Compound Attack Objects on page 46
- Example: Using Context Length Checking to Optimize Performance on page 46

### Example: Using Time Binding Parameters to Detect a Brute Force Attack

---

The time binding constraint requires the pattern to occur a certain number of times within a minute in order for the traffic to be considered a match.

You can use the time binding parameter along with the signature to detect signs of a brute force attack. A user changing her password is a harmless event, and is normally seen occasionally on the network. However, thousands of password changes in a minute is suspicious.

In a brute force attack, the attacker attempts to break through system defenses using sheer force, typically by overwhelming the destination server capacity or by repeated, trial-and-error attempts to match authentication credentials. In a brute force login attack, the attackers first gather a list of usernames and a password dictionary. Next, the attacker uses a tool that enters the first password in dictionary for the first user in the list, then tries every password for every user until it gets a match. If the attacker tries every combination of usernames and passwords, they always succeed. However, brute force attacks often fail because the password dictionary is typically limited (does not contain all possible passwords) and the attack tool does not perform permutations on the password (such as reversing letters or changing case).

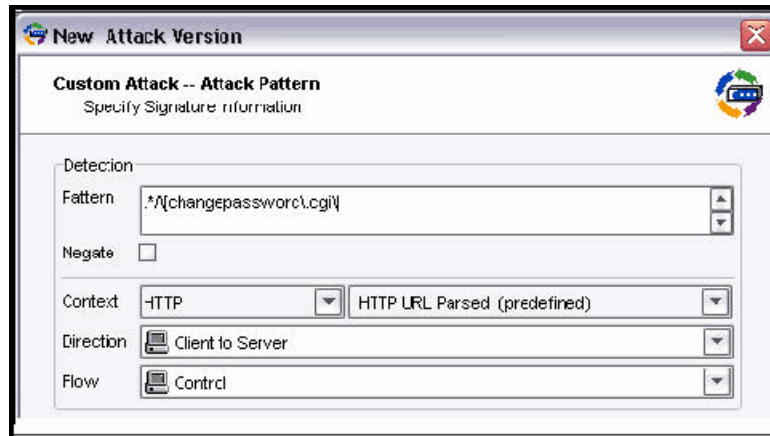
In this example, you create a signature attack object that detects an excessive number of password changes for users authenticated via HTTP (a Web-based application).

First, you configure an attack pattern:

`.*\[changepassword\.cgi\]`

Figure 12 on page 44 shows the Custom Attack – Attack Pattern page for this example.

Figure 12: Brute Force Attack Pattern

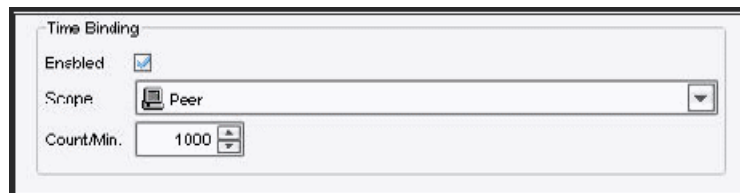


In this expression:

- The dot star combination (.\* ) indicates a wildcard match.
- The backslash before a character indicates that the character represents a regular expression and must be escaped. In this case, the character is an opening bracket. The backslash is also used in this expression before the file extension marker (the dot) and before the closing bracket.
- The name of the cgi script that is used to change user passwords is included, as well as the cgi extension.
- For context, select **HTTP-URL-PARSED** from the list because you are attempting to detect password changes that occur for Web-based applications. The `changepassword.cgi` script, when used, appears as part of the URL, but you need to tell the IDP Series device to parse the URL in order to find the name.

Next, you configure time binding. Figure 13 on page 44 shows the time binding settings for this example.

Figure 13: Brute Force Time Binding



In these settings:

- Scope is set to **Peer** so the attack pattern can match the event regardless of source or destination.

- Count is set to high number (to 1000) to avoid false positives. This value means that the changepassword.cgi script must appear in a URL 1000 times before the attack object is matched.

**Related  
Documentation**

- Reproducing an Attack on page 6
- Discovering the Attack Signature on page 8
- Creating a Signature Attack Object on page 9
- Creating a Compound Attack Object on page 29
- Testing a Custom Attack Object on page 33

## Example: Using the Within Bytes or Within Packets Constraint with Signature Attack Objects

---

With signature attack objects, you can use within bytes or within packets constraints to optimize inspection processing when you know the complete attack pattern occurs either within a specified range of bytes or packets or not at all. Inspection for this object terminates when the range limit is reached.

For example, if you know a pattern is a threat only if it occurs within the first 20 bytes of the http-variable context, you set the context to http-variable, and use the within bytes constraint to inspect bytes 1-20 of the generated http-variable context.

You can set multiple constraints. The constraints are evaluated as a Boolean OR. For example, suppose you configure two start-of-stream constraints with byte ranges of 20-40 and 80-100. The signature matches only if the pattern is found within bytes 20-40 or within bytes 80-100 from the start of the stream.

**Related  
Documentation**

- Creating a Signature Attack Object on page 9

## Example: Using the Within Packets Constraint with Compound Attack Objects

---

With compound attack objects, you can use the within packets constraint to add logic that reduces false positives.

You can select members one at a time and set a lower and upper limit for each one. The packet range for each member is from start-of-stream. The complete pattern for the member must be found within the range indicated.

For the within packets constraint, when you select two members and apply a packet constraint to them, the packet containing the first match is counted as packet 0. If your research shows that, in a particular attack, two patterns always appear within 1 or 2 packets you can select member 1 and member 2 and specify a range of 1-2. This specifies that the second pattern must occur within one or two packets from the first pattern.

If you know that the traffic pattern is an attack only when member 1 occurs before member 2, you can use a Boolean ordered AND to specify the order in which the patterns must appear. In this example, if you use a Boolean ordered AND and specify a range of 1-2, the

traffic matches only if the member 2 pattern occurs one or two packets after the packet in which the member 1 pattern is found.

**Related Documentation** • [Creating a Compound Attack Object on page 29](#)

---

## Example: Using the Same Context Constraint with Compound Attack Objects

With compound attack objects, you can use the same context constraint to require selected signature members to be found in the same context instance (in any order). You can specify up to 32 signature members.

Protocol anomaly members are not selectable and are not a component of this constraint.

Suppose you design a compound attack with service context ftp-filename, and you enable this restraint. The pattern for member 1 is **test**; the pattern for member 2 is **hello**. A user opens an FTP session and requests files test.txt and hello.txt. Each file transfer is occurs in its own context—not within the same context instance—so the FTP session does not trigger this attack object. Instead, consider what happens when the user requests a file named test-hello.txt. In this case, both members are found in a single context instance, so the FTP session is a match.

**Related Documentation** • [Creating a Compound Attack Object on page 29](#)

---

## Example: Using Context Length Checking to Optimize Performance

With signature or compound attack objects, you can use the context check constraint as a tuning device to skip processing for harmless traffic. For example, if you know that a certain class of attack, like a buffer overflow attack, always has an unusually large byte length in a given context, you can use this constraint to ignore contexts of normal length. If you set the FTP username context length requirement to be > 18, you only see signature hits when the FTP username context is longer than 18 bytes.

You can specify multiple constraints. For example, if you add a < 25 constraint to the previous example, you see hits only when the username context is between 18 and 25 bytes.

**Related Documentation** • [Creating a Compound Attack Object on page 29](#)



## CHAPTER 4

# Case Studies: Creating Custom Attack Objects to Respond to Vulnerability Reports

The following examples in this chapter illustrate how you can use custom attack objects to respond to new vulnerabilities:

- Example: Apache Tomcat Denial-of-Service Attacks on page 47
- Example: UNIX CDE/dtlogin Vulnerability on page 49
- Example: Detecting a Worm on page 53
- Example: Compound Signature to Detect Exploitation of an HTTP Vulnerability on page 55

### Example: Apache Tomcat Denial-of-Service Attacks

---

In this example, we assume you have a Web Server running Apache Tomcat. Your security administrator notifies you that a vulnerability has just been announced for Apache Tomcat, and you decide to create a custom attack object to protect your network until you can schedule downtime to patch the server.

The CVE advisory for the vulnerability (<http://nvd.nist.gov/nvd.cfm?cvename=CAN-2002-0682>) contains the following quotation:

A cross-site scripting vulnerability in Apache Tomcat 4.0.3 allows remote attackers to execute script as other web users via script in a URL with the `/servlet/` mapping, which does not filter the script when an exception is thrown by the servlet.

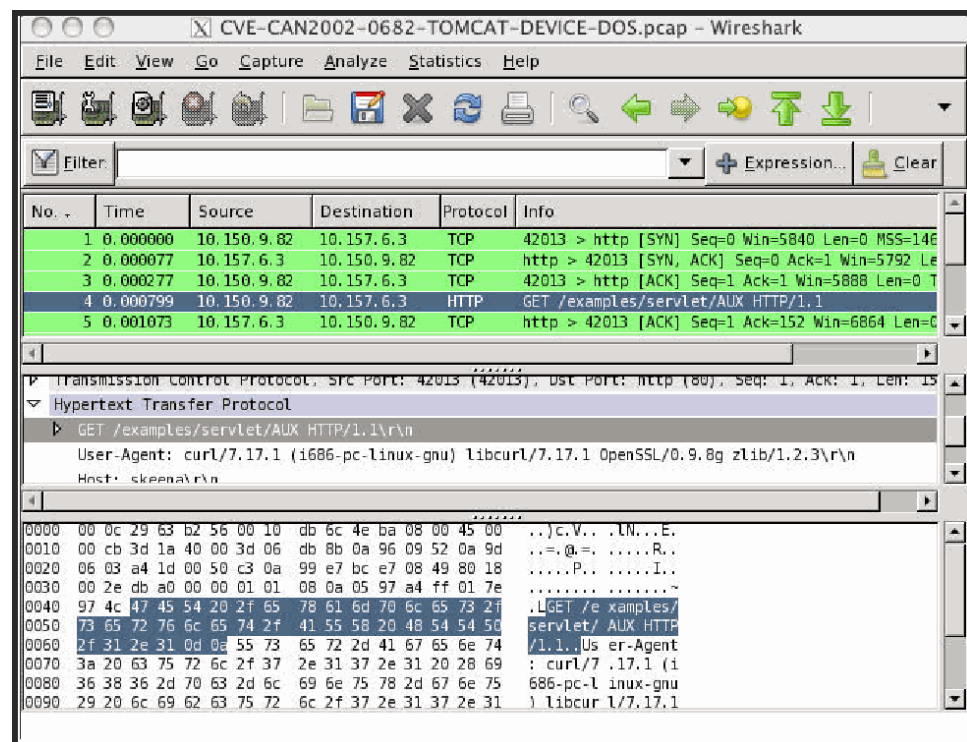
From this information, you know that the attack uses HTTP. Now you must locate the attack code. The advisory also includes references that link to more information about the attack. Unfortunately, none of the referenced Web pages contain exploit code. After searching the Web using the information you learned from the CVE advisory, you locate some exploit code at <http://packetstormsecurity.nl/0210-exploits/neuter.c>. Copy the script and move it to the attacker computer in your test lab.

To develop this attack object:

1. Reproduce the attack to determine the attack context, direction, and pattern. Ideally, use **scio ccap** and Wireshark concurrently so you have to run the attack only once.

Figure 14 on page 48 shows the packet capture in Wireshark.

Figure 14: CAN2002 Packet Capture



2. Discover the following elements of the attack signature:
  - Service. You know from the CVE advisory that the attack uses the HTTP protocol. Review the packet capture to confirm the protocol.
  - Context. Use **scio ccap** to determine whether you can match a particular service context. In this example, the signature pattern occurs in the service context HTTP URL Parsed.
  - Pattern. You know from the advisory that the attack occurs using an exploited GET method in the HTTP protocol. Select the frame that contains the GET method to view details for that section of the packet. You can quickly identify the signature pattern as **examples/servlet/AUX**.
  - Direction. Locate the source IP that initiated the session. Because this attack uses TCP, you can use the Follow TCP Stream option in Wireshark to quickly discover the source IP that initiated the session. The attack direction is client-to-server.
3. Create an attack object to match the attack signature. This example uses the following regular expression to match the signature:

```
.*examples/servlet/AUX|LPT1|CON|PRN.*
```

Figure 15 on page 49 shows the Custom Attack – Attack Pattern page for this example.

Figure 15: CAN-2002-0682 Attack Pattern

**New Attack Version**

**Custom Attack -- Attack Pattern**  
Specify Signature Information

Detection

Pattern:

Nogoto: ☐

Context: HTTP HTTP URL Parsed (predefined)

Direction: Client to Server

Flow: Control

In this expression:

- The dot star combination (.\* ) indicates a wildcard match.
- The /examples/servlet/ section is taken directly from the packet capture.
- The parentheses ( ) indicate a group of items, and the pipe character (|) indicates OR. These characters are often used together to indicate that an attack must include one item from the group. In this example, the attack must contain the word aux, lpt1, con, or prn after the string /examples/servlet/.

Notice that this example uses a group. The packet capture displays the signature pattern as /examples/servlet/AUX. AUX is a Windows device. You have good reason to be on guard for attempts to exploit LPT1, CON, and PRN devices.

4. Test the attack object.

#### Related Documentation

- Reproducing an Attack on page 6
- Discovering the Attack Signature on page 8
- Creating a Signature Attack Object on page 9
- Testing a Custom Attack Object on page 33

## Example: UNIX CDE/dtlogin Vulnerability

In this example, your network includes several user workstations and servers running UNIX. Many UNIX operating systems use the Common Desktop Environment (CDE) as a graphical user interface. Your security administrator notifies you of a new vulnerability in the dtlogin process for CDE (the dtlogin process handles a GUI login process to CDE).

The CERT advisory for the vulnerability (<http://www.kb.cert.org/vuls/id/179804>) contains the following information:

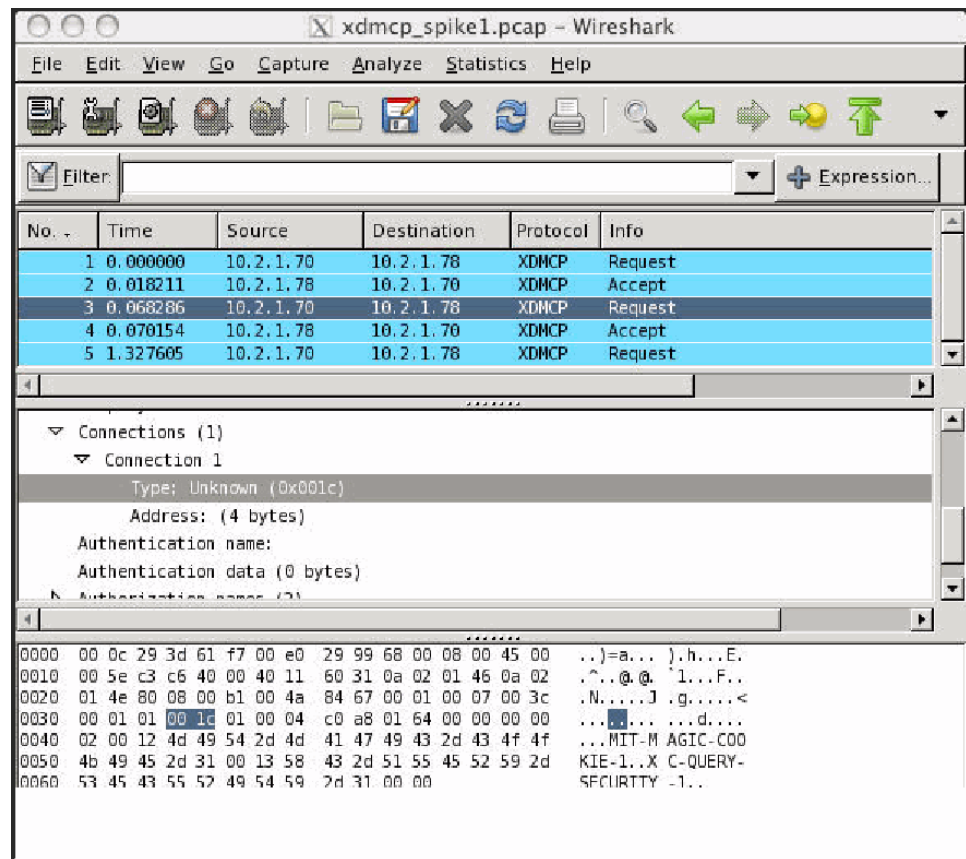
...The dtlogin program contains a "double-free" vulnerability that can be triggered by a specially crafted X Display Manager Control Protocol (XDMCP) packet... Block XDMCP traffic (177/udp) from untrusted networks such as the Internet...

From this information, you know that the attack uses XDMCP protocol packet, and runs on UDP/177. Now you must locate the attack code. The advisory also includes references that link to more information about the attack. One reference, <http://lists.immunitysec.com/pipermail/dailydave/2004-March/000402.html>, indicates that the person who first reported the attack has also written a script that replicates the attack. Obtain the script and move it to the attacker computer in your test lab.

To develop this attack object:

1. Reproduce the attack to determine the attack context, direction, and pattern. Ideally, use **scio ccap** and Wireshark concurrently so you have to run the attack only once. Figure 16 on page 51 shows the packet capture in Wireshark.

Figure 16: XDMP Packet Capture



2. Discover the elements of the attack signature:
  - Service. You know from the CERT advisory that the attack uses the XDMP protocol. Review the packet capture in Wireshark to confirm the protocol.
  - Context. Use **scio ccap** to determine whether you can match a particular service context. In this example, the XDMP service contexts are not supported by the IDP system, and the output of **scio ccap** is blank. You must specify the packet context for the attack.
  - Pattern. Using your knowledge of the XDMP protocol, you identify that the attack uses a non-NUL character (hexadecimal code 00 1b) to specify the connection type, which is invalid (the NUL character represents the Internet connection type in XDMP). To anchor the non-NUL character in a signature pattern, include some of the preceding bytes as part of the pattern. For this example, you choose to anchor the non-NUL character with the version number (hexadecimal code 00 01) and the request options code (hexadecimal code 00 07). The full attack pattern is 00 01

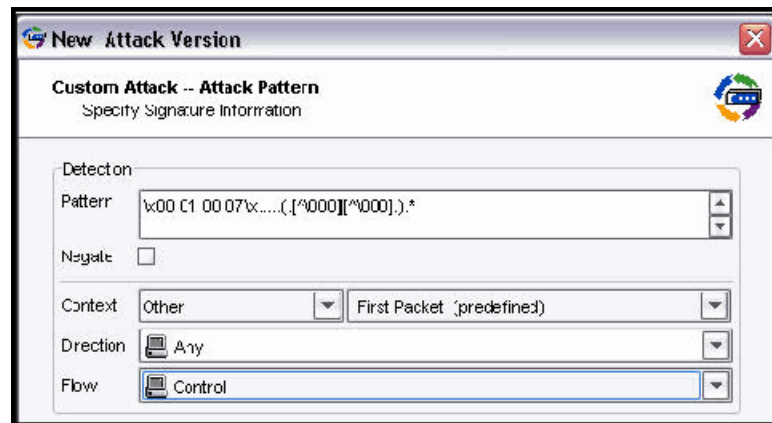
00 07 followed by five characters of any type, followed by a sixth character and either a non-NUL character (as shown above with 00 1b) or a non-NUL character and another character.

- Direction. Locate the source IP that initiated the session. In this example, you cannot determine the attack direction.
3. Create an attack object to match the attack signature. Use the following regular expression to match the signature:

```
\x00 01 00 07\x.....(.[^\000][^\000])..*
```

Figure 17 on page 52 shows the Custom Attack – Attack Pattern page for this example.

Figure 17: XDMP Attack Pattern



In this expression:

- The \x expression indicates a hexadecimal value.
- The numbers 00 01 00 07 in the XDMP protocol represent the version number (hexadecimal code 00 01 and the request options code (hexadecimal code 00 07).
- The five periods (.....) indicate five characters of any kind.
- The parentheses ( ) indicates a group of items, and the pipe character (|) indicates OR. These characters are often used together to indicate that an attack must include one item from the group.
- The opening and closing brackets combined with a caret [ ^ indicates negation.
- The backslash combined with a zero (\0) indicates an octal code number.

- The 00 characters are hexadecimal code for a NUL character. In this example, the attack must contain a non-NUL character, either preceded or followed by another character ([^\000] or [^\000]).
  - The dot star combination (.\* ) indicates a wildcard match. When used at the end of an expression, the wildcard indicates that anything can follow the specified expression.
4. Test the attack object.

**Related Documentation**

- Reproducing an Attack on page 6
- Discovering the Attack Signature on page 8
- Creating a Signature Attack Object on page 9
- Testing a Custom Attack Object on page 33

---

## Example: Detecting a Worm

Worms and Trojans often bypass firewalls and other traditional security measures to enter a network. In this example, you create a custom attack object to detect the Blaster worm on your network.

The CERT advisory (<http://www.cert.org/advisories/CA-2003-20.html>) for the Blaster worm gives the following information:

The W32/Blaster worm exploits a vulnerability in Microsoft's DCOM RPC interface..."

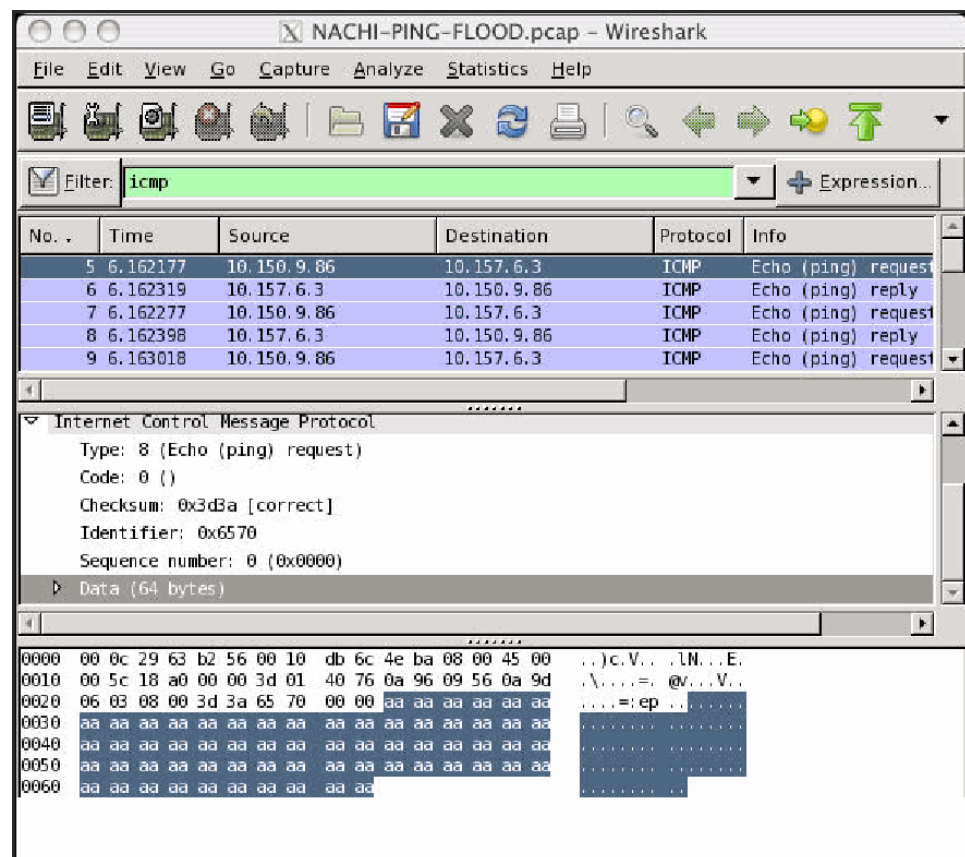
From this information, you know that the attack uses DCOM exploit, a previously identified security hole. Now you must locate the attack code. The advisory also includes references that link to more information about the attack. Unfortunately, none of the referenced Web pages contain exploit code. After searching the Web using the information you learned from the CERT advisory, you locate exploit code on PacketStorm (<http://packetstormsecurity.com/0307-exploits/dcom.c>).

To develop this attack object:

1. Reproduce the attack to determine the attack context, direction, and pattern. Ideally, use **scio ccap** and Wireshark concurrently so you have to run the attack only once.

Figure 18 on page 54 shows the packet capture on in Wireshark.

Figure 18: Blaster Worm Packet Capture



2. Discover the elements of the attack signature:

- Service. You know from the CERT advisory that the attack uses ICMP, for which the IDP OS does not support service contexts. Review the packet capture to confirm the protocol as ICMP.
- Context. Use **scio ccap** to determine whether we can match a particular service context. In this example, the ICMP service contexts are not supported by the IDP system, and the output of **scio ccap** is blank. You must specify the first packet context for the attack.
- Pattern. Select the first frame listed in Wireshark and review the information in the second section. Because you know that ICMP packets should not contain data, you investigate the 64 byte data payload. You can easily see the irregular payload is multiple "AA" characters, which is probably code attempting to overflow a buffer.



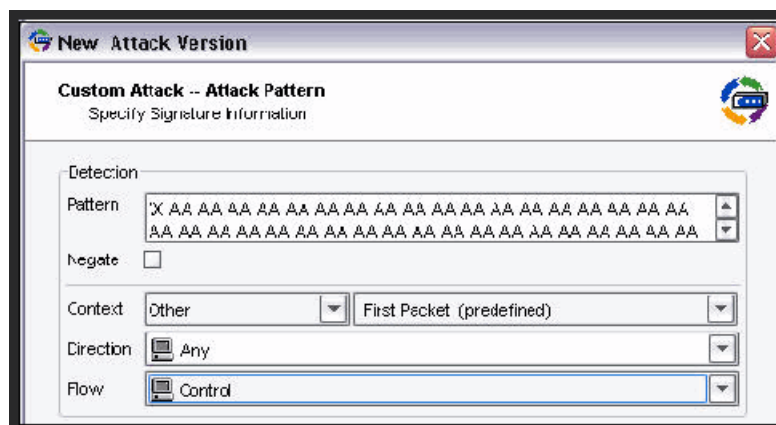
Because this pattern is unusual in the context of an ICMP packet, select it as your signature.

- **Direction.** Locate the source IP that initiated the session. In this example, you cannot determine the attack direction.
- Create an attack object to match the attack signature. In this example, we use the following regular expression to match the signature:

\X AA  
 AA  
 AA \X.\*

Figure 19 on page 55 shows the Custom Attack – Attack Pattern page for this example.

Figure 19: Blaster Worm Attack Pattern



In this expression:

- The \X expression indicates that a hexadecimal value is to follow.
  - The dot star combination (.\* ) indicates a wildcard match. When used at the end of an expression, the wildcard indicates that anything can follow the specified expression.
- Test the attack object.

## Related Documentation

- Reproducing an Attack on page 6
- Discovering the Attack Signature on page 8
- Creating a Signature Attack Object on page 9
- Testing a Custom Attack Object on page 33

## Example: Compound Signature to Detect Exploitation of an HTTP Vulnerability

Some attacks are crafted to appear benign when viewed at a packet-by-packet level. For these attacks, you can create a compound signature that detects multiple signature patterns in multiple contexts (service, nonservice, or both).

In this example, you have a Web server that uses Microsoft FrontPage Server extensions. Your security administrator notifies you of a new buffer overflow vulnerability in the FrontPage Server extensions.

The BugTraq advisory for the vulnerability (<http://www.securityfocus.com/bid/9007/discussion/>) contains the following information:

Microsoft FrontPage Server Extensions are prone to a remotely exploitable buffer overrun vulnerability ... It is possible to trigger this condition with a

chunked-encoded HTTP POST request...

The following proof-of-concept example is also provided:

```
POST /_vti_bin/_vti_aut/fp30reg.dll HTTP/1.1
Transfer-Encoding: chunked
PostLength
PostData
0
```

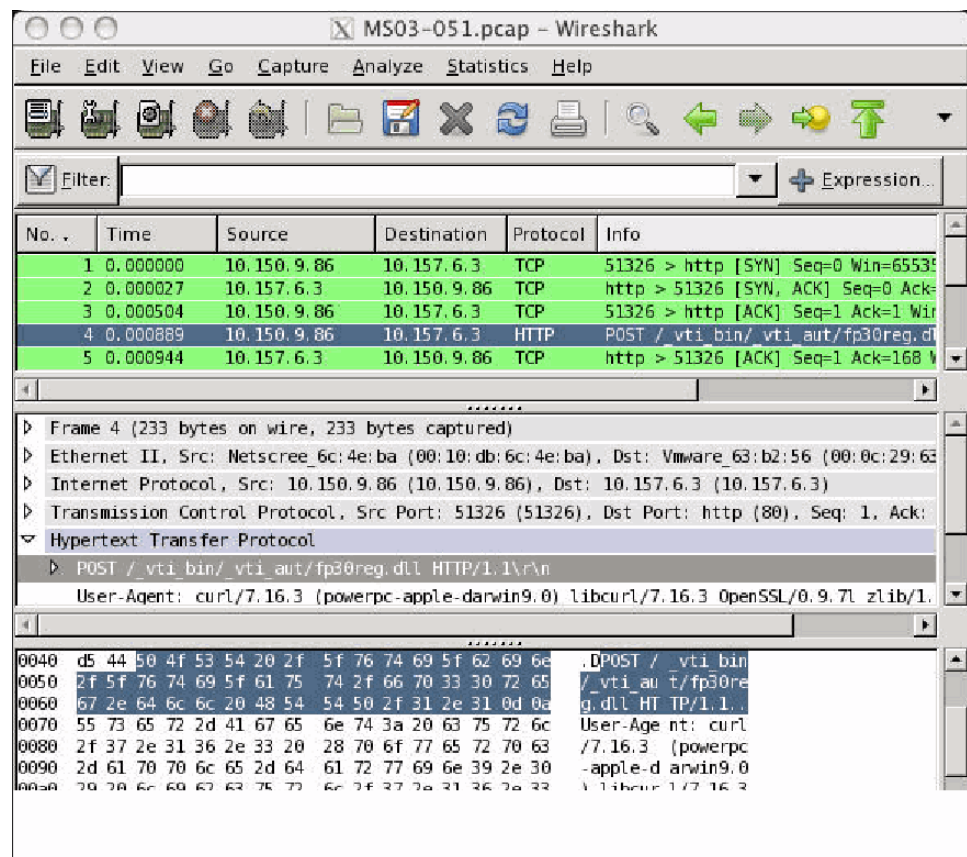
Additionally, a link to the compiled exploit is included.

From this information, you know that the attack uses the HTTP protocol and that at least part of the attack uses the POST method. Use the link to the compiled exploit to obtain the script, and move it to the attacker computer in your test lab.

To develop this attack object:

1. Reproduce the attack to determine the attack context, direction, and pattern. Ideally, use **scio ccap** and Wireshark concurrently so you only have to run the attack only once. Figure 20 on page 57 shows the packet capture in Wireshark.

Figure 20: HTTP Packet Capture



2. Discover the elements of the attack signature:
  - Service. You know from the BugTraQ advisory that the attack uses the HTTP protocol. Review the packet capture and locate the HTTP protocol usage.
  - Context. Use **scio ccap** to determine whether you can match a particular service context. In this example, the service context is HTTP URL Parsed.
  - Pattern. You quickly identify the signature pattern POST /\_vti\_bin/\_vti\_aut/fp30reg.dll within the HTTP service. However, because this pattern might trigger false positives, you also determine a second signature pattern to ensure that your rule detects only the attack. In this case, the second signature (noted in the BugTraQ advisory) is **Transfer-Encoding: chunked**.
  - Direction. Locate the source IP that initiated the session. In this example, the attack direction for both signature patterns is client-to-server.

3. Create an attack object to match the attack signature. Use the following regular expression to match the first signature:

```
\[/_vti_bin/_vti_aut/fp30reg\.dll\].*
```

Figure 21 on page 58 shows the Custom Attack – Attack Pattern page for this example.

Figure 21: First Attack Pattern



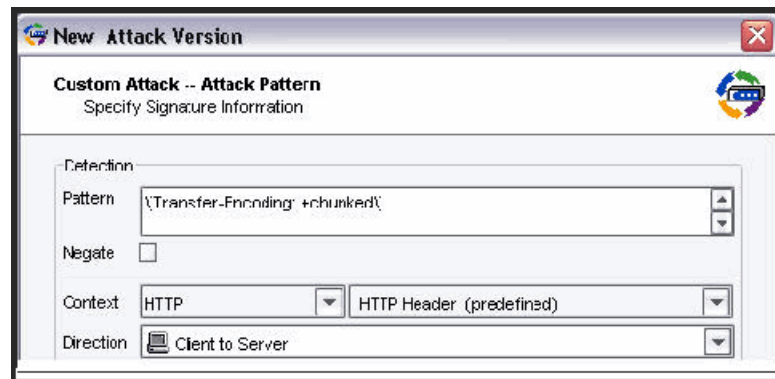
In this expression:

- The opening bracket (\[) indicates the beginning of a case-insensitive match for all characters until the closing bracket appears.
  - The pattern /\_vti\_bin/\_vti\_aut/fp30reg is a direct character match.
  - The backslash combined with the period (\.) indicates that the period is escaped (the period appears in the pattern).
  - The closing bracket (\]) indicates the end of a case-insensitive match.
  - The period combined with the asterisk character (.\* ) indicates that one or more characters must appear.
4. Add a second signature. Use the following regular expression to match the second signature:

```
\[Transfer-Encoding: +chunked\]
```

Figure 22 on page 59 shows the Custom Attack – Attack Pattern page for this example.

Figure 22: Second Attack Pattern



In this expression:

- The opening bracket (\\) indicates the beginning of a case-insensitive match for all characters until the closing bracket appears.
- The pattern Transfer-Encoding: is a direct character match.
- The plus sign (+) indicates that a space character must appear one or more times within the pattern.
- The pattern chunked is a direct character match.
- The closing bracket (\\) indicates the end of a case-insensitive match.

5. Test the attack object.

#### Related Documentation

- Reproducing an Attack on page 6
- Discovering the Attack Signature on page 8
- Creating a Compound Attack Object on page 29
- Testing a Custom Attack Object on page 33



## CHAPTER 5

# Reference Tables

- Reference: Custom Attack Object Protocol Numbers on page 61
- Reference: Custom Attack Object Service Properties on page 67
- Reference: Custom Attack Object Service Contexts on page 70
- Reference: Nonprintable and Printable ASCII Characters on page 106

### Reference: Custom Attack Object Protocol Numbers

---

Table 26 on page 61 protocol numbers used in the IDP system.

**Table 26: IDP Attack Objects: Protocol Numbers**

Protocol Name	Protocol Number
HOPOPT	0
ICMP	1
IGMP	2
GGP	3
IPIP	4
ST	5
TCP	6
CBT	7
EGP	8
IGP	9
BBN-RCC-MON	10
NVP-II	11

Table 26: IDP Attack Objects: Protocol Numbers (*continued*)

Protocol Name	Protocol Number
PUP	12
ARGUS	13
EMCON	14
XNET	15
CHAOS	16
UDP	17
MUX	18
DCN-MEAS	19
HMP	20
PRM	21
XND-IDP	22
TRUNK-1	23
TRUNK-2	24
LEAF-1	25
LEAF-2	26
RDP	27
IRTP	28
ISO-TP4	29
NETBLT	30
MFE-NSP	31
MERIT-INP	32
SEP	33
3PC	34
IDPR	35



Table 26: IDP Attack Objects: Protocol Numbers (*continued*)

Protocol Name	Protocol Number
XTP	36
DDP	37
TP_PLUS_PLUS	39
IL	40
IPV6	41
SDRP	42
IPV6-ROUTING	43
IDV6-FRAGMENT	44
IDRP	45
RSVP	46
GRE	47
MHRP	48
BNA	49
ESP	50
AH	51
I-NLSP	52
SWIPE	53
NARP	54
MOBILE	55
TLSP	56
SKIP	57
IPV6-ICMP	58
IPV6-NONXT	59
IPV6-OPTS	60

Table 26: IDP Attack Objects: Protocol Numbers (*continued*)

Protocol Name	Protocol Number
AHIP	61
CFTP	62
ALNP	63
SAT-EXPAK	64
KRYPTOLAN	65
RVD	66
IPPC	67
ADFSP	68
SAT-MON	69
VISA	70
IPCV	71
CPNX	72
CPHB	73
WSN	74
PVP	75
BR-SAT-MON	76
SUN-ND	77
WB-MON	78
WB-EXPAK	79
ISO-IP	80
VMTP	81
SECURE-VMTP	82
VINES	83
TTP	84

Table 26: IDP Attack Objects: Protocol Numbers (*continued*)

Protocol Name	Protocol Number
NSFNET-IBP	85
DGP	86
TCF	87
EIGRP	88
OSPFIGP	89
SPRITE-RPC	90
LARP	91
MTP	92
AX_25	93
IPIP	94
MICP	95
SCC-SP	96
ETHERIP	97
ENCAP	98
APES	99
GMTP	100
IFMP	101
PNNI	102
PIM	103
ARIS	104
SCPS	105
QNX	106
A/N	107
IPCOMP	108

Table 26: IDP Attack Objects: Protocol Numbers (*continued*)

Protocol Name	Protocol Number
SNP	109
COMPAT-PEER	110
IPZ-IN-IP	111
VRRP	112
PGM	113
HOP-O	114
L2TP	115
DDX	116
IATP	117
STP	118
SRP	119
UTI	120
SMP	121
SSM	122
PTP	123
ISIS	124
FIRE	125
CRTP	126
CRUDP	127
SSCOPMCE	128
IPLT	129
SPS	130
PIPE	131
SCTP	132

Table 26: IDP Attack Objects: Protocol Numbers (*continued*)

Protocol Name	Protocol Number
FC	133
RSVP-E2E-IGNORE	134
n/a	
n/a	
n/a	
RESERVED	255

## Reference: Custom Attack Object Service Properties

Table 27 on page 67 lists properties of service objects that are part of a custom attack object definition, including the standard ports associated with the service. With application identification enabled, the IDP system detects the service regardless of port.

Table 27: IDP Attack Object: Service Properties

Service	Standard Port	Line?	Transaction?
AIM	TCP/5190	No	No
ARP		No	No
CHARGEN	UDP/19	No	No
DHCP	UDP/68 UDP/67	No	No
DISCARD	UDP/9	No	No
DNS	TCP/53 and UDP/53	No	Yes
ECHO	UDP/7	Yes	No
FINGER	TCP/79	Yes	No
FTP	TCP/20 and TCP/21	Yes	No
GNUTELLA	TCP/6346 TCP/6347 UDP/6346 UDP/6347	No	No
GOPHER	TCP/70	No	No

Table 27: IDP Attack Object: Service Properties (*continued*)

Service	Standard Port	Line?	Transaction?
H225RAS	UDP/1718 UDp/1719	No	No
H225SGN	TCP/1720	No	No
HTTP	TCP/80	Yes	Yes
ICMP		No	No
IDENT	TCP/113	Yes	No
IEC104	TCP/113	Yes	No
IKE	UDP/500	No	No
IMAP	TCP/143	Yes	No
IRC	TCP/6667	Yes	No
LDAP	TCP/369	No	Yes
LPR	TCP/515	Yes	No
MGCP	UDP/2427 UDP/2727	No	No
MODBUS	TCP/502	No	No
MSN	TCP/1863	Yes	No
MSRPC	UDP/135 TCP/135	No	Yes
MSSQL	TCP/1433	No	No
MYSQL	TCP/3306	No	No
NBDS	TCP/445	No	No
NBNAME	TCP/137	No	No
NFS	UDP/2049 TCP/2049	No	Yes
NNTP	TCP/119	Yes	No

Table 27: IDP Attack Object: Service Properties (*continued*)

Service	Standard Port	Line?	Transaction?
NONE	No service used for packet, stream, or line	No	No
NTP	UDP/123	No	No
POP3	TCP/110	Yes	No
PORTMAPPER	TCP/111 UDP/111	No	Yes
RADIUS	TCP/1812	No	Yes
REXEC	TCP/512	No	No
RLOGIN	TCP/513	No	No
RSH	TCP/514	No	No
RTP		Yes	No
RTSP	TCP/554 UDP/554	No	No
RTPVIDEO		Yes	No
RUSERS		No	Yes
SCAN		No	No
SIP	TCP/5060 UDP/5060	Yes	No
SMB	TCP/138 TCP/139	No	No
SMTP	TCP/25	Yes	Yes
SNMP	UDP/161	No	Yes
SNMPTRAP	UDP/162	No	No
SQLMON	UDP/1434	No	No
SSH	TCP/22	No	No
SSL	TCP/443	No	No

Table 27: IDP Attack Object: Service Properties (*continued*)

Service	Standard Port	Line?	Transaction?
STP		No	No
SYSLOG	UDP/514	No	No
TELNET	TCP/23	No	No
TFTP	UDP/69	No	No
TNS	TCP/1521 TCP/2483 TCP/1525 TCP/1527 TCP/1529	No	Yes
VNC	TCP	Yes	No
WHOIS	TCP/43	Yes	No
YMSG	TCP/5050	No	No

## Reference: Custom Attack Object Service Contexts

The Juniper Networks Security Center team develops predefined service contexts you can use when you create custom attack objects. The predefined service contexts are added or modified during routine signature updates.

The following tables describe the predefined service contexts that are available in the NSM custom attack object wizard.

Table 28: Service Contexts: AIM

Context and Direction	Description
aim-auth-request-msg (ANY)	Matches the message sent from one user to another when requesting authorization to add to the buddy list.
aim-away-message (CTS)	Matches the message sent to other clients when a user changes status to 'away'.
aim-buddy-comment (ANY)	Matches the comment stored for a buddy in the contact list.
aim-capabilities (ANY)	Matches the set of features supported by the client.
aim-chat-info (STC)	Matches the information about a chatroom.
aim-chat-interests (STC)	Matches the categories of personal interests in a user's profile.
aim-chat-room-desc (STC)	Matches the description of a chatroom.
aim-chat-room-name (STC)	Matches the name of a chatroom in an AIM/ICQ session.



Table 28: Service Contexts: AIM (*continued*)

Context and Direction	Description
aim-client-ip (STC)	Matches the IP address of the client for direct P2P communication.
aim-client-port (STC)	Matches the port that the client is listening on for P2P communication.
aim-client-status (STC)	Matches the user's online status.
aim-decline-reason (ANY)	Matches the decline reason when a client refuses to be added to another user's contact list.
aim-described-url (ANY)	Matches the description and URL when sending a Web page to another address.
aim-email-address (STC)	Matches the e-mail address of a user as it appears in the profile.
aim-error-url (STC)	Matches the URL on the server where the user can reconfigure the account password.
aim-gcard-message (ANY)	Matches the message associated with a greeting card.
aim-gcard-recipient (ANY)	Matches the screen name of a greeting card recipient.
aim-gcard-sender (ANY)	Matches the screen name of a greeting card sender.
aim-gcard-theme (ANY)	Matches the theme of a greeting card sent from one client to another.
aim-gcard-title (ANY)	Matches the title of a greeting card sent from one user to another.
aim-gcard-url (ANY)	Matches the URL of the greeting card sent from one user to another.
aim-get-file (STC)	Matches the name of a file that the user is transferring from a peer.
aim-group (ANY)	Matches the name of a group of items (usually buddies).
aim-info-text (STC)	Matches additional information text that appears in a user's profile.
aim-local-ip (CTS)	Matches the IP address of a client used for P2P communication.
aim-local-port (CTS)	Matches the local port that the client is listening on for P2P communication.
aim-message-block (ANY)	Matches the instant message sent from one user to another.
aim-message-description (ANY)	Matches the description of a message.
aim-nick-name (ANY)	Matches the nickname of an AIM/ICQ user.
aim-oft-content (ANY)	Matches the contents of a file being transferred between peers.
aim-oft-name (ANY)	Matches the name of a file being transferred between peers.

**Table 28: Service Contexts: AIM (*continued*)**

Context and Direction	Description
aim-peer-ip (STC)	Matches the IP address of a peer for direct P2P communication.
aim-peer-port (STC)	Matches the port of a peer for direct P2P communication.
aim-put-file (CTS)	Matches the name of a file that the user is transferring to a peer.
aim-screen-name (ANY)	Matches the screen name of a user.
aim-server-ip (STC)	Matches the IP address of a server. Typically used when the main server redirects the client to another server.
aim-server-url (STC)	Matches any URL on the server.
aim-url (ANY)	Matches the URL of a user's profile.
aim-xml-value (STC)	Matches the XML string sent by the server with the value of a requested URL.

**Table 29: Service Contexts: BGP**

Context and Direction	Description
bgp-keepalive-msg (ANY)	Matches the BGP keep alive message.
bgp-message (ANY)	Matches any BGP message.
bgp-notification-msg (ANY)	Matches the BGP notification message.
bgp-open-msg (ANY)	Matches the BFP open message.
bgp-open-no-parm (ANY)	Matches the BFP open message without optional parameters.
bgp-open-parm (ANY)	Matches the optional parameters in the BGP open message.
bgp-update-attr-aggregator (ANY)	Matches the Aggregator path attribute data in the BGP update message.
bgp-update-attr-as-path (ANY)	Matches the AS path attribute data in the BGP update message.
bgp-update-attr-atomic-aggr (ANY)	Matches the atomic-aggregator path attribute data in the BGP update message.
bgp-update-attr-cluster-list (ANY)	Matches the Cluster-List path attribute data in the BGP update message.
bgp-update-attr-communities (ANY)	Matches the Communities path attribute data in the BGP update message.
bgp-update-attr-local-pref (ANY)	Matches the Local-Pref path attribute data in BGP update message.
bgp-update-attr-med (ANY)	Matches the Multi-Exit-Disc path attribute data in the BGP update message.

Table 29: Service Contexts: BGP (*continued*)

Context and Direction	Description
bgp-update-attr-next-hop (ANY)	Matches the Next-Hop path attribute data in the BGP update message.
bgp-update-attr-nonstd (ANY)	Matches any Non-Standard path attribute data in the BGP update message.
bgp-update-attr-rigin (ANY)	Matches the Origin path attribute data in the BGP update message.
bgp-update-attr-originator (ANY)	Matches the Originator path attribute data in BGP update message.
bgp-update-msg (ANY)	Matches the BGP update message.
bgp-update-nlri_infor (ANY)	Matches the Network Layer Reachability Information in the BGP update message.
bgp-update-norm-unfeasible-rte (ANY)	Matches the unfeasible routes data in BGP update message. This context shows each route expanded to 4 bytes, prefixed by a delimiter.
bgp-update-total-path-attribute (ANY)	Matches the Total Path Attribute data in the BGP update message.
bgp-update-unfeasible-rts (ANY)	Matches the unfeasible routes data in the BGP update message.

Table 30: Service Contexts: DHCP

Context and Direction	Description
dhcp-file-name (ANY)	Matches the filename in a DHCP/bootp message.
dhcp-option (ANY)	Matches each option in a DHCP/bootp message. Each option context contains the type and length of the option.
dhcp-server-name (ANY)	Matches the server name in a DHCP/bootp message.

Table 31: Service Contexts: DNS

Context and Direction	Description
dns-cname (ANY)	Matches the CNAME in a DNS request or response.
dns-rr-a6-rdata (ANY)	Match the rdata of an A6 RR in a DNS request response.
dns-rr-afsdB-rdata (ANY)	Matches the rdata of an AFSDB RR in a DNS request or response.
dns-rr-apl-rdata (ANY)	Matches the rdata of an APL RR in a DNS request or response.
dns-rr-atma-rdata (ANY)	Matches the rdata of an ATMA RR in a DNS request or response.
dns-rr-cname-rdata (ANY)	Matches the rdata of a CNAME RR in a DNS request or response.

Table 31: Service Contexts: DNS (*continued*)

Context and Direction	Description
dns-rr-dnskey-rdata (ANY)	Matches the rdata of a DNSKEY RR in a DNS request or response.
dns-rr-ds-rdata (ANY)	Matches the rdata of a DS RR in a DNS request or response.
dns-rr-eid-rdata (ANY)	Matches the rdata of an EID RR in a DNS request or response.
dns-rr-hinfo-rdata (ANY)	Matches the rdata of an HINFO RR in a DNS request or response.
dns-rr-key-rdata (ANY)	Matches the rdata of a KEY RR in a DNS request or response.
dns-rr-kx-rdata (ANY)	Matches the rdata of a KX RR in a DNS request or response.
dns-rr-mb-rdata (ANY)	Matches the rdata of an MB RR in a DNS request or response.
dns-rr-md-rdata (ANY)	Matches the rdata of an MD RR in a DNS request or response.
dns-rr-mf-rdata (ANY)	Matches the rdata of an MF RR in a DNS request or response.
dns-rr-mg-rdata (ANY)	Matches the rdata of an MG RR in a DNS request or response.
dns-rr-minfo-rdata (ANY)	Matches the rdata of an MINFO RR in a DNS request or response.
dns-rr-mr-rdata (ANY)	Matches the rdata of an MR RR in a DNS request or response.
dns-rr-mx-rdata (ANY)	Matches the rdata of an MX RR in a DNS request or response.
dns-rr-naptr-rdata (ANY)	Matches the rdata of a NAPTR RR in a DNS request or response.
dns-rr-nimloc-rdata (ANY)	Matches the rdata of an NIMLOC RR in a DNS request or response.
dns-rr-ns-rdata (ANY)	Matches the rdata of an NS RR in a DNS request or response.
dns-rr-nsap-rdata (ANY)	Matches the rdata of an NSAP RR in a DNS request or response.
dns-rr-ns-rdata (ANY)	Matches the rdata of an NS RR in a DNS request or response.
dns-rr-nsapptr-rdata (ANY)	Matches the rdata of an NSAPPTR RR in a DNS request or response.
dns-rr-nsec-rdata (ANY)	Matches the rdata of an NSEC RR in a DNS request or response.
dns-rr-null-rdata (ANY)	Matches the rdata of a NULL RR in a DNS request or response.
dns-rr-nxt-rdata (ANY)	Matches the rdata of a NXT RR in a DNS request or response.
dns-rr-ptr-rdata (ANY)	Matches the rdata of a PTR RR in a DNS request or response.
dns-rr-px-rdata (ANY)	Matches the rdata of a PX RR in a DNS request or response.

**Table 31: Service Contexts: DNS (*continued*)**

Context and Direction	Description
dns-rr-rp-rdata (ANY)	Matches the rdata of an RP RR in a DNS request or response.
dns-rr-rrsig-rdata (ANY)	Matches the rdata of an RRSIG RR in a DNS request or response.
dns-rr-sig-rdata (ANY)	Matches the rdata of an SIG RR in a DNS request or response.
dns-rr-soa-rdata (ANY)	Matches the rdata of an SOA RR in a DNS request or response.
dns-rr-sshfp-data (ANY)	Matches the rdata of an SSHFP RR in a DNS request or response.
dns-rr-tsip-rdata (ANY)	Matches the rdata of a TSIP RR in a DNS request or response.
dns-rr-txt-rdata (ANY)	Matches the rdata of a TXT RR in a DNS request or response.
dns-rr-type-rdata (ANY)	Matches the entire resource record in a DNS request or response, including the type and class.
dns-rr-wks-rdata (ANY)	Matches the rdata of a WKS RR in a DNS request or response.
dns-type-name (ANY)	Matches any name resource record in a DNS request or response. The first 2 bytes of the context contain the RFC-1035 type values.
dns-update-header	Matches the header of a DNS UPDATE request or response.

**Table 32: Service Contexts: Finger**

Context and Direction	Description
finger-host (CTS)	Matches each hostname in a FINGER request.
finger-user (CTS)	Matches the username in a FINGER request.

**Table 33: Service Contexts: First Data Packet**

Context and Direction	Description
first-data-packet (ANY)	Matches the first data packet of a session.
first-packet (ANY)	Matches the first packet of a session.

**Table 34: Service Contexts: FTP**

Context and Direction	Description
ftp-account (CTS)	Matches the FTP login account name.
ftp-banner (STC)	Matches the banner returned by the server at the start of an FTP session.

Table 34: Service Contexts: FTP (*continued*)

Context and Direction	Description
ftp-command (CTS)	Matches each of the FTP command names.
ftp-cwd-pathname (CTS)	Matches the directory name in the CWD command of an FTP session.
ftp-dele-pathname (CTS)	Matches the file name in the DELE command of an FTP session.
ftp-get-filename (CTS)	Matches the filename in the GET command of an FTP session.
ftp-list-pathname (CTS)	Matches the directory or file name in the LIST command of an FTP session.
ftp-mkd-pathname (CTS)	Matches the directory name in the MKD command of an FTP session.
ftp-nlst-pathname (CTS)	Matches the directory or file name in the NLST command of an FTP session.
ftp-password (CTS)	Matches the FTP login password.
ftp-pathname (CTS)	Matches a directory or file name in any of the FTP commands.
ftp-put-filename (CTS)	Matches the filename in the PUT command of an FTP session.
ftp-reply-100-line (STC)	Matches the FTP 1yz Positive Preliminary reply.
ftp-reply-200-line (STC)	Matches the FTP 2yz Positive Completion reply.
ftp-reply-300-line (STC)	Matches the FTP 3yz Positive Intermediate reply.
ftp-reply-400-line (STC)	Matches the FTP 4yz Transient Negative Completion reply.
ftp-reply-500-line (STC)	Matches the FTP 5yz Permanent Negative Completion reply.
ftp-reply-line (STC)	Matches the FTP reply line.
ftp-request (CTS)	Matches FTP request line (command and arguments).
ftp-rmd-pathname (CTS)	Matches the directory name in the RMD command of an FTP session.
ftp-rnfr-pathname (CTS)	Matches a directory or file name in the RNFR command of an FTP session.
ftp-rnto-pathname (CTS)	Matches a directory or file name in the RNTO command of an FTP session.
ftp-sitestring (CTS)	Matches the arguments of the SITE command in an FTP session.
ftp-smnt-pathname (CTS)	Matches the directory or file name in the SMNT command of an FTP session.
ftp-stat-pathname (CTS)	Matches the directory or file name in the STAT command of an FTP session.
ftp-username (CTS)	Matches the FTP login user name.

**Table 35: Service Contexts: Gnutella**

Context and Direction	Description
gnutella-connect-fail-reason (STC)	Matches the connection fail reason string in a Gnutella connection.
gnutella-connect-header (ANY)	Matches the contents of the HTTP style CONNECT message in a Gnutella session.
gnutella-http-get-filename (CTS)	Matches the name of the file that the client intends to retrieve.
gnutella-http-header (ANY)	Matches any HTTP style headers in a Gnutella session.
gnutella-queryhit-vendor (STC)	Matches the 4-byte vendor code in the reply for the QUERYHIT message.
gnutella-search-criteria (CTS)	Matches the search criteria in a QUERY message of a Gnutella session.
gnutella-user-agent (ANY)	Matches the name of the user agent in a Gnutella session.

**Table 36: Service Contexts: Gopher**

Context and Direction	Description
gopher-display (STC)	Matches the display string of a Gopher item.
gopher-file (STC)	Matches the contents of a Gopher item/file.
gopher-host-port (STC)	Matches the host and port used to get an item.
gopher-selector (STC)	Matches the selector string of a Gopher item.

**Table 37: Service Contexts: H225**

Context and Direction	Description
h225ras-admission (ANY)	Matches H225RAS admission messages (AdmissionConfirm, AdmissionReject, AdmissonRequest).
h225ras-bandwidth (ANY)	Matches H225RAS bandwidth messages (BandwidthConfirm, BandwidthReject, BandwidthRequest).
h225ras-command-state (ANY)	Matches the state of the H225RSA connection.
h225ras-disengage (ANY)	Matches H225RAS disengage messages (DisengageConfirm, DisengageReject, DisengageRequest).
h225ras-gatekeeper (ANY)	Matches H225RAS gatekeeper messages (GatekeeperConfirm, GatekeeperReject, GatekeeperRequest).
h225ras-info (ANY)	Matches H225RAS informational messages (InfoRequestAck, InfoRequestResponse, InfoRequest).

**Table 37: Service Contexts: H225 (*continued*)**

Context and Direction	Description
h225ras-location (ANY)	Matches H225RAS location messages (LocationConfirm, LocationReject, LocationRequest).
h225ras-message (ANY)	Matches the broad H225RAS message context.
h225ras-nonstandard (ANY)	Matches the H225RAS nonstandard message context.
h225ras-registration (ANY)	Matches the H225RAS registration message.
h225ras-resource (ANY)	Matches H225RAS resources available messages (ResourcesAvailableConfirm, ResourcesAvailableIndicate).
h225ras-rip (STC)	Matches the H225RAS request- in-progress message.
h225ras-servicecontrol (CTS)	Matches the H225RAS service control message.
h225ras-unknown-message (ANY)	Match the H225RAS Unknown message type.
h225ras-unregistration (ANY)	Matches the H225RAS unregistration message.
h225ras-unspecified-message (ANY)	Matches the H225RAS unspecified message.
h225ras-version (ANY)	Matches the H225RAS version message.
h225sgn-message (ANY)	Matches the H225SGN message body started with the message-type byte.
h225sgn-preamble (ANY)	Matches the H225SGN signaling protocol discriminator and call reference value.

**Table 38: Service Contexts: HTTP**

Context and Direction	Description
http-authorization (CTS)	Matches the username and password decoded from the Authorization: Basic header in an HTTP request.
http-data (ANY)	Matches any HTTP data in an HTTP transaction that is not text/html, text/plain, or FORM values in a POST request.
http-first-data-chunk (ANY)	Matches the first data chunk in an HTTP transaction.
http-form-data (CTS)	Matches each of the form values in a POST request of an HTTP transaction.
http-get-url (CTS)	Matches the URL in an HTTP get request as it appears in the stream.
http-get-url-parsed (CTS)	Matches the decoded, normalized URL in an HTTP get request.



Table 38: Service Contexts: HTTP (*continued*)

Context and Direction	Description
http-get-url-parsed-param (CTS)	Matches the decoded, normalized URL in an HTTP get request along with any CGI parameters.
http-get-url-parsed-param-parsed (CTS)	Matches the decoded, normalized URL in and HTTP GET request along with the any decoded CGI parameters.
http-head-url (CTS)	Matches the URL in an HTTP head request as it appears in the stream.
http-head-url-parsed (CTS)	Matches the decoded, normalized URL in an HTTP head request.
http-header (ANY)	Matches any HTTP header.
http-header-accept (CTS)	Matches each Accept: header in an HTTP request.
http-header-accept-encoding (CTS)	Matches each Accept-Encoding: header in an HTTP request.
http-header-accept-language (CTS)	Matches each Accept-Language: header in an HTTP request.
http-header-content-encoding (ANY)	Matches each Content-Encoding: header in an HTTP transaction.
http-header-content-language (ANY)	Matches each Content-Language: header in an HTTP transaction.
http-header-content-location (ANY)	Matches each Content-Location: header in an HTTP transaction.
http-header-content-md5 (ANY)	Matches each Content-MD5: header in an HTTP transaction.
http-header-content-type (ANY)	Matches each Content-Type: header in an HTTP transaction.
http-header-cookie (ANY)	Matches each Cookie: header in an HTTP transaction.
http-header-host (CTS)	Matches each Host: header in an HTTP request.
http-header-referer (CTS)	Matches each Referrer: header in an HTTP request.
http-header-server (STC)	Matches each Server: header in an HTTP reply.
http-header-soapaction (ANY)	Matches each soapaction: header in an HTTP transaction.
http-header-user-agent (CTS)	Matches each User-Agent: header in an HTTP request.
http-image (ANY)	Matches IMATE contents (BMP, PNG) in HTTP transaction.
http-jpeg-raw (ANY)	Matches JPEG content in HTTP transaction.
http-jpeg-tag (ANY)	Matches JPEG tag of JPEG content in HTTP transaction.
http-object-tag-clsid (STC)	Matches the CLSID of an object tag.

**Table 38: Service Contexts: HTTP (*continued*)**

Context and Direction	Description
http-param-parsed (CTS)	Matches the decoded CGI parameters in an HTTP request.
http-png-chunk (ANY)	Matches contents of PNG chunk to HTTP transaction.
http-post-url (CTS)	Matches the URL in an HTTP post request as it appears in the stream.
http-post-url-parsed (CTS)	Matches the decoded, normalized URL in an HTTP post request.
http-post-variable (CTS)	Matches each CGI variable in the form data of an HTTP POST request.
http-post-variable-parsed (CTS)	Matches each decoded CGI variable in the form data of an HTTP POST request.
http-request (CTS)	Matches each HTTP request line.
http-request-method (CTS)	Matches the method name in an HTTP request.
http-status (STC)	Matches the status line in an HTTP reply.
http-text-html (ANY)	Matches the text/html data in an HTTP transaction.
http-text-html-head (ANY)	Matches the header of text/html data in an HTTP transaction.
http-text-html-script (ANY)	Matches the script tag of text/html data in an HTTP transaction.
http-text-html-style (ANY)	Matches the style tag of text/html data in an HTTP transaction.
http-text-html-tag (ANY)	Matches any tag inside text/html data in an HTTP transaction.
http-text-plain (ANY)	Matches the text/plain data in an HTTP transaction.
http-text-soap (ANY)	Matches the text/soap data in and HTTP transaction.
http-text-xml (ANY)	Matches the tex/xml data in an HTTP transaction.
http-url (CTS)	Matches the URL in an HTTP request as it appears in the stream.
http-url-parsed (CTS)	Matches the decoded, normalized URL in an HTTP request.
http-url-variable (CTS)	Matches each CGI variable in the URL of an HTTP GET request.
http-url-variable-parsed (CTS)	Matches each decoded CGI variable in the URL of an HTTP GET request.
http-variable (CTS)	Matches each CGI variable in an HTTP GET or POST request.
http-variable-parsed (CTS)	Matches each decoded CGI variable in an HTTP GET or POST request.

**Table 39: Service Contexts: IEC**

Context and Direction	Description
iec104-message-type-i (ANY)	Matches the Type-I message of IEC104.
iec104-message-type-s (ANY)	Matches the Type-S message of IEC104.
iec104-message-type-u (ANY)	Matches the Type-U message of IEC104.

**Table 40: Service Contexts: IMAP**

Context and Direction	Description
imap-append (CTS)	Matches the e-mail contents in an IMAP append message.
imap-append-line (CTS)	Matches arguments of IMAP Append command line in an IMAP session.
imap-authenticate (CTS)	Matches arguments of IMAP Authenticate command in an IMAP session.
imap-banner-(STC)	Matches arguments of the first untagged OK response from an IMAP session.
imap-command (CTS)	Matches each IMAP command name in an IMAP session.
imap-command-line (CTS)	Matches each IMAP command name and arguments in an IMAP session.
imap-copy (CTS)	Matches arguments of IMAP Copy command in an IMAP session.
imap-create (CTS)	Matches arguments of IMAP Create command in an IMAP session.
imap-delete (CTS)	Matches arguments of IMAP Delete command in an IMAP session.
imap-deleteacl (CTS)	Matches arguments of IMAP DeleteACL command in an IMAP session.
imap-examine (CTS)	Matches arguments of IMAP Examine command in an IMAP session.
imap-fetch (CTS)	Matches arguments of IMAP Fetch command in an IMAP session.
imap-getacl (CTS)	Matches arguments of IMAP GetACL command in an IMAP session.
imap-list (CTS)	Matches arguments of IMAP List/RLIST command in an IMAP session.
imap-listrights (CTS)	Matches arguments of IMAP ListRights command in an IMAP session.
imap-login (CTS)	Matches arguments of IMAP Login command in an IMAP session.
imap-lsub (CTS)	Matches arguments of IMAP LSUB/RLSUB command in an IMAP session.
imap-mailbox (CTS)	Matches each mailbox name in an IMAP session.
imap-myrights (CTS)	Matches arguments of IMAP MyRights command in an IMAP session.

**Table 40: Service Contexts: IMAP (*continued*)**

Context and Direction	Description
imap-rename (CTS)	Matches arguments of IMAP Rename command in an IMAP session.
imap-search (CTS)	Matches arguments of IMAP Search command in an IMAP session.
imap-select (CTS)	Matches arguments of IMAP Select command in an IMAP session.
imap-setacl (CTS)	Matches arguments of IMAP SetACL command in an IMAP session.
imap-status (CTS)	Matches arguments of IMAP Status command in an IMAP session.
imap-store (CTS)	Matches arguments of IMAP Store command in an IMAP session.
imap-subscribe (CTS)	Matches arguments of IMAP Subscribe command in an IMAP session.
imap-uid (CTS)	Matches arguments of IMAP UID command in an IMAP session.
imap-unsubscribe (CTS)	Matches arguments of IMAP Unsubscribe command in an IMAP session.
imap-user (CTS)	Matches the IMAP user name in an IMAP session.

**Table 41: Service Contexts: IRC**

Context and Direction	Description
irc-command (ANY)	Matches any IRC command name.
irc-join-chan (ANY)	Matches the channel name in the JOIN command of an IRC session.
irc-nick-name (ANY)	Matches the name in the NICK command of an IRC session.
irc-notice-msg (ANY)	Matches the message in the NOTICE command of an IRC session.
irc-oper-name (ANY)	Matches the name in the OPER command of an IRC session.
irc-oper-password (ANY)	Matches the password in the OPER command of an IRC session.
irc-part-chan (ANY)	Matches the channel name in the PART command of an IRC session.
irc-password (ANY)	Matches the password in the PASS command of an IRC session.
irc-priv-msg (ANY)	Matches the message in the PRIVMSG command of an IRC session.
irc-real-name (ANY)	Matches the real name in the USER command of an IRC session.
irc-topic (ANY)	Matches the arguments of the TOPIC command of an IRC session.
irc-user-name (ANY)	Matches the name in the USER command of an IRC session.

Table 42: Service Contexts: LDAP

Context and Direction	Description
ldap-abandon-request (CTS)	Matches the entire Abandon Request message.
ldap-add-request (CTS)	Matches the entire Add Request message.
ldap-add-request-attribute (CTS)	Matches each attribute in an Add Request message. The values are NULL delimited and the type, and values are newline delimited.
ldap-add-request-attributetype (CTS)	Matches the type each attribute in an Add Request message.
ldap-add-request-attributevalue (CTS)	Matches the value of each attribute in an Add Request message.
ldap-add-request-entry (CTS)	Matches the object in an Add Request message.
ldap-bind-request (CTS)	Matches the entire LDAP Bind Request message.
ldap-bind-request-authentication (CTS)	Matches the authentication information in a Bind Request message including the 1-byte type.
ldap-bind-request-ldapDN (CTS)	Matches the name of the directory object to which the client wants to bind.
ldap-bind-request-version (CTS)	Matches the LDAP version in a Bind Request message.
ldap-compare-request (CTS)	Matches the entire Compare Request message.
ldap-compare-request-assertionvalue (CTS)	Matches the value against which the attribute value is compared in a Compare Request message.
ldap-compare-request-attributedesc (CTS)	Matches the attribute type of an entry in a Compare Request message.
ldap-compare-request-entry (CTS)	Matches the entry of the DN to be compared in a Compare Request message.
ldap-delete-request (CTS)	Matches the entire Delete Request message.
ldap-extended-request (CTS)	Matches the entire Extended Request message.
ldap-extended-request-requestName (CTS)	Matches the request name in the Extended Request message.
ldap-extended-request-requestValue (CTS)	Matches the request value in the Extended Request message.
ldap-extended-response-response (STC)	Matches the response field in the Extended Request message.
ldap-extended-response-responseName (STC)	Matches the response name in the Extended Response message.

Table 42: Service Contexts: LDAP (*continued*)

Context and Direction	Description
ldap-modify-request (CTS)	Matches the entire Modify Request message.
ldap-modify-request-attribute (CTS)	Matches each attribute in a Modify Request message including the 1-byte modify operation. The values are NULL delimited, and the type and values are newline delimited.
ldap-modify-request-attributetype (CTS)	Matches each attribute type in a Modify Request message.
ldap-modify-request-attributevalue (CTS)	Matches each attribute value in a Modify Request message.
ldap-modify-request-object (CTS)	Matches the object in the Modify Request message.
ldap-modifyDN-request (CTS)	Matches the entire Modify-DN Request message.
ldap-modifyDN-request-entry (CTS)	Matches the DN of the entry in a Modify-DN Request message.
ldap-modifyDN-request-newRDN (CTS)	Matches the new DN that replaces the old DN in a Modify-DN Request message.
ldap-modifyDN-request-newsuperior (CTS)	Matches the new DN that becomes the parent of the existing DN entry in a Modify-DN Request message.
ldap-result (STC)	Matches the entire Result message, including the 1-byte response type.
ldap-result-errorMessage (STC)	Matches the error message in the result.
ldap-result-matchedDN (STC)	Matches the base object in the Result message, including the 1-byte tag.
ldap-result-referral (STC)	Matches each referral URL in the result.
ldap-search-request (CTS)	Matches the entire LDAP Search Request message.
ldap-search-request-attribute (CTS)	Matches each attribute in a Search Request message.
ldap-search-request-attributelist (CTS)	Matches all the attributes in a Search Request message.
ldap-search-request-baseObject (CTS)	Matches the base object entry against which the search is performed. This includes the 1-byte scope, which can represent baseObject, singleLevel or wholeSubtree.
ldap-search-request-filter (CTS)	Matches the contents of the search filter.
ldap-search-request-sizeLimit (CTS)	Matches the sizeLimit field of the search request.
ldap-search-request-timeLimit (CTS)	Matches the timeLimit field of the search request.

**Table 42: Service Contexts: LDAP (*continued*)**

Context and Direction	Description
ldap-search-resentry (STC)	Matches the entire Search Result message.
ldap-search-resentry-attribute (STC)	Matches each attribute in the search result. The values are NULL delimited, and the type and value list are newline delimited.
ldap-search-resentry-attributetype (STC)	Matches each attribute type in the search result.
ldap-search-resentry-attributevalue (STC)	Matches each attribute value in the search result.
ldap-search-resentry-objectname (STC)	Matches the base object of the search result.
ldap-search-resref (STC)	Matches the entire Search Result Reference message.
ldap-search-resref-referral (STC)	Matches each referral URL in the Search Result Reference message.

**Table 43: Service Contexts: Line**

Context and Direction	Description
line (ANY)	Matches a line extracted from the reassembled, normalized TCP stream data. This context is available for only those protocols that are line based.

**Table 44: Service Contexts: LPR**

Context and Direction	Description
lpr-cfile-command (CTS)	Matches the entire CFILE subcommand line, including the first byte of the subcommand type.
lpr-cfile-name (CTS)	Matches the name of the control filename that is sent as part of the RECEIVE-JOB command.
lpr-command (CTS)	Matches the entire command line, including the first byte of the command code.
lpr-dfile-name (CTS)	Matches the name of the data filename that is sent as part of the RECEIVE-JOB command.

**Table 45: Service Contexts: MGCP**

Context and Direction	Description
mgcp-call-id (ANY)	Matches the MGCP call ID parameter value.
mgcp-command (ANY)	Matches the MGCP command line.

**Table 45: Service Contexts: MGCP (*continued*)**

Context and Direction	Description
mgcp-ep-name (ANY)	Matches the MGCP endpoint name specified in command line or command parameters.
mgcp-parm (ANY)	Matches the MGCP command parameter value.
mgcp-rsp (ANY)	Matches the entire MGCP response line with the return code.
mgcp-rsp-000-line (ANY)	Matches the MGCP 0yz response acknowledgment.
mgcp-rsp-100-line (ANY)	Matches the MGCP 1yz provisional response.
mgcp-rsp-200-line (ANY)	Matches the MGCP 2yz successful completion response.
mgcp-rsp-400-line (ANY)	Matches the MGCP 4yz permanent error response
mgcp-rsp-500-line (ANY)	Matches the MGCP 5yz permanent error response.
mgcp-rsp-800-line (ANY)	Matches the MGCP 8yz package-specific response codes.
mgcp-rsp-bad-rcode (ANY)	Matches any MGCP invalid response code.
mgcp-sdp-line (ANY)	Matches MGCP/SDP contents data line.
mgcp-trans-id (ANY)	Matches the MGCP transaction ID parameter value.

**Table 46: Service Contexts: Modbus**

Context and Direction	Description
modbus-except-rsp (STC)	Matches a Modbus Exception Response.
modbus-request (CTS)	Matches a Modbus Request
modbus-response (STC)	Matches a Modbus Response.
modbus-trailing-data (ANY)	Matches trailing data after the first MODBUS PDU.

**Table 47: Service Contexts: MSN**

Context and Direction	Description
msn-addrbook-url (STC)	Matches the URL for a user's address book.
msn-compose-url (STC)	Matches the URL for composing an e-mail.
msn-display-name (ANY)	Matches the display name of a user.



**Table 47: Service Contexts: MSN (*continued*)**

Context and Direction	Description
msn-get-file (STC)	Matches the name of a file that the client is downloading from a peer.
msn-group-name (ANY)	Matches the name of a group of contacts.
msn-inbox-url (STC)	Matches the URL for a user's Inbox.
msn-ip-port (STC)	Matches the address and port of a switchboard server.
msn-message (ANY)	Matches the instant message text.
msn-message-application (ANY)	Matches the line of an application message (like file transfer).
msn-message-email-notification (STC)	Matches the line sent by the server to notify a client of new or unread e-mail.
msn-message-header (ANY)	Matches the header line of an instant message.
msn-message-profile (STC)	Matches the line containing the profile of a message sender.
msn-passport-url (STC)	Matches login passport URL.
msn-phone-number (ANY)	Matches the user's phone number.
msn-png-chunk (ANY)	Matches contents of PNG chunk in MSN transaction.
msn-profile-url (STC)	Matches the URL of a user's passport profile.
msn-put-file (CTS)	Matches the name of a file that the client is sending to a peer.
msn-sign-in-name (ANY)	Matches the screen name (login name) of a user.
msn-url (STC)	Matches any URL in an MSN session
msn-user-state (ANY)	Matches the user's online state.

**Table 48: Service Contexts: MSRPC**

Context and Direction	Description
msrpc-ifid-str (ANY)	Matches the interface ID string in an MSRPC session.

**Table 49: Service Contexts: MS-SQL**

Context and Direction	Description
mssql-0x12 (CTS)	Matches the content of an MS-SQL type 0x12 request message.

**Table 49: Service Contexts: MS-SQL (*continued*)**

Context and Direction	Description
mssql-cancel (CTS)	Matches the content of an MS-SQL cancel message
mssql-login (CTS)	Matches the content of an MS-SQL login message.
mssql-login-app (CTS)	Matches the name of the application in an MS-SQL Login message.
mssql-login-client (CTS)	Matches the name of the client in an MS-SQL Login message.
mssql-login-database (CTS)	Matches the name of the database in an MS-SQL Login message.
mssql-login-language (CTS)	Matches the name of the language in an MS-SQL Login message.
mssql-login-lib (CTS)	Matches the name of the library in an MS-SQL Login message.
mssql-login-pass (CTS)	Matches the password in an MS-SQL Login message.
mssql-login-server (CTS)	Matches the name of the server in an MS-SQL Login message.
mssql-login-user (CTS)	Matches the name of the user in an MS-SQL Login message.
mssql-query (CTS)	Matches the content of a MS-SQL query message.
mssql-request-other (CTS)	Matches the content of an MS-SQL unknown Request message.
mssql-rpe (CTS)	Matches the content of an MS-SQL RPC message.
mssql-rpc-name (CTS)	Matches the RPC name in an MS-SQL request message.

**Table 50: Service Contexts: MySQL**

Context and Direction	Description
mysql-login-request-caps (CTS)	Matches the MYSQL Login Request Caps Data.
mysql-login-request-caps-pswd (CTS)	Matches the MYSQL Login Request Caps Password.
mysql-login-request-caps-user (CTS)	Matches the MYSQL Login Request Caps Username.
mysql-preamble (ANY)	Matches the 4 first bytes of the packet.
mysql-request-command (CTS)	Matches the MYSQL Request Command.
mysql-response (STC)	Matches the MYSQL Response.
mysql-server-greeting (STC)	Matches the MYSQL Server Greeting Data.

Table 51: Service Contexts: NetBIOS

Context and Direction	Description
nbds-browse-backup-server (ANY)	Matches the name of a backup server in a NetBIOS browse message.
nbds-browse-server-name (ANY)	Matches the name of a server in a NetBIOS browse message.
nbds-destination-name (ANY)	Matches the destination name field in a NetBIOS message.
nbds-mailslot-name (ANY)	Matches the name of a mailslot in the NetBIOS mailslot message.
nbds-source-ip-address (ANY)	Matches the source IP field in the NetBIOS datagram header.
nbds-source-name (ANY)	Matches the source name field in a NetBIOS message.
nbds-source-port (ANY)	Matches the source port fields in the NetBIOS datagram header.
nbname-node-name (ANY)	Matches the node name in the status response message.
nbname-node-status (ANY)	Matches the statistics field of a node status response.
nbname-nsd-ip-address (ANY)	Matches the IP address of a NetBIOS name server specified in a redirect name query response message.
nbname-nsd-name (ANY)	Matches the name of a NetBIOS name server specified in a redirect name query response message.
nbname-resource-address (ANY)	Matches the IP address of a resource from the resource record.
nbname-type-name (ANY)	Matches the type and name in a question or a resource record.

Table 52: Service Contexts: NFS

Context and Direction	Description
nfs-create-name (CTS)	Matches the name of a file or directory in the CREATE procedure.
nfs-dir-entry (STC)	Matches the name of each directory entry returned by the READDIR procedure.
nfs-link-target (CTS)	Matches the name of the hard link in the LINK procedure.
nfs-lookup-name (CTS)	Matches the name of a file or directory in the LOOKUP procedure.
nfs-mkdir-name (CTS)	Matches the name of a directory in the MKDIR procedure.
nfs-mknod-name (CTS)	Matches the name of the special file in the MKNOD procedure.
nfs-readlink-name (STC)	Matches the name returned by the READLINK procedure
nfs-remove-name (CTS)	Matches the name of a file in the REMOVE procedure.

**Table 52: Service Contexts: NFS (*continued*)**

Context and Direction	Description
nfs-rename-from (CTS)	Matches the source file or directory name in the RENAME procedure.
nfs-rename-to (CTS)	Matches the destination file or directory name in the RENAME procedure.
nfs-rmdir-name (CTS)	Matches the name of a directory in the RMDIR procedure.
nfs-symlink-source (CTS)	Matches the source of the symbolic link in the SYMLINK procedure.
nfs-symlink-target (CTS)	Matches the target of the symbolic link in the SYMLINK procedure.

**Table 53: Service Contexts: NNTP**

Context and Direction	Description
nntp-banner (STC)	Matches the NNTP banner.
nntp-body (ANY)	Matches each line of an NNTP message body.
nntp-command-line (CTS)	Matches the entire NNTP command line.
nntp-header (ANY)	Matches any header in an NNTP session.
nntp-ihave-msgid (CTS)	Matches the message ID that appears in the IHAVE command of an NNTP session.
nntp-mode (CTS)	Matches the NNTP mode.
nntp-msgid (ANY)	Matches the message ID that appears in various commands of an NNTP session.
nntp-newsgroup (ANY)	Matches the name of news groups in an NNTP session.

**Table 54: Service Contexts: Normalized Stream**

Context and Direction	Description
normalized-stream (ANY)	Normalized Stream for services Telnet, IMAP, NFS, RPC, and Ruser only.
normalized-stream1k (ANY)	Matches the first 1024 bytes of reassembled, normalized TCP stream data.
normalized-stream256 (ANY)	Matches the first 256 bytes of reassembled, normalized TCP stream data.
normalized-stream8k (ANY)	Matches the first 8192 bytes of reassembled, normalized TCP stream data.

Table 55: Service Contexts: NTP

Context and Direction	Description
ntp-ctrl-data-opt (ANY)	Matches the data field in an NTP control message.
ntp-ctrl-opcode-response-var (ANY)	Matches each of the name and value pairs found in the NTP control message data field. The context includes a 1-byte NTP control message opcode and a 1-byte NTP response type.

Table 56: Service Contexts: Packet

Context and Direction	Description
packet (ANY)	Matches any packet in a session.

Table 57: Service Contexts: POP3

Context and Direction	Description
pop3-apop (CTS)	Matches the arguments of the APOP command in a POP3 session.
pop3-auth (CTS)	Matches the arguments of the AUTH command in a POP3 session.
pop3-command (CTS)	Matches each of the POP3 command names in a POP3 session.
pop3-command-line (CTS)	Matches each command line in a POP3 session.
pop3-data-line (STC)	Matches lines in the e-mail body of an POP3 transaction.
pop3-data-text-html (STC)	Matches lines in a text/html MIME attachment in the body of an POP3 transaction.
pop3-data-text-plain (STC)	Matches lines in a text/plain MIME attachment in the body of an POP3 transaction.
pop3-dele (CTS)	Matches the arguments of the DELE command in a POP3 session.
pop3-header-comment (STC)	Matches the Comment: header of an e-mail in a POP3 transaction.
pop3-header-from (STC)	Matches the From: header of an e-mail in a POP3 transaction.
pop3-header-line (STC)	Matches each header line of an e-mail in POP3 transaction.
pop3-header-reply-to (STC)	Matches the Reply-To: header of an e-mail in a POP3 transaction.
pop3-header-sender (STC)	Matches the Sender: header of an e-mail in a POP3 transaction.
pop3-header-subject (STC)	Matches the Subject: header of an e-mail in a POP3 transaction.
pop3-header-to (STC)	Matches the To: header of an e-mail in a POP3 transaction.

**Table 57: Service Contexts: POP3 (*continued*)**

Context and Direction	Description
pop3-header-x-field (STC)	Matches each extended header (that start with X-) of an e-mail in a POP3 transaction.
pop3-header-x-mailer (STC)	Matches the X-Mailer: header of an e-mail in a POP3 transaction.
pop3-list (CTS)	Matches the arguments of the LIST command in a POP3 session.
pop3-mime-content-data (STC)	Matches the first 64 bytes of the base-64 decoded MIME attachment data in a POP3 session.
pop3-mime-content-filename (STC)	Matches the content filename of a MIME attachment in a POP3 session.
pop3-mime-content-name (STC)	Matches the content name of a MIME attachment in a POP3 session.
pop3-retr (CTS)	Matches the arguments of the RETR command in a POP3 session.
pop3-top (CTS)	Matches the arguments of the TOP command in a POP3 session.
pop3-uidl (CTS)	Matches the arguments of the UIDL command in a POP3 session.
pop3-user (CTS)	Matches the user name of a POP3 session.
pop3-xtnd (CTS)	Matches the arguments of the XTND command in a POP3 session.

**Table 58: Service Contexts: RADIUS**

Context and Direction	Description
radius-access-accept (STC)	Matches the attribute fields of a RADIUS Access-Accept message.
radius-access-challenge (STC)	Matches the attribute fields of a RADIUS Access-Challenge message.
radius-access-reject (STC)	Matches the attribute fields of a RADIUS Access-Reject message.
radius-access-request (CTS)	Matches the attribute fields of a RADIUS Access-Request message.
radius-acct-request (CTS)	Matches the attribute fields of a RADIUS Accounting-Request message.
radius-acct-response (STC)	Matches the attribute fields of a RADIUS Accounting-Response message.
radius-attr-acct-multi-session-id (CTS)	Matches the value of an Account-Multi-Session-Id attribute.
radius-attr-acct-session-id (CTS)	Matches the value of an Account-Session-Id attribute.
radius-attr-acct-tunnel-connection (CTS)	Matches the value of an Account-Tunnel-Connection attribute.

Table 58: Service Contexts: RADIUS (*continued*)

Context and Direction	Description
radius-attr-arap-features (STC)	Matches the value of an ARAP-Features attribute.
radius-attr-arap-password (CTS)	Matches the value of an ARAP-Password attribute.
radius-attr-arap-security-data (ANY)	Matches the value of an ARAP-Security-Data attribute.
radius-attr-callback-number (ANY)	Matches the value of a Callback-Number attribute.
radius-attr-called-station-id (CTS)	Matches the value of a Caller-Station-Id attribute.
radius-attr-calling-station-id (CTS)	Matches the value of a Calling-Station-Id attribute.
radius-attr-chap-challenge (CTS)	Matches the value of a Chap-Challenge attribute.
radius-attr-chap-password (CTS)	Matches the value of a Chap-Password attribute.
radius-attr-configuration-token (STC)	Matches the value of a Configuration-Token attribute.
radius-attr-connect-info (CTS)	Matches the value of a Connect-Info attribute.
radius-attr-eap-message (ANY)	Matches the value of an EAP-Message attribute.
radius-attr-filter-id (ANY)	Matches the value of a Filter-Id attribute.
radius-attr-framed-appletalk-zone (ANY)	Matches the value of a Framed-Appletalk-Zone attribute.
radius-attr-framed-pool (STC)	Matches the value of a Framed-Pool attribute.
radius-attr-framed-route (ANY)	Matches the value of a Framed-Route attribute.
radius-attr-login-lat-group (ANY)	Matches the value of a Login-LAT-Group attribute.
radius-attr-login-lat-node (ANY)	Matches the value of a Login-LAT-Node attribute.
radius-attr-login-lat-port (ANY)	Matches the value of a Login-LAT-Port attribute.
radius-attr-login-lat-service (ANY)	Matches the value of a Login-LAT-Service attribute.
radius-attr-message-authenticator (ANY)	Matches the value of a Message-Authenticator attribute.
radius-attr-nas-identifier (CTS)	Matches the value of a NAS-Identifier attribute.
radius-attr-nas-port-id (CTS)	Matches the value of a NAS-Port-Id attribute.
radius-attr-proxy-state (ANY)	Matches the value of a Proxy-State attribute.

**Table 58: Service Contexts: RADIUS (continued)**

Context and Direction	Description
radius-attr-reply-message (STC)	Matches the value of a Reply-Message attribute.
radius-attr-state (ANY)	Matches the value of a State attribute
radius-attr-tunnel-assignment-id (ANY)	Matches the value of a Tunnel-Assignment-Id attribute.
radius-attr-tunnel-client-auth-id (ANY)	Matches the value of a Tunnel-Client-Auth-Id attribute
radius-attr-tunnel-client-endpoint (ANY)	Matches the value of a Tunnel-Client-Endpoint attribute.
radius-attr-tunnel-password (STC)	Matches the value of a Tunnel-Password attribute.
radius-attr-tunnel-private-group-id (ANY)	Matches the value of a Tunnel-Private-Group-Id attribute.
radius-attr-tunnel-server-auth-id (ANY)	Matches the value of a Tunnel-Server-Auth-Id attribute.
radius-attr-tunnel-server-endpoint (ANY)	Matches the value of a Tunnel-Server-Endpoint attribute.
radius-attr-user-name (ANY)	Matches the value of a User-Name attribute.
radius-attr-user-password (CTS)	Matches the value of a User-Password attribute.
radius-attr-vendor-specific (ANY)	Matches the value of a Vendor-Specific attribute.
radius-attribute (ANY)	Matches any RADIUS attribute, including the type, length and value.

**Table 59: Service Contexts: REXEC**

Context and Direction	Description
rexec-remote-command (CTS)	Matches the remote command in an REXEC session.
rexec-remote-user (CTS)	Matches the remote username in an REXEC session.

**Table 60: Service Contexts: RLOGIN**

Context and Direction	Description
rlogin-local-user (CTS)	Matches the local username in an RLOGIN session.
rlogin-remote-user (CTS)	Matches the remote username in an RLOGIN session.



**Table 61: Service Contexts: RSH**

Context and Direction	Description
rsh-local-user (CTS)	Matches the local username in an RSH session.
rsh-remote-command (CTS)	Matches the remote command in an RSH session.
rsh-remote-user (CTS)	Matches the remote username in an RSH session.

**Table 62: Service Contexts: RUSERS**

Context and Direction	Description
rusers-device (STC)	Matches the name of the device in an RUSERS session.
rusers-host (STC)	Matches the name of the host in an RUSERS session.
rusers-user (STC)	Matches the name of the user in an RUSERS session.

**Table 63: Service Contexts: SIP**

Context and Direction	Description
sip-bad-header (ANY)	Matches SIP headers with bad syntax.
sip-command-state (ANY)	Matches the state of the SIP connection.
sip-content-any (ANY)	Matches SIP contents portion of packet data.
sip-content-sdp (ANY)	Matches SIP/SDP content data.
sip-display-name (ANY)	Matches the display name of URL in related headers.
sip-header-any (ANY)	Matches SIP headers with no designated context.
sip-header-callid (ANY)	Matches the SIP <Call-ID> header.
sip-header-from (ANY)	Matches the SIP <From> header.
sip-header-maxforwards (CTS)	Matches the SIP <Max-Forwards> header.
sip-header-to (ANY)	Matches SIP <To> header.
sip-header-value-len (ANY)	Artificially created context for putting thresholds on a header value.
sip-headr-via (ANY)	Matches the SIP <Via> header.
sip-parameter (ANY)	Matches parsed parameters in the headers.
sip-parameter-bad (ANY)	Matches parsed invalid parameters in the headers.

**Table 63: Service Contexts: SIP (*continued*)**

Context and Direction	Description
sip-reply (STC)	Matches any SIP reply line with the return code.
sip-reply-100-line (STC)	Matches the SIP 1yz Positive Preliminary reply.
sip-reply-200-line (STC)	Matches the SIP 2yz Positive Completion reply.
sip-reply-300-line (STC)	Matches the SIP 3yz Postive Intermediate reply.
sip-reply-400-line (STC)	Matches the SIP 4yz Transient Negative Completion reply.
sip-reply-500-line (STC)	Matches the SIP 5yz Permanent Negative Completion reply.
sip-reply-600-line (STC)	Matches the SIP 6yz Failure Completion reply.
sip-reply-bad-rcode (STC)	Matches any SIP invalid response code.
sip-request (CTS)	Matches the SIP request command line.
sip-request-unknown (CTS)	Matches the SIP request with unknown command.
sip-sdp-line (ANY)	Matches the SIP/SDP contents data line.
sip-unknown-data (ANY)	Matches SIP unknown data.
sip-unknown-header (ANY)	Matches a SIP unknown header.
sip-uri-host (ANY)	Matches the host-name/IP-address of URI in related headers.
sip-uri-parameter (ANY)	Matches the parameter of URI in related headers.

**Table 64: Service Contexts: SMB**

Context and Direction	Description
smb-account-name (ANY)	Matches the SMB account name in the SESSION_SETUP_ANDX request of an SMB session.
smb-atsvc-request (CTS)	Matches any AT Service requests sent as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-atsvc-response (STC)	Matches any AT Service responses received as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-browser-request (CTS)	Matches any Browser requests sent as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.

Table 64: Service Contexts: SMB (*continued*)

Context and Direction	Description
smb-browser-response (STC)	Matches any Browser responses received as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-called-name (ANY)	Matches the NetBIOS name of the initiator of an SMB session.
smb-calling-name (ANY)	Matches the NetBIOS name of the receiver of an SMB session.
smb-connect-path (CTS)	Matches the connect path in the TREE_CONNECT_ANDX request of an SMB session.
smb-connect-service (CTS)	Matches the connect service in the TREE_CONNECT_ANDX request of an SMB session.
smb-copy-filename (CTS)	Matches the filename in the COPY request of an SMB session.
smb-data (ANY)	Matches any SMB data portion.
smb-dce-rpc (ANY)	Matches any DCE/RPC message sent over the SMB Transport Layer.
smb-dce-rpc-alter-ctx (CTS)	Matches any DCE/RPC alter-context message sent over the SMB Transport Layer.
smb-dce-rpc-alter-ctx-reply (STC)	Matches any DCE/RPC alter-context-reply message sent over the SMB Transport Layer.
smb-dce-rpc-bind (CTS)	Matches any DCE/RPC bind message sent over the SMB Transport Layer.
smb-dce-rpc-bind-ack (STC)	Matches any DCE/RPC bind-ack message sent over the SMB Transport Layer.
smb-dce-rpc-bind-nack (STC)	Matches any DCE/RPC bind-nack message sent over the SMB Transport Layer.
smb-dce-rpc-cancel (CTS)	Matches any DCE/RPC cancel message sent over the SMB Transport Layer.
smb-dce-rpc-orphaned (CTS)	Matches any DCE/RPC orphaned message sent over the SMB Transport Layer.
smb-dce-rpc-request (CTS)	Matches any DCE/RPC request message sent over the SMB Transport Layer.
smb-dce-rpc-request-obj-uuid (CTS)	Matches object UUID of any DCE/RPC request message.
smb-dce-rpc-response (STC)	Matches any DCE/RPC response message sent over the SMB Transport Layer.
smb-dce-rpc-shutdown (STC)	Matches any DCE/RPC shutdown message sent over the SMB Transport Layer.
smb-delete-filename (CTS)	Matches the filename in the DELETE request of an SMB session.

Table 64: Service Contexts: SMB (*continued*)

Context and Direction	Description
smb-dialect (CTS)	Matches each SMB dialect string in the NEGOTIATE request of an SMB session.
smb-lanman-request (CTS)	Matches any LANMAN requests sent as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-lanman-response (STC)	Matches any LANMAN responses received as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-lsarpc-request (CTS)	Matches any Local Security Authority requests sent as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-lsarpc-response (STC)	Matches any Local Security Authority responses received as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-move-filename (CTS)	Matches the filename in the MOVE request of an SMB session.
smb-msgsvc-request (CTS)	Matches any Messenger Service requests sent as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-msgsvc-response (STC)	Matches any Messenger Service responses received as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-native-lanman (ANY)	Matches the native LANMAN in the SESSION_SETUP_ANDX request of an SMB session.
smb-native-os (ANY)	Matches the native OS in the SESSION_SETUP_ANDX request of an SMB session.
smb-netlogon-request (CTS)	Matches any Netlogon requests sent as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-netlogon-response (STC)	Matches any Netlogon responses received as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-open-filename (CTS)	Matches the filename in the NT_CREATE_ANDX and OPEN_ANDX requests of an SMB session.
smb-pipe-request (CTS)	Matches any generic named pipe requests over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-pipe-response (STC)	Matches any generic named pipe responses over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.

Table 64: Service Contexts: SMB (*continued*)

Context and Direction	Description
smb-primary-domain (ANY)	Matches the SMB primary domain name in the SESSION_SETUP_ANDX request of an SMB session.
smb-rename-filename (CTS)	Matches the filename in the RENAME request of an SMB session.
smb-samr-request (CTS)	Matches any Security Account Manager requests sent as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-samr-response (STC)	Matches any Security Account Manager responses received as named pipe transactions over the SMB Transport Layer. The first 2 bytes of this context contains the opcode of the function.
smb-spoolss-request (CTS)	Matches any Spool Subsystem requests sent as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-spoolss-response (STC)	Matches any Spool Subsystem responses received as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-srvsvc-request (CTS)	Matches any Server Service requests sent as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-srvsvc-response (STC)	Matches any Server Service responses received as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-svcctl-request (CTS)	Matches any Service Control Manager requests sent as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-svcctl-response (STC)	Matches any Service Control Manager responses received as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-trans2-create-directory (CTS)	Matches any SMB Transaction2 CREATE-DIRECTORY request.
smb-trans2-request (CTS)	Matches any SMB Transaction2 request.
smb-trans2-response (STC)	Matches any SMB Transaction2 response.
smb-trans2-session-setup (CTS)	Matches any SMB Transaction2 SESSION-SETUP request.
smb-trans2-set-file-info (CTS)	Matches any SMB Transaction2 SET-FILE-INFORMATION request.
smb-trans2-set-path-info (CTS)	Matches any SMB Transaction2 SET-PATH-INFORMATION request.

**Table 64: Service Contexts: SMB (*continued*)**

Context and Direction	Description
smb-winreg-request (CTS)	Matches any Windows Remote Registry requests sent as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-winreg-response (STC)	Matches any Windows Remote Registry responses received as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-wkssvc-request (CTS)	Matches any Workstation Service requests sent as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.
smb-wkssvc-response (STC)	Matches any Workstation Service responses received as named pipe transactions over the SMB Transport Layer. The first two bytes of this context contains the opcode of the function.

**Table 65: Service Contexts: SMTP**

Context and Direction	Description
smtp-banner (STC)	Matches the banner returned by the server at the start of an SMTP transaction.
smtp-command-line (CTS)	Matches any SMTP command line.
smtp-data-line (CTS)	Matches lines in the e-mail body of an SMTP transaction.
smtp-data-text-html (CTS)	Matches lines in a text/html MIME attachment in the body of an SMTP transaction.
smtp-data-text-plain (CTS)	Matches lines in a text/plain MIME attachment in the body of an SMTP transaction.
smtp-from (CTS)	Matches the contents of the MAIL, SAML, SEND, and SOML commands.
smtp-header (CTS)	Matches any unfolded header in the SMTP data.
smtp-header-comment (CTS)	Matches the Comment: header in the SMTP data.
smtp-header-from (CTS)	Matches the From: header in the SMTP data.
smtp-header-line (CTS)	Matches any header lines in the SMTP data.
smtp-header-reply-to (CTS)	Matches the Reply-To: header in the SMTP data.
smtp-header-sender (CTS)	Matches the Sender: header in the SMTP data.
smtp-header-subject (CTS)	Matches the Subject: header in the SMTP data.
smtp-header-to (CTS)	Matches the To: header in the SMTP data.

Table 65: Service Contexts: SMTP (*continued*)

Context and Direction	Description
smtp-header-x-field (CTS)	Matches all extended headers that start with X- in the SMTP data.
smtp-header-x-mailer (CTS)	Matches the X-Mailer: header in the SMTP data.
smtp-mime-content-data (CTS)	Matches the first 64 bytes of the base-64 decoded MIME attachment data in an SMTP session.
smtp-mime-content-filename (CTS)	Matches the content filename of a MIME attachment in an SMTP session.
smtp-mime-content-name (CTS)	Matches the content name of a MIME attachment in an SMTP session.
smtp-rcpt (CTS)	Matches the contents of the RCPT command in an SMTP transaction.
smtp-reply-100-line (STC)	Matches the SMTP 1yz Positive Preliminary reply.
smtp-reply-200-line (STC)	Matches the SMTP 2yz Positive Completion reply.
smtp-reply-300-line (STC)	Matches the SMTP 3yz Positive Intermediate reply.
smtp-reply-400-line (STC)	Matches the SMTP 4yz Transient Negative Completion reply.
smtp-reply-500-line (STC)	Matches the SMTP 5yz Permanent Negative Completion reply.
smtp-reply-line (STC)	Matches the SMTP reply line.

Table 66: Service Contexts: SNMP

Context and Direction	Description
snmp-community (ANY)	Matches the community name in any SNMP request or response.
snmp-get-bulk-oid (CTS)	Matches the binary OID in any SNMP Get-Bulk request.
snmp-get-bulk-oid-parsed (CTS)	Matches the human-readable OID in any SNMP Get-Bulk request.
snmp-get-next-oid (CTS)	Matches the binary OID in any SNMP Get-Next request.
snmp-get-next-oid-parsed (CTS)	Matches the human-readable OID in any SNMP Get-Next request.
snmp-get-oid (CTS)	Matches the binary OID in any SNMP Get request.
snmp-get-oid-parsed (CTS)	Matches the human-readable OID in any SNMP Get request.
snmp-oid (ANY)	Matches the binary OID in any SNMP request or response.
snmp-oid-parsed (ANY)	Matches the human-readable OID in any SNMP request or response.

**Table 66: Service Contexts: SNMP (*continued*)**

Context and Direction	Description
snmp-set-oid (CTS)	Matches the binary OID in any SNMP Set request.
snmp-set-oid-parsed (CTS)	Matches the human-readable OID in any SNMP Set request.
snmptrap-community (CTS)	Matches the community name in any SNMPTRAP message.
snmptrap-eid (CTS)	Matches the binary EID (Enterprise-ID) in any SNMPTRAP message.
snmptrap-eid-parsed (CTS)	Matches the human-readable EID (Enterprise-ID) in any SNMPTRAP message.
snmptrap-inform-oid (CTS)	Matches the binary OID in any SNMPTRAP Inform message.
snmptrap-inform-oid-parsed (CTS)	Matches the human-readable OID in any SNMPTRAP Inform message.
snmptrap-oid (CTS)	Matches the binary OID in any SNMPTRAP message.
snmptrap-oid-parsed (CTS)	Matches the human-readable OID in any SNMPTRAP message.
snmptrap-v2-oid (CTS)	Matches the binary OID in any SNMPTRAP v2 message.
snmptrap-v2-oid-parsed (CTS)	Matches the human-readable OID in any SNMPTRAP v2 message.

**Table 67: Service Contexts: SSH**

ssh-header (ANY)	Matches the header at the start of an SSH session.
------------------	--

**Table 68: Service Contexts: SSL**

Context and Direction	Description
ssl-cert-common-name (ANY)	Matches the common name attribute of the SSL certificate.
ssl-cert-organization-name (ANY)	Matches the organization name in the SSL certificate.
ssl-cert-organizational-unit-name (ANY)	Matches the organizational unit name in the SSL certificate.
ssl-certificate (ANY)	Matches the entire SSL certificate content.
ssl-client-hello (CTS)	Matches SSL client hello message content.
ssl-client-key-exchange (CTS)	Matches SSL client key exchange message content.
ssl-client-version (CTS)	Matches the client SSL version.
ssl-selected-cipher-suite (STC)	Matches the selected cipher suite in the server hello message.



**Table 68: Service Contexts: SSL (*continued*)**

Context and Direction	Description
ssl-server-hello (STC)	Matches SSL server hello message content.
ssl-server-key-exchange (STC)	Matches SSL server key exchange message content.
ssl-server-version (STC)	Matches the SSL server version.

**Table 69: Service Contexts: Stream**

Context and Direction	Description
stream (ANY)	Matches the reassembled, normalized TCP stream data.
stream1k (ANY)	Matches the first 1024 bytes of reassembled TCP stream data.
stream256 (ANY)	Matches the first 256 bytes of reassembled, normalized TCP stream data.
stream8k (ANY)	Matches the first 8192 bytes of reassembled TCP stream data.

**Table 70: Service Contexts: Telnet**

Context and Direction	Description
telnet-option (ANY)	Matches each of the telnet options in a Telnet session.
telnet-subnegotiation (ANY)	Matches each of the telnet subnegotiation options in a Telnet session.
telnet-user (CTS)	Matches the Telnet user name.

**Table 71: Service Contexts: TFTP**

Context and Direction	Description
tftp-filename (CTS)	Matches any filename in a TFTP session.
tftp-get-filename (CTS)	Matches the get filename in a TFTP session.
tftp-put-filename (CTS)	Matches the put filename in a TFTP session.

**Table 72: Service Contexts: TNS**

Context and Direction	Description
tns-accept-section (STC)	Matches the Accept Section Data in a TNS session.
tns-connect-addr-dev (CTS)	Matches the Connect Address-Dev in a TNS session.
tns-connect-addr-host (CTS)	Matches the Connect Address-Host in a TNS session.

**Table 72: Service Contexts: TNS (*continued*)**

Context and Direction	Description
tns-connect-addr-key (CTS)	Matches the Connect Address-Key in a TNS session.
tns-connect-addr-port (CTS)	Matches the Connect Address-Port in a TNS session.
tns-connect-addr-proto (CTS)	Matches the Connect Address-Protocol in an TNS session.
tns-connect-cid-host (CTS)	Matches the Connect Data CID Host in a TNS session.
tns-connect-cid-user (CTS)	Matches the Connect Data CID User in a TNS session.
tns-connect-data-cid-prog (CTS)	Matches the Connect Data CID Program in a TNS session.
tns-connect-data-sid (CTS)	Matches the Connect Data SID in a TNS session.
tns-connect-data-svname (CTS)	Matches the Connect Data Service Name in an TNS session.
tns-connect-section (CTS)	Matches the Connect Section Data in a TNS session.
tns-data-section (ANY)	Matches the Data Section Data in a TNS session.
tns-message-body (ANY)	Matches any Message Body in a TNS session.
tns-message-type (ANY)	Matches the Message Type in a TNS session.
tns-preamble (ANY)	Matches the first 8 bytes of a TNS message.
tns-redirect-section (STC)	Matches the Redirect Section in a TNS session.

**Table 73: Service Contexts: VNC**

Context and Direction	Description
vnc-client-version (CTS)	Matches the version number of the VNC protocol sent by the client.
vnc-reason (STC)	Matches the connection fail reason reported by the VNC server.
vnc-server-version (STC)	Matches the version number of the VNC protocol sent by the server.

**Table 74: Service Contexts: YMSG**

Context and Direction	Description
ymsg-alias (ANY)	Matches the alternate name associated with the main username.
ymsg-buddy-name (ANY)	Matches the name of a user that appears on the friends list.
ymsg-chatroom-chatter (ANY)	Matches the name of a user participating in a chat session

Table 74: Service Contexts: YMSG (*continued*)

Context and Direction	Description
ymsg-chatroom-invitee (ANY)	Matches the name of the user who is being invited to join a chatroom.
ymsg-chatroom-message (ANY)	Matches the messages exchanged in a chatroom.
ymsg-chatroom-name (ANY)	Matches the name of a chatroom in a YMSG session.
ymsg-conf-host (ANY)	Matches the name of the user who is hosting the conference.
ymsg-conf-invitee (ANY)	Matches the name of a user who is invited to a conference.
ymsg-conf-join-msg (ANY)	Matches the content of a message sent as part of a conference invitation.
ymsg-conf-name (ANY)	Matches the name of a conference session.
ymsg-config-url (STC)	Matches the URL at which the user can configure the password after the account is disabled.
ymsg-contact-name (ANY)	Matches the contact name in a friends list or invitation.
ymsg-group-name (ANY)	Matches the name of a group used to categorize friends.
ymsg-header (ANY)	Matches data in the protocol header.
ymsg-ignored-user (ANY)	Matches the name of the user being added to, or appearing on, the ignored users list.
ymsg-mail-sender (STC)	Matches the name of the user sending an e-mail message.
ymsg-mail-sender-address (STC)	Matches the e-mail address of sender.
ymsg-mail-subject (STC)	Matches the e-mail subject.
ymsg-main-identity (ANY)	Matches the main identity name of the user.
ymsg-message (ANY)	Matches the instant message that is sent from one client to another.
ymsg-message-server-filename-url (STC)	Matches the message with the name of the file on the client from which the server can download and transfer to peers.
ymsg-nickname (ANY)	Matches the nickname of a user.
ymsg-p2p-get-filename (STC)	Matches the name of the file on the peer from which the file can be downloaded.
ymsg-p2p-get-filename-url (STC)	Matches the location of a file on the peer from which the file can be downloaded.
ymsg-p2p-put-filename (CTS)	Matches the name of the file on the client that other peers can download.

**Table 74: Service Contexts: YMSG (*continued*)**

Context and Direction	Description
ymsg-p2p-put-filename-url (CTS)	Matches the location of a file on the client from which other peers can download.
ymsg-recipient (ANY)	Matches the identity of the recipient of a message or file.
ymsg-sender (ANY)	Matches the identity of a sender of a message or file.
ymsg-server-get-filename-url (STC)	Matches the location of a file on the client from which the server can download and transfer to peers.
ymsg-system-message (STC)	Matches the content of a message sent from the server to the client.
ymsg-user-name (ANY)	Matches the identity of the login user or one of the user's alias.

## Reference: Nonprintable and Printable ASCII Characters

The following tables provide details on ASCII representation of nonprintable and printable characters.

**Table 75: ASCII Reference: Nonprintable Characters**

Dec	Hex	Oct	Char	Comment
0	0	000	NUL	Null
1	1	001	SOH	Start of Heading
2	2	002	STX	Start of Text
3	3	003	ETX	End of Text
4	4	004	EOT	End of Transmission
5	5	005	ENQ	Enquiry
6	6	006	ACK	Acknowledge
7	7	007	BEL	Bell
8	8	010	BS	Backspace
9	9	011	TAB	Horizontal Tab
10	A	012	LF	Line Feed
11	B	013	VT	Vertical Tab

Table 75: ASCII Reference: Nonprintable Characters (*continued*)

Dec	Hex	Oct	Char	Comment
12	C	014	FF	Form Feed
13	D	015	CR	Carriage Return
14	E	016	SO	Shift Out
15	F	017	SI	Shift In
16	10	020	DLE	Data Link Escape
17	11	021	DC1	Device Control 1
18	12	022	DC2	Device Control 2
19	13	023	DC3	Device Control 3
20	14	024	DC4	Device Control 4
21	15	025	NAK	Negative Acknowledgement
22	16	026	SYN	Synchronous Idle
23	17	027	ETB	End of Transmission Block
24	18	030	CAN	Cancel
25	19	031	EM	End of Medium
26	1A	032	SUB	Substitute
27	1B	033	ESC	Escape
28	1C	034	FS	File Separator
29	1D	035	GS	Group Separator
30	1E	036	RS	Record Separator
31	1F	037	US	Unit Separator

Table 76: ASCII Reference: Printable Characters

Dec	Hex	Oct	Char
32	20	040	Space
33	21	041	!
34	22	042	
35	23	043	#
36	24	044	\$
37	25	045	%
38	26	046	&
39	27	047	
40	28	050	(
41	29	051	)
42	2A	052	*
43	2B	053	+
44	2C	054	,
45	2D	055	-
46	2E	056	.
47	2F	057	/
48	30	060	0
49	31	061	1
50	32	062	2
51	33	063	3
52	34	064	4
53	35	065	5
54	36	066	6
55	37	067	7

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
56	38	070	8
57	39	071	9
58	3A	072	:
59	3B	073	;
60	3C	074	<
61	3D	075	=
62	3E	076	>
63	3F	077	?
64	40	100	@
65	41	101	A
66	42	102	B
67	43	103	C
68	44	104	D
69	45	105	E
70	46	106	F
71	47	107	G
72	48	110	H
73	49	111	I
74	4A	112	J
75	4B	113	K
76	4C	114	L
77	4D	115	M
78	4E	116	N
79	4F	117	O

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
80	50	120	P
81	51	121	Q
82	52	122	R
83	53	123	S
'84	54	124	T
85	55	125	U
86	56	126	V
87	57	127	W
88	58	130	X
89	59	131	Y
90	5A	132	Z
91	5B	133	[
92	5C	134	\
93	5D	135	]
94	5E	136	^
95	5F	137	_
96	60	140	`
97	61	141	a
98	62	142	b
99	63	143	c
100	64	144	d
101	65	145	e
102	66	146	f
103	67	147	g



Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
104	68	150	h
105	69	151	i
106	6A	152	j
107	6B	153	k
108	6C	154	l
109	6D	155	m
110	6E	156	n
111	6F	157	o
112	70	160	p
113	71	161	q
114	72	162	r
115	73	163	s
116	74	164	t
117	75	165	u
118	76	166	v
119	77	167	w
120	78	170	x
121	79	171	y
122	7A	172	z
123	7B	173	{
124	7C	174	
125	7D	175	}
126	7E	176	~
127	7F	177	DEL

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
128	80	200	Ç
129	81	201	Ü
130	82	202	É
131	83	203	Â
132	84	204	Ä
133	85	205	À
134	86	206	Å
135	87	207	ç
136	88	210	ê
137	89	211	ë
138	8A	212	è
139	8B	213	ï
140	8C	214	î
141	8D	215	ì
142	8E	216	Ä
143	8F	217	Å
144	90	220	É
145	91	221	æ
146	92	222	Æ
147	93	223	ô
148	94	224	ö
149	95	225	ò
150	96	226	ó
151	97	227	ù

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
152	98	230	ÿ
153	99	231	Ö
154	9A	232	Ü
155	9B	233	ƒ
156	9C	234	£
157	9D	235	¥
158	9E	236	Þ
159	9F	237	ƒ
160	A0	240	á
161	A1	241	í
162	A2	242	ó
163	A3	243	ú
164	A4	244	ñ
165	A5	245	Ñ
166	A6	246	ä
167	A7	247	ø
168	A8	250	¿
169	A9	251	⌘
170	AA	252	
171	AB	253	½
172	AC	254	¼
173	AD	255	ì
174	AE	256	"
175	AF	257	"

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
176	B0	260	
177	B1	262	
178	B2	262	
179	B3	263	
180	B4	264	
181	B5	265	
182	B6	266	
183	B7	267	+
184	B8	270	+
185	B9	271	
186	BA	272	
187	BB	273	+
188	BC	274	+
189	BD	275	+
190	BE	276	+
191	BF	277	+
192	C0	300	+
193	C1	301	-
194	C2	302	-
195	C3	303	+
196	C4	304	-
197	C5	305	+
198	C6	306	
199	C7	307	

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
200	C8	310	+
201	C9	311	+
202	CA	312	-
203	CB	313	-
204	CC	314	
205	CD	315	-
206	CE	316	+
207	CF	317	-
208	D0	320	-
209	D1	321	-
210	D2	322	-
211	D3	323	+
212	D4	324	+
213	D5	325	+
214	D6	326	+
215	D7	327	+
216	D8	330	+
217	D9	331	+
218	DA	332	+
219	DB	333	
220	DC	334	_
221	DD	335	
222	DE	336	
223	DF	337	-

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
224	E0	340	a
225	E1	341	ß
226	E2	342	G
227	E3	343	p
228	E4	344	S
229	E5	345	s
230	E6	346	µ
231	E7	347	t
232	E8	350	F
233	E9	351	T
234	EA	352	O
235	EB	353	d
236	EC	354	8
237	ED	355	f
238	EE	356	e
239	EF	357	n
240	F0	360	=
241	F1	361	ffl
242	F2	362	=
243	F3	363	=
244	F4	364	(
245	F5	365	)
246	F6	366	÷
247	F7	367	~

Table 76: ASCII Reference: Printable Characters (*continued*)

Dec	Hex	Oct	Char
248	F8	370	°
249	F9	371	
250	FA	372	
251	FB	373	˘
252	FC	374	˙
253	FD	375	˚
254	FE	376	¸
255	FF	377	





## CHAPTER 6

# Index

- Index on page 121



# Index

## A

ASCII printable and nonprintable characters.....	106
attack objects	
creating compound.....	29
creating signature.....	9
attack signature properties.....	8

## B

binary bitmask patterns.....	35
bitmask patterns for binary protocols.....	35
Boolean expressions in compound attack	
objects.....	32
brute force attack.....	43
BugTraq Web site.....	2

## C

CDE vulnerability.....	49
CERT Web site.....	2
Common Vulnerabilities and Exposures Web	
site.....	2
compound attack objects	
Boolean expressions.....	32
creating.....	29, 30
example.....	55
protocol anomalies.....	32
compound attack objects, creating.....	29, 30
context check constraint.....	18, 46
context, specifying.....	22
cross-site scripting vulnerability.....	47
customer support.....	xvi
contacting JTAC.....	xvi

## D

denial-of-service attack.....	47
DFA expressions	
ASCII character representation.....	106
examples.....	35, 37, 43, 48, 52, 55, 58
syntax.....	19

## direction

for known contexts.....	70
specifying.....	22
documentation.....	xv
comments on.....	xvi
dtlogin vulnerability.....	49

## F

flow, specifying.....	22
FrSIRT (French Security Incident Response Team)	
Web site.....	3

## I

ICMP packet header matching.....	23
IP packet header matching.....	23
ISS (Internet Security Systems) Web site.....	3

## K

known vulnerabilities, described.....	2
---------------------------------------	---

## L

lab requirements.....	3
-----------------------	---

## M

manuals	
comments on.....	xvi
Metasploit Web site.....	3

## N

National Vulnerability Database Web site.....	2
negating a pattern.....	37

## P

packet header matching.....	23
Packet Storm Security Web site.....	3
pattern negation.....	37
protocol anomalies in compound attack	
objects.....	32
protocol number reference.....	61

**R**

reference tables	
ASCII characters.....	106
protocol numbers.....	61
service contexts.....	70
service properties.....	67

**S**

same context constraint.....	46
SANS (SysAdmin, Audit, Network, Security Institute) Web site.....	2
scio pcap.....	5
scio pcap.....	5
service binding.....	16
service contexts reference.....	70
service object properties.....	67
signature attack objects	
creating.....	9
signature attack objects, creating.....	9
signature properties.....	8
support, technical See technical support	

**T**

TCP packet header matching.....	23
tcpdump.....	6
Tcpreplay.....	5
technical support	
contacting JTAC.....	xvi
time binding.....	17, 43
trojans.....	53

**U**

UDP packet header matching.....	23
unknown vulnerabilities, description of.....	3

**V**

vulnerabilities, described.....	2
---------------------------------	---

**W**

Wireshark.....	5
within bytes constraint.....	18, 45
within packets constraint.....	18, 45
worms.....	53