Snort 3 インスペクタリファレンス

最終更新：2026 年 5 月 11 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

【注意】シスコ製品をご使用になる前に、安全上の注意（**www.cisco.com/jp/go/safety_warning/**）をご確認ください。本書は、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2021–2024 Cisco Systems, Inc. All rights reserved.

目 次

第 1 章 はじめに 1

Snort 3 検査について 1

Snort 3 インスペクタの概要 3

Snort 3 のプロトコルとサービスの識別 8

第 I 部 : Snort 3 インスペクタ 11

第 2 章 ARP スプーフィングインスペクタ 13

ARP スプーフィングインスペクタの概要 13

ARP スプーフィングインスペクタのパラメータ 14

ARP スプーフィングインスペクタのルール 14

ARP スプーフィングインスペクタの侵入ルールのオプション 14

第 3 章 バインダインスペクタ 15

バインダインスペクタの概要 15

ポートレス設定でのサービスの自動検出 16

バインダインスペクタを設定するためのベストプラクティス 17

バインダインスペクタのパラメータ 19

バインダインスペクタのルール 20

バインダインスペクタの侵入ルールのオプション 21

第 4 章 CIP インスペクタ 23

CIP インスペクタの概要 23

CIP インスペクタを設定するためのベストプラクティス 24

CIP インスペクタのパラメータ 24
CIP インスペクタのルール 26
CIP インスペクタの侵入ルールのオプション 26

第 5 章 **DCE SMB インスペクタ 29**
DCE SMB インスペクタの概要 29
DCE SMB インスペクタのパラメータ 32
DCE SMB インスペクタのルール 36
DCE インスペクタの侵入ルールのオプション 39

第 6 章 **DCE TCP インスペクタ 43**
DCE TCP インスペクタの概要 43
DCE TCP インスペクタのパラメータ 45
DCE TCP インスペクタのルール 47
DCE インスペクタの侵入ルールのオプション 48

第 7 章 **DNP3 インスペクタ 53**
DNP3 インスペクタの概要 53
DNP3 インスペクタのパラメータ 53
DNP3 インスペクタのルール 54
DNP3 インスペクタの侵入ルールのオプション 55

第 8 章 **FTP クライアントインスペクタ 59**
FTP クライアントインスペクタの概要 59
FTP クライアントインスペクタのパラメータ 60
FTP クライアントインスペクタのルール 61
FTP クライアントインスペクタの侵入ルールのオプション 61

第 9 章 **FTP サーバーインスペクタ 63**
FTP サーバーインスペクタの概要 63
FTP サーバーインスペクタのパラメータ 64

FTP サーバーインスペクタのルール 69
FTP サーバーインスペクタの侵入ルールのオプション 70

第 10 章 GTP 検査インスペクタ 71
GTP 検査インスペクタの概要 71
GTP 検査インスペクタのパラメータ 71
GTP 検査インスペクタのルール 74
GTP 検査インスペクタの侵入ルールのオプション 74

第 11 章 HTTP 検査インスペクタ 89
HTTP 検査インスペクタの概要 89
HTTP 検査インスペクタを設定するためのベストプラクティス 91
HTTP 検査インスペクタのパラメータ 91
HTTP 検査インスペクタのルール 100
HTTP 検査インスペクタの侵入ルールのオプション 107

第 12 章 IEC104 インスペクタ 125
IEC104 インスペクタの概要 125
IEC104 インスペクタのパラメータ 126
IEC104 インスペクタのルール 126
IEC104 インスペクタの侵入ルールのオプション 130

第 13 章 IMAP インスペクタ 133
IMAP インスペクタの概要 133
IMAP インスペクタのパラメータ 134
IMAP インスペクタのルール 136
IMAP インスペクタの侵入ルールのオプション 137

第 14 章 MMS インスペクタ 139
MMS インスペクタの概要 139
MMS インスペクタのパラメータ 140

MMS インスペクタのルール 140
MMS インスペクタの侵入ルールのオプション 140

第 15 章 **Modbus インスペクタ 143**
Modbus インスペクタの概要 143
Modbus インスペクタを設定するためのベストプラクティス 144
Modbus インスペクタのパラメータ 144
Modbus インスペクタのルール 144
Modbus インスペクタの侵入ルールのオプション 145

第 16 章 **Normalizer インスペクタ 147**
ノーマライザインスペクタの概要 147
Normalizer インスペクタのパラメータ 148
Normalizer インスペクタのルール 153
Normalizer インスペクタの侵入ルールのオプション 153

第 17 章 **POP インスペクタ 155**
POP インスペクタの概要 155
POP インスペクタのパラメータ 156
POP インスペクタのルール 158
POP インスペクタの侵入ルールのオプション 159

第 18 章 **ポートスキャンインスペクタ 161**
ポートスキャンインスペクタの概要 161
ポートスキャンインスペクタを設定するためのベストプラクティス 164
ポートスキャンインスペクタのパラメータ 165
ポートスキャンインスペクタのルール 175
ポートスキャンインスペクタの侵入ルールのオプション 177

第 19 章 **レートフィルタ 179**
レートフィルタの概要 179

レートフィルタのパラメータ 181
レートフィルタのルール 183
レートフィルタの侵入ルールのオプション 183
レートフィルタ操作の確認 183

第 20 章 S7CommPlus インスペクタ 185
S7CommPlus インスペクタの概要 185
S7CommPlus インスペクタを設定するためのベストプラクティス 186
S7CommPlus インスペクタのパラメータ 186
S7CommPlus インスペクタのルール 186
S7CommPlus インスペクタの侵入ルールのオプション 187

第 21 章 SIP インスペクタ 191
SIP インスペクタの概要 191
SIP インスペクタのパラメータ 192
SIP インスペクタのルール 195
SIP インスペクタの侵入ルールのオプション 197

第 22 章 SMTP インスペクタ 199
SMTP インスペクタの概要 199
SMTP インスペクタを設定するためのベストプラクティス 200
SMTP インスペクタのパラメータ 200
SMTP インスペクタのルール 209
SMTP インスペクタの侵入ルールのオプション 210

第 23 章 SnortML 211
SnortML ルール 212
SnortML パラメータ 212
SnortML ルールの有効化 212
SnortML インスペクタの無効化 213
統合イベントでの SnortML ルールイベントの表示 214

侵入イベントでの SnortML ルールイベントの表示 214

第 24 章 SSH インスペクタ 215

SSH インスペクタの概要 215

SSH インスペクタを設定するためのベストプラクティス 216

SSH インスペクタのパラメータ 216

SSH インスペクタのルール 218

SSH インスペクタの侵入ルールのオプション 218

第 25 章 ストリーム ICMP インスペクタ 219

ストリーム ICMP インスペクタの概要 219

ストリーム ICMP インスペクタを設定するためのベストプラクティス 220

ストリーム ICMP インスペクタのパラメータ 220

ストリーム ICMP インスペクタのルール 220

ストリーム ICMP インスペクタの侵入ルールのオプション 220

第 26 章 ストリーム IP インスペクタ 221

ストリーム IP インスペクタの概要 221

ストリーム IP インスペクタを設定するためのベストプラクティス 222

ストリーム IP インスペクタのパラメータ 222

ストリーム IP インスペクタのルール 224

ストリーム IP インスペクタの侵入ルールのオプション 225

第 27 章 ストリーム TCP インスペクタ 227

ストリーム TCP インスペクタの概要 227

ストリーム TCP インスペクタを設定するためのベストプラクティス 228

TCP ストリームのリアセンブルのためのベストプラクティス 229

ストリーム TCP インスペクタのパラメータ 230

ストリーム TCP インスペクタのルール 235

ストリーム TCP インスペクタの侵入ルールのオプション 237

第 28 章 ストリーム UDP インスペクタ 241

ストリーム UDP インスペクタの概要 241

ストリーム UDP インスペクタを設定するためのベストプラクティス 242

ストリーム UDP インスペクタのパラメータ 242

ストリーム UDP インスペクタのルール 242

ストリーム UDP インスペクタの侵入ルールのオプション 242

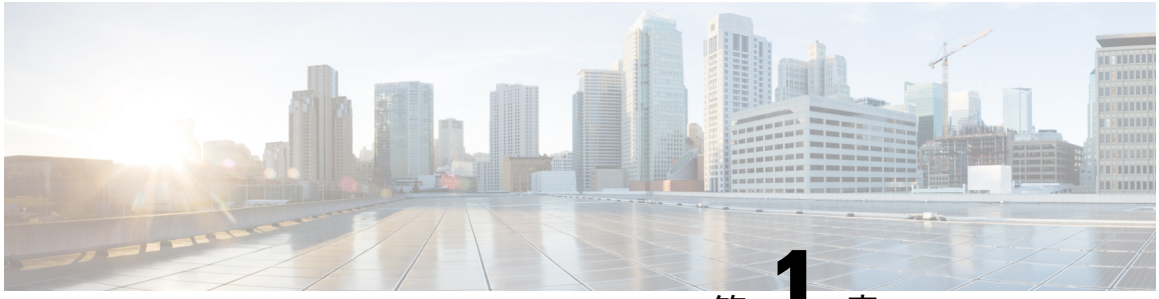
第 29 章 Telnet インスペクタ 243

Telnet インスペクタの概要 243

Telnet インスペクタのパラメータ 244

Telnet インスペクタのルール 245

Telnet インスペクタの侵入ルールのオプション 245



第 1 章

はじめに

- Snort 3 検査について （1 ページ）
- Snort 3 インスペクタの概要 （3 ページ）
- Snort 3 のプロトコルとサービスの識別 （8 ページ）

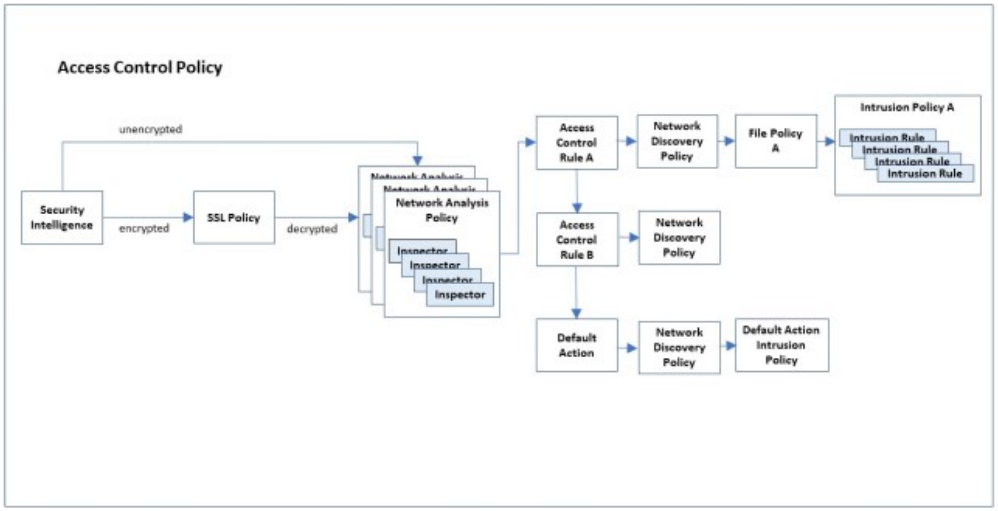
Snort 3 検査について

Snort 侵入防御システム（IPS）はリアルタイムでネットワークトラフィックを分析してパケットを詳細に検査します。Snort は、トラフィックの異常や、ネットワークプローブおよび攻撃を検出してブロックできます。Snort 3 は Snort の最新バージョンです。詳細については、https://snort.org/snort3を参照してください。

Snort は高いパフォーマンスと拡張性を実現するように設計されています。Snort には、インスペクタと呼ばれる設定可能な一連のプラグインが含まれています。Snort インスペクタは、特定のタイプのネットワークプロトコルまたはプローブのトラフィックを検出および分析し、メッセージを正規化してパケット分析を強化し、メッセージに埋め込まれた特定のタイプのファイルを検査できます。ネットワーク分析ポリシー（NAP）で Snort インスペクタを設定し、侵入ポリシーで侵入ルールを有効にします。

アクセス コントロール ポリシー

アクセス コントロール ポリシーは、いくつかの段階でトラフィックを処理します。次の図に、ポリシーの展開の例を示します。このドキュメントで取り上げる要素は、侵入ルールで使用される Snort 3 インスペクタとルールのオプションで、どちらも青色で強調表示されています。



ネットワーク分析ポリシーを使用すると、Snort 3 インスペクタを設定して、トラフィックプロトコルを決定し、データを抽出して正規化できます。複数のネットワーク分析ポリシーを設定でき、それぞれがデータを正規化するために独自に設定された Snort 3 インスペクタのコレクションを使用します。インスペクタは、データストリーム内の異常を検出すると警告を発することができますが、主な目的は、侵入ルール用のデータを準備することです。侵入ポリシーでは、設定した侵入ルールを適用して、回避、侵入、または攻撃の兆候がないかデータを調べます。

ネットワーク分析ポリシー内では、そのプロトコルを処理するインスペクタに固有の設定パラメータを設定することにより、特定のプロトコルを使用してデータの検査動作をカスタマイズできます。たとえば、POP データの検査動作を設定するには、`pop` インスペクタの構成パラメータを設定します。

それらのプロトコルに固有のルールオプションを使用してカスタム侵入ルールを作成することで、一部のプロトコルの侵入ポリシーをカスタマイズすることもできます。

複数のネットワーク分析ポリシーと複数の侵入ポリシーを使用して複雑な設定を確立する場合、システムは最初にデータを処理するネットワーク分析ポリシーを選択します。ネットワーク分析ポリシーで適切なインスペクタを適用して分析を実行した後、そのプロトコルの対応する侵入ポリシーにデータが自動的に渡されることはありません。アクセス コントロール ポリシーは、追加のテストを実行して、どの侵入ポリシーがデータを取得するかを決定します。そのため、アクセス制御、ネットワーク分析、および侵入ポリシーを設定するときは、データが正しいネットワーク分析と侵入ポリシーのペアによって分析されるようにしてください。詳細については、『*Cisco Secure Firewall Management Center Snort 3 Configuration Guide*』を参照してください。

侵入ルールの更新

Cisco は、Lightweight Security Packages（LSP）形式で侵入ルールの更新を定期的に発行します。これらの更新により、Snort 3 インスペクタの設定パラメータと侵入ルールのオプションのデフォルト値が変更される場合があります。

インスペクタの設定

Cisco Secure Firewall Management CenterWeb インターフェイスを介し、インスペクタを有効または無効にしたり、その設定を表示および変更したりできます。Cisco Secure Firewall Management CenterWeb インターフェイスは JSON 形式を使用してインスペクタ設定を記述します。詳細については、『*Cisco Secure Firewall Management Center Snort 3 Configuration Guide*』を参照してください。

インスペクタを使用するには、Firewall Management Center Web インターフェイスを介してインスペクタを有効にする必要があります。さらに、サービスインスペクタの場合、`binder` インスペクタでサービスインスペクタのエントリを設定する必要があります。詳細については、バインダインスペクタの概要 （15 ページ）を参照してください。

Snort 3 インスペクタ リファレンスには、Snort 3 インスペクタのパラメータと組み込みの侵入ルールのオプションのデフォルト設定が反映されています。システムは、LSP の更新、またはシステムで提供される基本のネットワーク アクセス ポリシーに応じて、異なるデフォルトを使用する場合があります。ネットワーク アクセス ポリシーのインスペクタ設定を最も正確に理解するには、Firewall Management Center Web インターフェイスに設定を表示します。

Snort 3 インスペクタの概要

Snort 3 のインスペクタは、Snort 2 プリプロセッサと同様に、パケットを分析して正規化するプラグインです。Snort 3 のインスペクタと設定のリストは、Snort 2 のプリプロセッサと設定のリストに直接対応していません。

Snort 3 インスペクタ

- ARP スプーフィングインスペクタ （13 ページ）
- バインダインスペクタ （15 ページ）
- CIP インスペクタ （23 ページ）
- DCE SMB インスペクタ （29 ページ）
- DCE TCP インスペクタ （43 ページ）
- DNP3 インスペクタ （53 ページ）
- FTP クライアントインスペクタ （59 ページ）
- FTP サーバーインスペクタ （63 ページ）
- GTP 検査インスペクタ （71 ページ）
- HTTP 検査インスペクタ （89 ページ）
- IEC104 インスペクタ （125 ページ）
- IMAP インスペクタ （133 ページ）

• MMS インスペクタ （139 ページ）
• Modbus インスペクタ （143 ページ）
• Normalizer インスペクタ （147 ページ）
• POP インスペクタ （155 ページ）
• ポートスキャンインスペクタ （161 ページ）
• レートフィルタ （179 ページ）
• S7CommPlus インスペクタ （185 ページ）
• SIP インスペクタ （191 ページ）
• SMTP インスペクタ （199 ページ）
• SSH インスペクタ （215 ページ）
• ストリーム ICMP インスペクタ （219 ページ）
• ストリーム IP インスペクタ （221 ページ）
• ストリーム TCP インスペクタ （227 ページ）
• ストリーム UDP インスペクタ （241 ページ）
• Telnet インスペクタ （243 ページ）

このドキュメントでは、Snort 3 インスペクタごとに次について説明します。

- インスペクタの目的と機能に関する一般的な情報。
- インスペクタのタイプ：
  - サービス：インターネット サービス プロトコル（HTTP、FTP、TCP、または UDP）で使用されるプロトコルデータユニット（PDU）を分析するインスペクタ。たとえば、`http_inspect`、`ftp_server` などがあります。
  - パッシブ：設定（ `ftp_client`、`ftp_server`）のみを提供するか、他の処理を容易にするインスペクタ（`binder`）。
  - パケット：他のインスペクタが処理を実行する前に生のパケットで処理を実行するインスペクタ。たとえば、`normalizer` などがあります。
  - プローブ：すべての検出が完了した後に、すべてのパケットに対して処理を実行するインスペクタ。たとえば、`port_scan` などがあります。
  - ストリーム：フロートラッキング、インターネットプロトコルの最適化、およびTCPのリアセンブルを実行するインスペクタ。たとえば、`stream_tcp`、`stream_ip` などがあります。

- 基本モジュール：複数のタイプのトラフィックの検査プロセスをサポートする機能を提供する、設定可能な組み込みの Snort 3 コンポーネント。たとえば、`rate_filter` などがあります。

- 使用法：
  - 検査：ネットワーク分析ポリシー（NAP）内でこれらのインスペクタを設定します。たとえば、`imap`、`ssh` などがあります。
  - グローバル、コンテキスト：これらのインスペクタを一度に設定します。たとえば、`port_scan`、`rate_filter` などがあります。

- インスタンスタイプ：
  - シングルトン：ネットワークアクセスポリシー内の単一のインスタンスに対してこれらのインスペクタを設定します。詳細については、シングルトンインスペクタ （6 ページ）を参照してください。
  - マルチトン：ネットワークアクセスポリシー（NAP）内の複数のインスタンスに対してこれらのインスペクタを設定します。NAP には、ネットワーク、ポート、または VLAN によって区別される複数のインスタンスを含めることができます。各インスタンスは、特定のトラフィックセグメントを処理するように一意に設定されています。詳細については、マルチトンインスペクタ （6 ページ）を参照してください。

- 他のインスペクタが必要：多くのインスペクタは、データストリームを完全に処理するために他のインスペクタに依存しています。あるインスペクタが他のインスペクタの設定を必要とする場合、このドキュメントではそれらの追加インスペクタを識別しています。
- インスペクタを設定するためのベストプラクティス：これらは、各インスペクタに固有の最適なパフォーマンスのための推奨事項です。
- インスペクタの設定パラメータ：Firewall Management Center Web インターフェイスで [ポリシー（Policies）] > [アクセス制御（Access Control）] > [ネットワーク分析ポリシー（Network Analysis Policy）] > ポリシー名 > [Snort 3 バージョン（Snort 3 Version）] > インスペクタ名で設定パラメータを設定できます。

（注） インスペクタのパラメータを変更する前に、インスペクタと有効な侵入ルール間の連携動作を理解しておくことをお勧めします。

- ルール：Snort 3 インスペクタはルールを使用してイベントを生成します。組み込みルールには、クラスタイプ、参照、およびその他のメタデータが含まれている場合があります。
- 侵入ルールのオプション：インスペクタによって処理されるデータタイプの侵入ルールのオプションを定義することで、侵入ルールをカスタマイズします。カスタム侵入ルールの管理については、*Cisco Secure Firewall Management Center Snort 3 Configuration Guide*を参照してください。

（注） カスタム侵入ルールの作成は高度な作業であり、注意して行う必要があります。このドキュメントに記載されていないインスペクタとルールのオプションを使用する必要がある場合があります。このドキュメントに記載されている一部のインスペクタと侵入ルールのオプションを使用するには、Snort のオープンソースドキュメントに記載されているインスペクタとルールのオプションの特定の設定が必要です。一部のルールのオプションは、Snort 高速パターンマッチ機能または検出カーソルの配置に影響を与えます。詳細については、https://www.snort.org/snort3 で入手可能な Snort 3 のオープンソースドキュメントを参照してください。

シングルトンインスペクタ

ネットワーク アクセス ポリシー（NAP）では、シングルトンインスペクタのインスタンスを 1 つだけ使用できます。

- シングルトンインスペクタは、マルチトンインスペクタのように NAP ごとに複数のインスタンスをサポートすることはできません。
- シングルトンインスペクタは、一部の特定のフローには適用されない場合があります。

次に例を示します。

```
{
   "normalizer":{
      "enabled":true,
      "type":"singleton",
      "data":{
         "ip4":{
            "df":true
         }
      }
   }
}
```

マルチトンインスペクタ

ネットワーク アクセス ポリシーでは、必要に応じて設定できるマルチトンインスペクタのインスタンスを 1 つ以上使用できます。マルチトンインスペクタは、ネットワーク、ポート、VLAN などの特定の条件に基づく設定をサポートしています。サポートされている一連の設定でインスタンスが構成されます。マルチトンはデフォルトのインスタンスを提供しますが、特定の条件に基づいて追加のインスタンスを定義できます。トラフィックがカスタマイズされたインスタンスの条件と一致すると、そのインスタンスの設定が適用されます。それ以外の場合は、デフォルトインスタンスの設定が適用されます。デフォルトのインスタンスの名前はインスペクタの名前と同じです。

マルチトンインスペクタの場合、オーバーライドされたインスペクタ設定をアップロードするときは、JSON ファイル内の各インスタンスの一致する `binder` 設定を定義する必要もあります。そのようにしないと、アップロードがエラーになります。新しいインスタンスを作成する

こともできますが、エラーを回避するために、作成するすべての新しいインスタンスに必ず binder 条件を含めてください。

次に例を示します。

- デフォルトのインスタンスが変更されたマルチトンインスペクタは、次のとおりです。

```
{
   "http_inspect":{
      "instances":[
         {
            "name":"http_inspect",
            "data":{
               "response_depth":5000
            }
         }
      ]
   }
}
```

- デフォルトのインスタンスとデフォルトの binder が変更されたマルチトンインスペクタは次のとおりです。

```
{
   "http_inspect":{
      "instances":[
         {
            "name":"http_inspect",
            "data":{
               "response_depth":5000
            }
         }
      ]
   },
   "binder":{
      "rules":[
         {
            "use":{
               "type":"http_inspect"
            },
            "when":{
               "role":"any",
               "ports":"8080",
               "proto":"tcp",
               "service":"http"
            }
         }
      ]
   }
}
```

- カスタムインスタンスとカスタム binder が追加されたマルチトンインスペクタは次のとおりです。

```
{
   "http_inspect":{
      "instances":[
         {
            "name":"http_inspect1",
            "data":{
               "response_depth":5000
            }
         }
```

```
        ]
      },
      "binder":{
         "rules":[
            {
               "use":{
                  "type":"http_inspect",
                  "name":"http_inspect1"
               },
               "when":{
                  "role":"any",
                  "ports":"8080",
                  "proto":"tcp",
                  "service":"http"
               }
            }
         ]
      }
   }
```

Snort 3 のプロトコルとサービスの識別

binder インスペクタは、すべての Snort サービスインスペクタに影響を与える独自の機能を実行します。binder は、Snort wizard モジュールとともに、ネットワークトラフィックを検査できるストリームまたはサービスインスペクタを決定します。binder インスペクタの設定には、ネットワーク分析ポリシーの別のインスペクタがトラフィックを検査する必要がある場合に定義するポート、ホスト、CIDR、およびサービスが含まれます。

wizard は、マルウェアコマンドチャネルと制御チャネルの検出を可能にする、ポートに依存しないサービス設定をサポートします。

（注） Cisco Secure Firewall Management CenterWeb インターフェイスから wizard を設定することはできません。

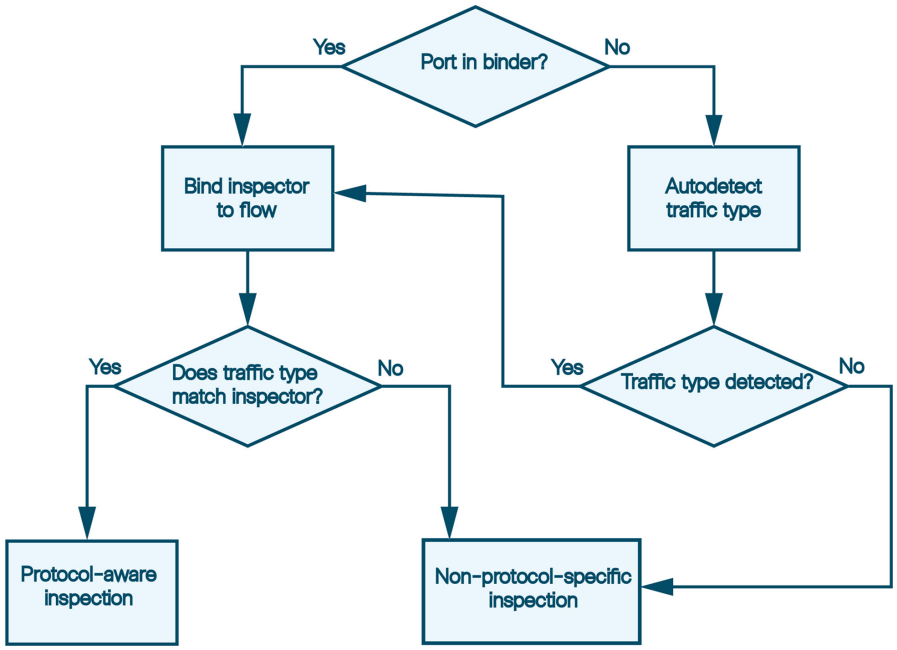
トラフィックがファイアウォールデバイスに到着すると、binder インスペクタは侵入ポリシーを検索し、適用する適切なネットワークアクセスポリシー（NAP）を選択します。NAP 内で、binder はデータフローに使用する適切なストリームとサービスインスペクタを決定します。その後、フローに関連付けられたサービスが変更された場合、NAP は binder を使用して別のサービス インスペクタを選択します。

binder インスペクタの設定には、トラフィックの特性を説明する when パラメータと、それらの特性に一致するトラフィックに適用するインスペクタを指定する use パラメータが含まれています。データフローに適用するインスペクタを決定する場合、binder インスペクタはトラフィックをその when 句に対して上から順に比較し、トラフィックに一致する最初の when 句に対応する use 句を適用します。

特定の binder 条件がデータフローに一致しない場合、wizard はデータフローを分析してサービスを決定します。wizard は binder を呼び出して、そのサービスに適したインスペクタを選択します。サービスを識別できない場合、binder は通常、ストリームインスペクタをフローに

バインドし、システムはペイロードの内容に関係なく、プロトコル固有ではないデータパケットのリアセンブルを実行します。

次の図に、インスペクタがプロトコル固有または非プロトコル固有の検査を実行する方法を示します。サービス検査は、`binder` インスペクタでの `port`、`host`、`service`、および `CIDR` パラメータの設定方法によって異なります。



Firewall Management Center Web インターフェイスの NAP の `binder` インスペクタで `use` パラメータと `when` パラメータを定義することでインスペクタの選択基準をカスタマイズできます。`binder` パラメータの詳細についてはバインダインスペクタの概要 （15 ページ）を参照してください。Firewall Management Center Web インターフェイスをナビゲートしてインスペクタを設定する方法については、*Cisco Secure Firewall Management Center Snort 3 Configuration Guide*を参照してください。

`binder` を正しく設定しないと、フローのサービスを検出したり、インスペクタをバインドしたりできません。ルールエンジンと自動検出がトラフィックを理解できず、識別できない場合、`binder` インスペクタでポートなどの `when` 基準を設定しても、検査は適用されません。たとえば、`binder` でポート 88 を HTTP ポートとして設定すると、その `binder` は `http_inspect` インスペクタをそのポートのすべてのフローにバインドします。ただし、フローが HTTP ではない場合、ルールエンジンはデータを HTTP として検査せず、代わりにポートベースの検出を実行します。

ネットワーク分析ポリシーでの自動検出と有効化または無効化されたインスペクタ

自動検出の動作は、対象のインスペクタがネットワーク分析ポリシーで有効化されているか無効化されているかによって異なります。対象のインスペクタがネットワーク分析ポリシーで有効になっている場合、自動検出は上記のように機能します。

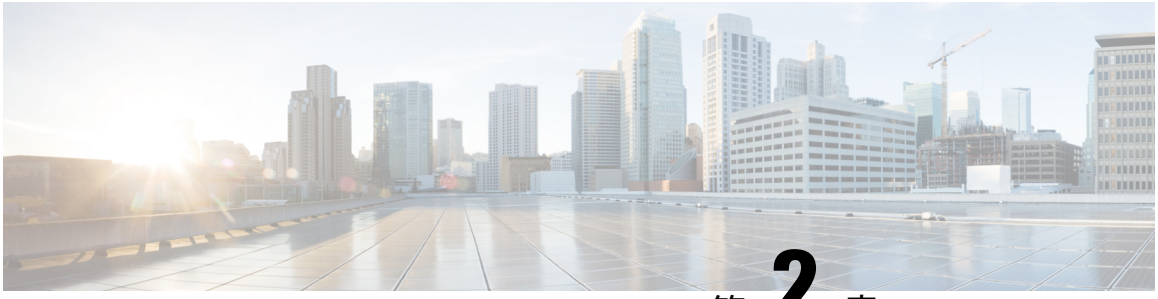
対象のインスペクタがネットワーク分析ポリシーで無効になっている場合でも、通常、自動検出は引き続きストリームインスペクタ（ストリーム TCP やストリーム UDP など）をフローにバインドします。ただし、ルールエンジンはサービスの検査も検出も実行しません。TCP フローの場合、ストリーム TCP インスペクタはリアセンブルを実行します。

第 部

Snort 3 インスペクタ

- ARP スプーフィングインスペクタ （13 ページ）
- バインダインスペクタ （15 ページ）
- CIP インスペクタ （23 ページ）
- DCE SMB インスペクタ （29 ページ）
- DCE TCP インスペクタ （43 ページ）
- DNP3 インスペクタ （53 ページ）
- FTP クライアントインスペクタ （59 ページ）
- FTP サーバーインスペクタ （63 ページ）
- GTP 検査インスペクタ （71 ページ）
- HTTP 検査インスペクタ （89 ページ）
- IEC104 インスペクタ （125 ページ）
- IMAP インスペクタ （133 ページ）
- MMS インスペクタ （139 ページ）
- Modbus インスペクタ （143 ページ）
- Normalizer インスペクタ （147 ページ）
- POP インスペクタ （155 ページ）
- ポートスキャンインスペクタ （161 ページ）
- レートフィルタ （179 ページ）
- S7CommPlus インスペクタ （185 ページ）
- SIP インスペクタ （191 ページ）
- SMTP インスペクタ （199 ページ）

- SnortML （211 ページ）
- SSH インスペクタ （215 ページ）
- ストリーム ICMP インスペクタ （219 ページ）
- ストリーム IP インスペクタ （221 ページ）
- ストリーム TCP インスペクタ （227 ページ）
- ストリーム UDP インスペクタ （241 ページ）
- Telnet インスペクタ （243 ページ）



第 2 章

ARP スプーフィングインスペクタ

- ARP スプーフィングインスペクタの概要 （13 ページ）
- ARP スプーフィングインスペクタのパラメータ （14 ページ）
- ARP スプーフィングインスペクタのルール （14 ページ）
- ARP スプーフィングインスペクタの侵入ルールのオプション （14 ページ）

ARP スプーフィングインスペクタの概要

タイプ	インスペクタ（ネットワーク）
使用方法	検査
インスタンス タイプ	単一
その他のインスペクタが必要	なし
有効	`true`

Address Resolution Protocol（ARP）は、アドレス解決のために単一のネットワーク内で使用されるステートレス通信プロトコルです。要求と応答を交換する場合、ARP はホスト間の認証を行いません。

ARP スプーフィングは、ローカルエリアネットワーク（LAN）内で ARP を使用する中間者攻撃の一種です。攻撃者は、特定のホストのMedia Access Control（MAC）アドレス宛てのメッセージを傍受することで、ホストへの通信を変更します。

`arp_spoof` インスペクタは、ARP パケットを分析し、ユニキャスト ARP 要求を検出します。ARP キャッシュ上書き攻撃を検出するために、ARP スプーフィングインスペクタは、一貫性のないイーサネットから IP へのマッピングを識別します。

有効になっている場合、`arp_spoof` インスペクタは次のことを行います。

- イーサネットアドレスと ARP パケット内のアドレスを検査します。不整合が発生すると、インスペクタはルール 112:2 またはルール 112:3 を使用してアラートを生成し、インライン展開で問題のあるパケットをドロップします。

- ユニキャスト ARP 要求を確認します。ユニキャスト ARP 要求が検出されると、インスペクタはルール 112:1 を使用してアラートを生成し、インライン展開で問題のあるパケットをドロップします。
- `hosts[]` パラメータが指定されている場合、インスペクタはその情報を使用して ARP キャッシュ上書き攻撃を検出します。このような攻撃が検出された場合、インスペクタはルール 112:4 を使用してアラートを生成し、インライン展開で問題のあるパケットをドロップします。

ARP スプーフィングインスペクタのパラメータ

`arp_spoof` インスペクタは、Cisco Secure Firewall Management Center Web インターフェイスにデフォルト設定パラメータ値を提供しません。

ARP スプーフィングインスペクタのルール

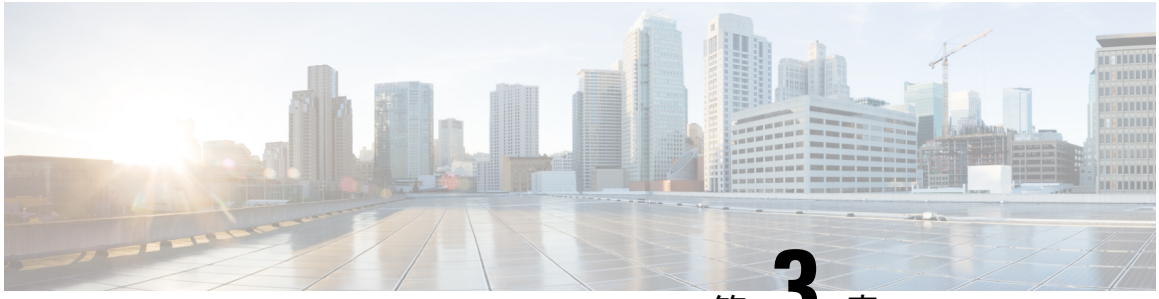
`arp_spoof` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *1: ARP* スプーフィングインスペクタのルール

GID:SID	ルール メッセージ
112:1	ユニキャスト ARP 要求（unicast ARP request）
112:2	送信元のイーサネット/ARP 不一致要求（ethernet/ARP mismatch request for source）
112:3	接続先に対するイーサネット/ARP 不一致要求（ethernet/ARP mismatch request for destination）
112:4	ARP キャッシュの上書き攻撃が試行された（attempted ARP cache overwrite attack）

ARPスプーフィングインスペクタの侵入ルールのオプション

`arp_spoof` インスペクタには、侵入ルールオプションはありません。



第 3 章

バインダインスペクタ

- バインダインスペクタの概要 （15 ページ）
- ポートレス設定でのサービスの自動検出 （16 ページ）
- バインダインスペクタを設定するためのベストプラクティス （17 ページ）
- バインダインスペクタのパラメータ （19 ページ）
- バインダインスペクタのルール （20 ページ）
- バインダインスペクタの侵入ルールのオプション （21 ページ）

バインダインスペクタの概要

タイプ	インスペクタ（パッシブ）
使用方法	検査
インスタンス タイプ	単一
その他のインスペクタが必要	確立されたバインディングに依存
有効	`true`

各ネットワーク分析ポリシー（NAP）には、`binder`インスペクタが 1 つあります。`binder`は、トラフィックを検査するために特定のサービスインスペクタを使用するタイミングを決定します。`binder` インスペクタの設定には、同じ NAP 内の別のインスペクタがトラフィックを検査する必要がある場合に定義するポート、ホスト、CIDR、およびサービスが含まれます。`binder` ルールが新しいフローに一致すると、対象のインスペクタがフローにバインドされます。

`binder`インスペクタは、自動検出ウィザードと連携して、ポートに依存しないサービスの設定と、マルウェア コマンドおよび制御チャネルの検出を実行できます。詳細については、Snort 3 のプロトコルとサービスの識別 （8 ページ）を参照してください。

バインディングは、セッションの開始時に評価され、適切なサービスがセッションで識別された場合に再度評価されます。バインディングは、上から下に評価される使用時ルールのリストです。Snort は、最初に一致したネットワークおよびサービス設定を使用してトラフィックを検査します。

例

たとえば、CIP トラフィックを検査するように NAP を設定する場合は、次の手順を実行します。

- NAP の `binder` インスペクタで、検査するトラフィックの正しいポート、ロール、およびプロトコル情報を使用して、`"type":"cip"` セクションを更新します。
- 同じ NAP の `cip` インスペクタでデフォルト値を確認し、CIP トラフィックを検査するために必要な調整を行います。

次に、`cip` の設定とバインディングの例を示します。この例では、バインダインスペクタのパラメータ （19 ページ）で説明したオプションを使用します。

```
{
   "use": {
     "type":"cip"
   },
   "when": {
     "proto":"udp",
     "ports":"22222 33333",
     "role":"server"
   }
 },
 {
   "use": {
     "type":"cip"
   },
   "when": {
     "role":"server",
     "ports":"44818",
     "proto":"tcp"
   }
 },
```

ポートレス設定でのサービスの自動検出

自動検出ウィザードにより、ポートに依存しないサービスの設定と、マルウェアコマンドおよび制御チャネルの検出が可能になります。トラフィックが着信すると、`binder` インスペクタは最初に自動検出 `wizard` をフローに付加して最初のペイロードを確認し、トラフィックが使用しているサービスを特定します。たとえば、`GET` は HTTP を示し、`HELO` は SMTP を示します。サービスが決定されると、Snort は適切なサービスインスペクタをフローにバインドし、自動検出 `wizard` をフローから切り離します。

（注） Cisco Secure Firewall Management Center Web インターフェイスから自動検出 `wizard` を設定することはできません。

ルールエンジンと自動検出 `wizard` がトラフィックを理解して識別できない場合、`binder` インスペクタでポートを設定しても検査は強制されません。

自動検出とバインダの設定

binder インスペクタは侵入ルールを上から順に照合し、トラフィックに一致する最初のルールを適用します。フローで検出されたサービスに binder インスペクタを設定していない場合でも、自動検出ウィザードはフローを関連するインスペクタにバインドできます。次に例を示します。

- ペイロードが GET で、自動検出ウィザードがトラフィックタイプを HTTP として識別した場合、binder インスペクタは HTTP インスペクタをそのフローにバインドします。
- トラフィックタイプを識別できない場合、ルールエンジンは非プロトコル固有の検査を実行します。

ポートを正しく設定しないと、binder インスペクタはそのフローのサービスを自動検出できず、インスペクタをバインドできません。たとえば、ポート 88 をバインダに HTTP ポートとして設定すると、binder インスペクタは HTTP インスペクタをそのポートのすべてのフローにバインドします。ただし、フローが HTTP ではない場合、ルールエンジンはフローを HTTP として検査しません。代わりに、検査と検出がタイムアウトします。

ネットワーク分析ポリシーでの自動検出とインスペクタの有効化または無効化

自動検出の動作は、対象のインスペクタがネットワーク分析ポリシーで有効化されているか無効化されているかによって異なります。対象のインスペクタがネットワーク分析ポリシーで有効になっている場合、自動検出は期待どおりに機能します。

対象のインスペクタがネットワーク分析ポリシーで無効になっている場合でも、通常、自動検出は引き続きストリームインスペクタ（ストリーム TCP やストリーム UDP など）をフローにバインドします。ただし、ルールエンジンはサービスの検査も検出も実行しません。TCP フローの場合、ストリーム TCP インスペクタはリアセンブルを実行します。

バインダインスペクタを設定するためのベストプラクティス

バインダインスペクタを設定するときは、次のベストプラクティスを考慮してください。

- そのインスペクタに必要な場合を除き、バインダインスペクタでポートを設定しないでください。ルールエンジンがトラフィックを自動検出できる場合、ポートの設定は有効性を改善しません。ただし、ポート設定が正しくないと、回避の検出に失敗する可能性があります。
- 1 つのインスペクタのみにポートを設定します。異なるプロトコルとインスペクタのバインダでポートが 2 回設定されている場合、最初のインスペクタが自動的にトリガーされます。
- デフォルトの binder インスペクタの設定に表示されない場合は、サービスインスペクタの設定を binder インスペクタに追加します。たとえば、cip インスペクタを使用する場合は、その cip インスペクタの use オプションと when オプションをバインダに追加します。

- ストリーム TCP インスペクタの場合、オペレーティングシステムの設定をカスタムバインドするようにネットワークを設定します。ネットワークの設定はすべてのポートに適用されます。
- サービスインスペクタの場合、バインダがフロー内のプロトコルを自動検出できる場合は、ハードポートバインディングを回避します。プロトコルが検出可能でない場合、ハードポートバインディングは検出と検査を保証しません。

ポートの設定が必要なインスペクタ

関連するプロトコルでは自動検出が機能しないため、次のインスペクタのバインダインスペクタでポートを設定します。

- `cip`
- `gtp_inspect`
- `iec104`
- `modbus`
- `s7commplus`

ポートの設定を必要としないインスペクタ

関連するプロトコルに対して自動検出が機能するため、次のインスペクタのバインダインスペクタでポートを設定しないでください。

- `arp_spoof`
- `dce_smb`
- `dce_tcp`
- `dnp3`
- `ftp_client`
- `ftp_server`
- `http_inspect`
- `imap`
- `normalizer`
- `pop`
- `port_scan`
- `sip`
- `smtp`
- `ssh`
- `stream_icmp`

- stream_ip
- stream_tcp
- stream_udp
- telnet

バインダインスペクタのパラメータ

binder[]

バインダには、 when オブジェクトと use オブジェクトのペアとして定義されたルールの配列が含まれています。

型：配列

例：

```
{
    binder: {
        rules: [
            {
                "when": {
                    ...
                },
                "use": {
                    ...
                 }
            },
            {
                "when": {
                    ...
                },
                "use": {
                    ...
                }
            }
        ]
    }
}
```

binder[].use.type

when パラメータの条件が一致したときにデータフローにバインドするインスペクタを指定します。たとえば、CIP トラフィックを検査するには、値 cip を使用して use.type を追加します。

型：文字列

有効な値：このドキュメントで説明されている Snort 3 インスペクタの名前。

デフォルト値：binder インスペクタには、サポートされている各インスペクタの use.type パラメータが含まれています。

binder[].when.proto

use.type で指定されたインスペクタにデータフローをバインドするためにトラフィックが一致する必要があるプロトコルを指定します。たとえば、ネットワーク分析ポリシーが TCP トラフィックを検査するように設定されている場合、binder インスペクタはこのパラメータを tcp に設定する必要があります。

型：列挙体

有効な値：any、ip、icmp、tcp、udp、user、file

デフォルト値：binder インスペクタには、各プロトコルの when.proto パラメータが含まれています。

binder[].when.ports

use.type で指定されたインスペクタにデータフローをバインドするためにトラフィックが一致する必要があるポートを指定します。たとえば、TCP ポート 80 のトラフィックを検査するには、when.proto を tcp に設定し、when.ports を 80 に設定します。

10 進数または 16 進数の整数で表される 1 つ以上のポートのリストを指定します。複数のポートはスペースで区切り、リストを二重引用符で囲みます。

型：文字列

有効な範囲：1 ～ 65535

デフォルト値：65535（この値は when.proto の値によって異なる場合があります。）

binder[].when.role

use.type で指定したインスペクタにフローをバインドするためにトラフィックが一致する必要があるロールを指定します。

型：列挙体

有効な値：client、server、any

デフォルト値：any

use.type で指定されたインスペクタにフローをバインドするためにトラフィックが一致する必要があるサービスを指定します。

型：文字列

有効な値：着信データをカプセル化する可能性のあるサービスの名前（例： netbios-ssn または dcerpc）。

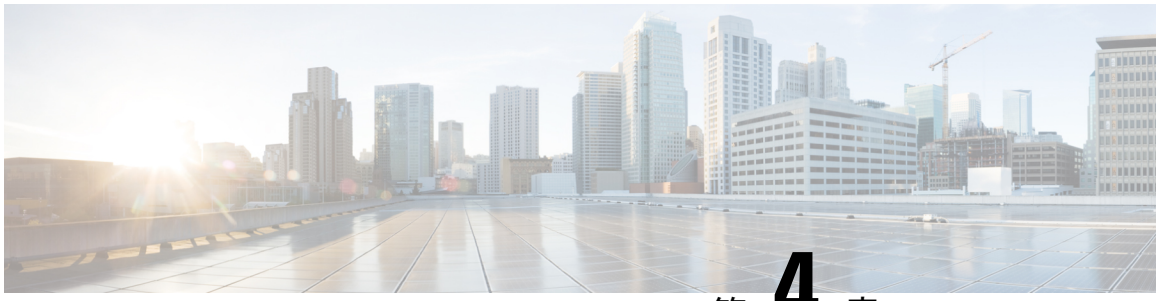
デフォルト値：なし

バインダインスペクタのルール

binder インスペクタに関連付けられたルールはありません。

バインダインスペクタの侵入ルールのオプション

binder インスペクタには、侵入ルールのオプションはありません。



第 4 章

CIP インスペクタ

- CIP インスペクタの概要 （23 ページ）
- CIP インスペクタを設定するためのベストプラクティス （24 ページ）
- CIP インスペクタのパラメータ （24 ページ）
- CIP インスペクタのルール （26 ページ）
- CIP インスペクタの侵入ルールのオプション （26 ページ）

CIP インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`false`

Common Industrial Protocol（CIP）は、産業用自動化アプリケーションをサポートするためのアプリケーションプロトコルです。EtherNet/IP（ENIP）は、イーサネットベースのネットワークで使用される CIP の実装です。

`cip` インスペクタは、TCP または UDP で実行されている CIP トラフィックと ENIP トラフィックを検出し、それを侵入ルールエンジンに送信します。カスタム侵入ルールで CIP および ENIP のキーワードを使用すると、CIP および ENIP トラフィックで攻撃を検出できます。

（注） Snort 3 では、`cip` インスペクタは CIP アプリケーションディテクタをサポートしていません。CIP アプリケーション検出を実装するには、カスタム CIP 侵入ルールを作成してインポートし、適切な IPS ルールを有効にします。詳細については、管理アプリケーションの Snort 3 設定に関するドキュメントを参照してください。

CIP インスペクタを設定するためのベストプラクティス

cip インスペクタを設定するときは、次のベストプラクティスを考慮してください。

- デフォルトの CIP 検出ポート 44818 とその他の CIP ポートを binder インスペクタに追加する必要があります。
- 侵入防御アクションをアクセス コントロール ポリシーのデフォルトのアクションとして使用することをお勧めします。
- CIP アプリケーションと ENIP アプリケーションを検出するには、対応するカスタム ネットワーク分析ポリシーで cip インスペクタを有効にする必要があります。
- アクセス コントロール ルールを使用して CIP アプリケーションまたは ENIP アプリケーションのトラフィックをブロックするには、対応するネットワーク分析ポリシーで Normalizer インスペクタとそのインラインモードが有効になっている（デフォルト設定）ことを確認してください。
- cip インスペクタのルールと CIP 侵入ルールをトリガーするトラフィックをドロップするには、対応する侵入ポリシーの **Drop when Inline** が有効になっていることを確認します。
- cip インスペクタは、次のいずれかのアクセス コントロール ポリシーのデフォルトアクションをサポートしていません。
  - **アクセス コントロール：すべてのトラフィックを信頼**
  - **アクセス コントロール：すべてのトラフィックをブロック**
- cip インスペクタは、CIP アプリケーションのアプリケーション可視性（ネットワーク検出を含む）をサポートしていません。

CIP インスペクタのパラメータ

CIP TCP ポートの設定

binder インスペクタは、CIP TCP ポートの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
                    "role": "server",
                    "proto": "tcp",
                    "ports": "44818"
        },
        "use": {
                    "type": "cip"
        }
```

```
    }
]
```

embedded_cip_path

埋め込まれた CIP 接続パスをインスペクタで確認するかどうかを決定します。

型：文字列

有効な値は、次のとおりです。

- `"false"`
- 二重引用符で囲まれた CIP パス（`"0x2 0x36"` など）。

デフォルト値：`"false"`

unconnected_timeout

デフォルトの未接続タイムアウトを秒単位で設定します。CIP 要求メッセージにプロトコル固有のタイムアウト値が含まれておらず、TCP 接続あたりの未接続な同時要求の最大数に達した場合は、このパラメータで指定した秒数の間、システムがメッセージの時間を測定します。タイマーが満了すると、他の要求用のスペースを確保するために、メッセージが削除されます。

0 を指定すると、プロトコル固有のタイムアウトが設定されていないすべてのトラフィックが最初にタイムアウトになります。

型：整数

有効な範囲：`0` 〜 `360`

デフォルト値：`300`

max_unconnected_messages

TCP 接続あたりの同時未接続 CIP メッセージの最大数を設定します。無応答のままになる可能性がある同時要求の最大数に到達すると、システムはその接続を閉じます。

型：整数

有効な範囲：`1` 〜 `10000`

デフォルト値：`100`

max_cip_connections

システムが TCP 接続ごとに許可する同時 CIP 接続の最大数を設定します。

型：整数

有効な範囲：`1` 〜 `10000`

デフォルト値：`100`

CIP インスペクタのルール

cip インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *2: CIP* インスペクタのルール

GID:SID	ルール メッセージ
148:1	CIP データの形式が誤っている（CIP data is malformed）
148:2	CIP データは ODVA 基準に準拠していない（CIP data is non-conforming to ODVA standard）
148:3	CIP 接続の制限を超えている。最も長い間使用されていない接続が削除された（CIP connection limit exceeded. Least recently used connection removed）
148:4	CIP 未接続要求の制限を超えている。最も古い要求が削除された（CIP unconnected request limit exceeded. Oldest request removed）

CIP インスペクタの侵入ルールのオプション

cip_attribute

CIP 属性を照合する検出パラメータ。

型：間隔

シンタックス：`cip_attribute: <range_operator><positive integer>;` または `cip_attribute: <positive integer><range_operator><positive integer>;`

有効な値：0 ～ 65535 の 1 つ以上の一連の整数と範囲の形式の表に指定されている `range_operator` の 1 つ。

例：`cip_attribute: <100;`

cip_class

CIP クラスを照合する検出パラメータ。

型：間隔

シンタックス：`cip_class: <range_operator><positive integer>;` または `cip_class: <positive integer><range_operator><positive integer>;`

有効な値：0 ～ 65535 の 1 つ以上の一連の整数と範囲の形式の表に指定されている `range_operator` の 1 つ。

例：`cip_class: <25;`

cip_conn_path_class

CIP 接続パスクラスを照合する検出パラメータ。

型：間隔

シンタックス：`cip_conn_path_class: <range_operator><positive integer>;` または `cip_conn_path_class: <positive integer><range_operator><positive integer>;`

有効な値：0 ～ 65535 の 1 つ以上の一連の整数と範囲の形式の表に指定されている `range_operator` の 1 つ。

例：`cip_conn_path_class: <85;`

cip_instance

CIP インスタンスを照合する検出パラメータ。

型：間隔

シンタックス：`cip_instance: <range_operator><positive integer>;` または `cip_instance: <positive integer><range_operator><positive integer>;`

有効な値：0 ～ 65535 の 1 つ以上の一連の整数と範囲の形式の表に指定されている `range_operator` の 1 つ。

例：`cip_instance: <15;`

cip_req

CIP 要求を照合する検出パラメータ。

シンタックス：`cip_req;`

例：`cip_req;`

cip_rsp

CIP 応答を照合する検出パラメータ。

シンタックス：`cip_rsp;`

例：`cip_rsp;`

cip_service

CIP サービスを照合する検出パラメータ。

型：間隔

シンタックス：`cip_service: <range_operator><positive integer>;` or `cip_service: <positive integer><range_operator><positive integer>;`

有効な値：0 ～ 127 の 1 つ以上の一連の整数と範囲の形式の表に指定されている `range_operator` の 1 つ。

例：`cip_service: <50;`

cip_status

CIP 応答ステータスを照合する検出パラメータ。

型：間隔

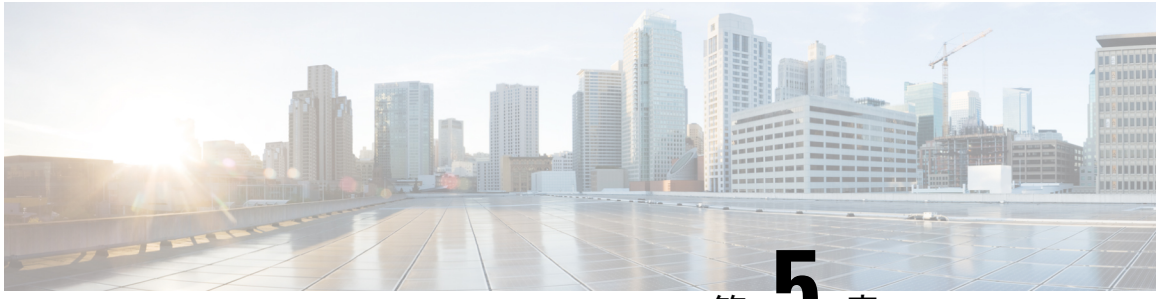
シンタックス：`cip_status: <range_operator><positive integer>;` または `cip_status: <positive integer><range_operator><positive integer>;`

有効な値：0 ～ 255 の 1 つ以上の一連の整数と範囲の形式の表に指定されている `range_operator` の 1 つ。

例：`cip_status: <250;`

表 *3*: 範囲の形式

範囲の形式	演算子	説明
operator i		
	<	より少ない
	>	右辺と比較して大きい
	=	等しい
	≠	等しくない
	≤	以下
	≥	以上
j operator k		
	<>	j よりも大きく、k よりも小さい
	<=>	j 以上で k 以下



第 5 章

DCE SMB インスペクタ

- DCE SMB インスペクタの概要 （29 ページ）
- DCE SMB インスペクタのパラメータ （32 ページ）
- DCE SMB インスペクタのルール （36 ページ）
- DCE インスペクタの侵入ルールのオプション （39 ページ）

DCE SMB インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	なし
有効	`true`

DCE/RPC プロトコルにより、別々のネットワーク ホスト上のプロセスが、同一ホストに配置されている場合と同様に通信できます。通常、このようなプロセス間通信はホスト間で TCP および UDP 経由で転送されます。TCP 転送では、DCE/RPC が Windows Server Message Block（SMB）プロトコルまたは Samba でさらにカプセル化されることがあります。Samba は、Windows や UNIX または Linux のオペレーティングシステムから構成される混合環境でプロセス間通信に使用されるオープンソースの SMB 実装です。

ほとんどの DCE/RPC エクスプロイトは、DCE/RPC サーバ（ネットワーク上の Windows または Samba が稼働している任意のホスト）を対象とした DCE/RPC クライアント要求で発生します。またエクスプロイトはサーバ応答でも発生することがあります。

IP によりすべての DCE/RPC トランスポートがカプセル化されます。TCP は、SMB など、すべてのコネクション型 DCE/RPC を伝送します。

`dce_smb` インスペクタは、SMB プロトコルでコネクション型 DCE/RPC を検出し、ヘッダー長やデータフラグメントの順序などのプロトコル固有の特性を使用して、次のことを行います。

- SMB トランスポートでカプセル化されている DCE/RPC 要求と応答を検出します。
- DCE/RPC データストリームを分析し、DCE/RPC トラフィック内の異常な動作と回避技術を検出します。
- SMB データストリームを分析し、異常な SMB 動作と回避技術を検出します。
- SMB のセグメント化を解除し、DCE/RPC をデフラグします。
- ルールエンジンで処理できるように DCE/RPC トラフィックを正規化します。

次の図に、DCE SMB インスペクタが SMB トランスポートのためにトラフィックの処理を開始するポイントを示します。

dce_smb インスペクタは通常、NetBIOS セッションサービス用のウェルノウン TCP ポート 139 か、または同様に実装されているウェルノウン Windows ポート 445 で SMB トラフィックを受信します。SMB には DCE/RPC の伝送以外にも多数の機能があるため、インスペクタは SMB トラフィックが DCE/RPC トラフィックを伝送しているかどうかをまずテストします。伝送していない場合は処理を停止し、伝送している場合は処理を続行します。

dce_smb インスペクタのパラメータと機能の説明には、Microsoft Remote Procedure Call（MSRPC）と呼ばれる DCE/RPC の Microsoft の実装と、SMB および Samba の両方が含まれています。

ターゲットベースのポリシー

Windows および Samba の DCE/RPC の実装は大きく異なります。たとえば、Windows のすべてのバージョンは、DCE/RPC トラフィックの最適化時に最初のフラグメントの DCE/RPC コンテキスト ID を使用しますが、Samba のすべてのバージョンは、最後のフラグメントのコンテキスト ID を使用します。また、特定の関数呼び出しを識別するために、Windows Vista では最初のフラグメントの opnum（操作番号）ヘッダーフィールドを使用しますが、Samba とその他のすべてのバージョンの Windows では最後のフラグメントの opnum フィールドを使用します。

Windows と Samba の SMB の実装には大きな違いがあります。たとえば、Windows は名前付きパイプの操作時に SMB OPEN および READ コマンドを認識しますが、Samba はこれらのコマンドを認識しません。

そのため、dce_smb インスペクタはターゲットベースのアプローチを使用します。dce_smb インスペクタインスタンスを設定すると、policy パラメータは DCE/RPC SMB プロトコルの実装を指定します。これをホスト情報と組み合わせることで、デフォルトのターゲットベースのサーバーポリシーが確立されます。必要に応じて、他のホストおよび DCE/RPC SMB の実装を対象とする追加のインスペクタを設定できます。デフォルトのターゲットベースのサーバーポリシーで指定されている DCE/RPC SMB の実装は、別の dce_smb インスペクタインスタンスの対象になっていないすべてのホストに適用されます。

dce_smb インスペクタが policy パラメータを使用して対象にできる DCE/RPC SMB の実装は次のとおりです。

- WinXP（デフォルト）
- Win2000
- WinVista
- Win2003
- Win2008
- Win7
- Samba
- Samba-3.0.37
- Samba-3.0.22
- Samba-3.0.20

ファイル検査

dce_smb インスペクタは、SMB バージョン 1、2、および 3 のファイル検査をサポートします。

dce_smb インスペクタは、通常の SMB ファイル転送を確認します。これには、ファイル処理によるファイルタイプと署名の確認と、file_data ルールオプションのポインタの設定が含まれます。 dce_smb インスペクタは、file_id インスペクタ（Snort 3 オープンソースドキュメントで説明されており、https://www.snort.org/snort3 で入手可）と連携して使用する場合、SMB バージョン 1、2、および 3 の通常の SMB ファイル転送の検査をサポートします。ファイル検査を有効にするには、必要に応じて file_id インスペクタを構成し、dce_smb smb_file_inspection および smb_file_depth パラメータを設定します。smb_file_depth パラメータは file_data IPS ルールオプションが示すポインタから開始して、file_id が確認するファイルデータのバイト数を示します。詳細については、https://www.snort.org/snort3 で入手可能な Snort 3 のオープンソースのドキュメントを参照してください。

最適化

dce_smb インスペクタは、フラグメント化されたデータパケットのリアセンブルをサポートしています。この機能は、インラインモードで、完全な最適化が実行される前の検査プロセスの早い段階でエクスプロイトをキャッチしたり、フラグメント化を利用してそれ自体を隠すエクスプロイトをキャッチしたりするのに役立ちます。最適化を無効にすると、多数の誤検知が発生する可能性があることに注意してください。

DCE SMB インスペクタのパラメータ

DCE SMB ポートの設定

binder インスペクタは、DCE SMB ポートの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
   {
       "when": {
           "role":"any",
           "service":"netbios-ssn",
           "ports": ""
       },
        "use": {
            "type":"dce_smb"
       }
   }
]
```

max_frag_len

最適化のために許可される最大フラグメント長をバイト単位で指定します。より大きなフラグメントを処理する場合、インスペクタは、最適化する前にパケットコンテンツをこのサイズに考慮します。

（注） このパラメータで指定する値は、確実に検出するためにルールでデータを調べる必要がある深さ以上にする必要があります。すべてのデータの検出が確実に行われるようにするには、デフォルト値を使用します。

型：整数

有効な範囲：1514 ～ 65535

デフォルト値：65535

smb_max_compound

1 つの SMB 要求で処理するコマンドの最大数を指定します。

型：整数

有効な範囲：0 ～ 255

デフォルト値：3

smb_max_chain

許可されている連結 SMB AndX コマンドの最大数を指定します。通常、多数の連結 AndX コマンドは異常な動作を表し、場合によっては回避試行を示している可能性があります。連結コマンドを許可しない場合は `1` を指定し、連結コマンドの数の検出を無効にするには `0` を指定します。

`dce_smb` インスペクタは最初に連結コマンドの数をカウントし、関連する SMB インスペクタのルールが有効であり、連結コマンドの数が設定されている値と等しいかそれ以上の場合にはイベントが生成されます。その後、処理が続行されます。

イベントを生成し、インライン展開でこのパラメータの問題のあるパケットをドロップするには、ルール 133:20 を有効にします。

型：整数

有効な範囲：`0` 〜 `255`

デフォルト値：`3`

disable_defrag

フラグメント化された DCE/RPC トラフィックを最適化するかどうかを指定します。有効にすると、`dce_smb` インスペクタは異常を検出して DCE/RPC データをルールエンジンに送信しますが、フラグメント化された DCE/RPC データでのエクスプロイトを見落とすリスクがあります。

`disable_defrag` は、トラフィックを最適化しない柔軟性を提供して処理を高速化しますが、ほとんどの DCE/RPC エクスプロイトは、エクスプロイトを隠蔽するフラグメント化を試行します。このパラメータを有効にすると、既知のエクスプロイトのほとんどがバイパスされ、誤検知が大量に発生します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

limit_alerts

DCE アラートをフローごとの署名ごとに最大 1 つに制限するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`true`

reassemble_threshold

リアセンブルされたパケットをルールエンジンに送信する前にキューに入れるには、DCE/RPC の最適化と最適化バッファの最小バイト数を指定します。このパラメータは、完全な最適化が実行される前の早い段階でエクスプロイトを検出する可能性があるため、インラインモードで役立ちます。

低い値を指定すると、早期検出の可能性が高くなりますが、パフォーマンスに悪影響を及ぼす可能性があることに注意してください。このパラメータを有効にした場合は、パフォーマンスへの影響をテストする必要があります。

値 `0` は、リアセンブルを無効にします。

型：整数

有効な範囲：`0` ～ `65535`

デフォルト値：`0`

policy

モニタ対象のネットワークセグメント上のターゲットホスト（複数可）が使用する Windows または Samba の DCE/RPC の実装を指定します。

型：列挙体

有効な値：`Win2000`、`WinXP`、`WinVista`、`Win2003`、`Win2008`、`Win7`、`Samba`、`Samba-3.0.37`、`Samba-3.0.22`、`Samba-3.0.20` から選択した文字列

デフォルト値：`WinXP`

smb_max_credit

未処理の要求の最大数を指定します。

型：整数

有効な範囲：`1` ～ `65536`

デフォルト値：`8192`

smb_file_depth

ファイルが SMB トラフィックで検出されたときに検査されるバイト数を指定します。これは、`file_data` IPS ルールオプションで指定した場所から始まります（https://www.snort.org/snort3 で入手可能な Snort 3 オープンソースドキュメントで説明されています）。

ファイルの検査を無効にするには、`-1` を指定します。

ファイルの検査を無制限に行うことを示すには、`0` を指定します。

型：整数

有効な範囲：`-1` ～ `32767`

デフォルト値：`16384`

SMB トラフィックでファイルが検出された場合、`smb_file_depth` パラメータは、`file_data` IPS ルールオプションで設定されたポインタから開始してインスペクタが確認するファイルデータのバイト数を示します。ファイルタイプと署名を検査するために、`dce_smb` は、`file_id` インスペクタで設定された `enable_type`、`type_depth`、`enable_signature`、および `signature_depth` のパラメータを使用します。`file_id` インスペクタの詳細については、https://www.snort.org/snort3 で入手可能な Snort オープンソースドキュメントを参照してください。

memcap

インスペクタに割り当てられる最大メモリ制限をバイト単位で指定します。最大メモリ キャップに達するか、または超過すると、dce_smb インスペクタは最近の使用頻度の最も低いデータを削除して、より多くの領域を生み出します。

型：整数

有効な範囲：512 ～ 9,007,199,254,740,992 (maxSZ)

デフォルト値：8,388,608

smb_fingerprint_policy

インスペクタが SMB Session Setup AndX の要求と応答で識別されている Windows バージョンまたは Samba のバージョンを検出するようにします。検出されたバージョンが policy インスペクタのパラメータに設定されている Windows バージョンまたは Samba バージョンと異なる場合、検出されたバージョンでそのセッションに設定されているバージョンのみがオーバーライドされます。

たとえば、policy を Windows XP に設定し、インスペクタが Windows Vista を検出した場合は、インスペクタはそのセッションに Windows Vista ポリシーを使用します。その他の設定は引き続き有効です。

型：列挙体

有効な値：none、client、server、または both

- サーバー/クライアント トラフィックでポリシータイプを検査するには、client を使用します。
- クライアント/サーバー トラフィックでポリシータイプを検査するには、server を使用します。
- サーバー/クライアントトラフィックとクライアント/サーバー トラフィックでポリシーを検索するには、both を使用します。
- Windows または Samba のバージョン検査を無効にするには、none を使用します。

デフォルト値：none

smb_legacy_mode

smb_legacy_mode が true の場合、システムは SMB バージョン 1 トラフィックにのみ SMB 侵入ルールを適用し、トランスポートとして SMB バージョン 1 を使用して DCE/RPC 侵入ルールを DCE/RPC トラフィックに適用します。

smb_legacy_mode が false の場合、システムは SMB バージョン 1 と 2 を使用して SMB 侵入ルールをトラフィックに適用し、次を実行します。

- バージョン 7.0 と 7.0.x の場合、システムは SMB バージョン 1 のみのトランスポートとして SMB を使用して、DCE/RPC 侵入ルールを DCE/RPC トラフィックに適用します。

- バージョン 7.1 以降の場合、システムは SMB バージョン 1 と 2 のトランスポートとして SMB を使用して、DCE/RPC 侵入ルールを DCE/RPC トラフィックに適用します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

valid_smb_versions

検査する SMB バージョンを指定します。複数の SMB バージョンは余白文字で区切ります。

型：文字列

有効な値：`v1`、`v2`、`v3`、`all`

デフォルト値：`all`

DCE SMB インスペクタのルール

`dce_smb` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *4 : DCE SMB* インスペクタのルール

GID:SID	ルール メッセージ
133:2	SMB：NetBIOS セッションのサービスセッションタイプが不適切（SMB - bad NetBIOS session service session type）
133:3	SMB：SMB メッセージタイプが不適切（SMB - bad SMB message type）
133:4	SMB：SMB ID が不適切（SMB1 に対して \xffSMB ではなく、SMB2 に対して \xfeSMB でない）（SMB - bad SMB Id (not \xffSMB for SMB1 or not \xfeSMB for SMB2)）
133:5	SMB：単語数または構造サイズが不適切（SMB - bad word count or structure size）
133:6	SMB：バイト数が不適切（SMB - bad byte count）
133:7	SMB：フォーマットタイプが不適切（SMB - bad format type）
133:8	SMB：オフセットが不適切（SMB - bad offset）
133:9	SMB：合計データ数がゼロになっている（SMB - zero total data count）
133:10	SMB：NetBIOS データ長が SMB ヘッダー長未満（SMB - NetBIOS data length less than SMB header length）

GID:SID	ルール メッセージ
133:11	SMB：残りの NetBIOS データ長がコマンド長未満（SMB - remaining NetBIOS data length less than command length）
133:12	SMB：残りの NetBIOS データ長がコマンドバイト数未満（SMB - remaining NetBIOS data length less than command byte count）
133:13	SMB：残りの NetBIOS データ長がコマンドデータサイズ未満（SMB - remaining NetBIOS data length less than command data size）
133:14	SMB：残りの合計データ数がこのコマンド データ サイズ未満（SMB - remaining total data count less than this command data size）
133:15	SMB： 送信された合計データ（STDu64）が予想されるコマンドの合計データよりも多い（SMB - total data sent (STDu64) greater than command total data expected）
133:16	SMB：バイト数がコマンドデータサイズ未満（STDu64）（SMB - byte count less than command data size (STDu64)）
133:17	SMB：バイト数のコマンドデータサイズが無効（SMB - invalid command data size for byte count）
133:18	SMB：保留中のツリー接続応答があるツリー接続要求が多すぎる（SMB - excessive tree connect requests with pending tree connect responses）
133:19	SMB：保留中の読み取り応答がある読み取り要求が多すぎる（SMB - excessive read requests with pending read responses）
133:20	SMB：コマンドチェーンが多すぎる（SMB - excessive command chaining）
133:21	SMB：チェーンログイン要求が複数ある（SMB - multiple chained login requests）
133:22	SMB：チェーンツリー接続要求が複数ある（SMB - multiple chained tree connect requests）
133:23	SMB：チェーン/複合ログインの後にログオフする（SMB - chained/compounded login followed by logoff）
133:24	SMB：チェーン/複合ツリー接続の後にツリーが切断される（SMB - chained/compounded tree connect followed by tree disconnect）
133:25	SMB：チェーン/複合オープンパイプ後のクローズパイプ（SMB - chained/compounded open pipe followed by close pipe）
133:26	SMB：無効な共有アクセス（SMB - invalid share access）

GID:SID	ルール メッセージ
133:44	SMB：無効な SMB バージョン 1 が表示された（SMB - invalid SMB version 1 seen）
133:45	SMB：無効な SMB バージョン 2 が表示された（SMB - invalid SMB version 2 seen）
133:46	SMB：ユーザー、ツリー接続、ファイルバインディングが無効（SMB - invalid user, tree connect, file binding）
133:47	SMB：コマンドの複合化が過度（SMB - excessive command compounding）
133:48	SMB：データ数がゼロ（SMB - zero data count）
133:50	SMB：未処理の要求の最大数を超えた（SMB - maximum number of outstanding requests exceeded）
133:51	SMB：未処理要求の MID が同じ（SMB - outstanding requests with same MID）
133:52	SMB：非推奨の言語がネゴシエートされた（SMB - deprecated dialect negotiated）
133:53	SMB：非推奨のコマンドが使用された（SMB - deprecated command used）
133:54	SMB：異常なコマンドが使用された（SMB - unusual command used）
133:55	SMB：コマンドの無効なセットアップ数（SMB - invalid setup count for command）
133:56	SMB：クライアントがセッションで複数言語のネゴシエーションを試みた（SMB - client attempted multiple dialect negotiations on session）
133:57	SMB：クライアントがファイルの属性を読み取り専用/非表示/システムに作成または設定しようとした（SMB - client attempted to create or set a file's attributes to readonly/hidden/system）
133:58	SMB：提供されたファイルオフセットが指定したファイルサイズを超えている（SMB - file offset provided is greater than file size specified）
133:59	SMB：SMB2 ヘッダーに指定した次のコマンドがペイロード境界を超えている（SMB - next command specified in SMB2 header is beyond payload boundary）

DCE インスペクタの侵入ルールのオプション

dce_iface

次のコンマ区切りの要素を指定します。

- サービスインターフェイスの UUID。
- インターフェイスのバージョン（オプション）。デフォルト設定は、どのバージョンにも一致します。
- ルールが要求内のいずれかのフラグメントに一致する必要があるかどうかのインジケータ（オプション）。デフォルト設定は、最初のフラグメントのみに一致します。

DCE/RPC プロトコルでは、クライアントはサービスを呼び出す前にサービスにバインドする必要があります。クライアントは、バインド要求をサーバーに送信するときに、バインド先の 1 つ以上のサービスインターフェイスを指定できます。各インターフェイスはUUIDで表され、各インターフェイスの UUID は一意のインデックス（またはコンテキスト ID）とペアになっており、このインデックスを使用して、クライアントが呼び出しているサービスを今後の要求で参照できます。サーバーは、有効なものとして受け入れるインターフェイス UUID で応答し、クライアントがそれらのサービスに要求を行うことを許可します。クライアントが要求を行うと、コンテキストIDが指定されるため、サーバーはクライアントが要求を行っているサービスを認識します。

`dce_iface` ルールオプションを使用すると、ルールは、クライアントが特定のインターフェイスUUIDにバインドされているかどうかと、このクライアント要求がそのインターフェイスへの要求を行っているかどうかをインスペクタに問い合わせることができます。これにより、インスペクタがバインド UUID を要求で使用されるコンテキスト ID に関連付けることができるため、複数のサービスが正常にバインドされている場合の誤検知を排除できます。

`dce_iface` オプションは、インスペクタでのコネクション型の DCE/RPC に対するサーバーの Bind Ack 応答と Alter Context 応答だけでなく、クライアントの Bind 要求と Alter Context 要求の追跡が必要です。BindおよびAlterContext要求ごとに、クライアントは、インターフェイスを参照するために DCE/RPC セッション中に使用される各インターフェイス UUID のハンドル（またはコンテキスト ID）とともに、インターフェイス UUID のリストを指定します。サーバー応答は、クライアントが要求を行うことを許可するインターフェイスを示します。これは、特定のインターフェイスにバインドするクライアントの要求を受け入れるか拒否します。この追跡は、要求が処理されるときに、要求で使用されるコンテキスト ID を、それがハンドルであるインターフェイス UUID と関連付けることができるようにするために必要です。

`dce_iface` ルールオプションは、次の場合に一致します。

- 指定したインターフェイス UUID が DCE/RPC 要求のインターフェイス UUID（コンテキスト ID によって参照される）と一致する

および

- version 引数が指定されていないか、または version 引数が指定されていて、DCE/RPC 要求のインターフェイス UUID と一致する

および

- any_frag 引数が指定されているか、any_frag 引数がなく、dce_iface オプションが最初の要求フラグメントの UUID とバージョン基準に一致する

例：

```
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188;
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188, <2;
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188,any_frag;
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188, =1, any_frag;
```

dce_iface.uuid

DCE/RPC 要求では、UUID 番号がビッグエンディアンまたはリトルエンディアンのどちらで表されるかを指定できます。要求でのインターフェイス UUID の表現は、要求で指定したエンディアンに応じて異なります。dce_rpc インスペクタは UUID を正規化します。つまり、dce_iface ルールオプションの UUID の指定は、要求のエンディアンに関係なく、同じように記述する必要があります。

たとえば、リトルエンディアンのバインド要求は、次のように UUID を表します。

```
|f8 91 7b 5a 00 ff d0 11 a9 b2 00 c0 4f b6 e6 fc|
```

ビッグエンディアンのバインド要求は、次のように同じ UUID を表します。

```
|5a 7b 91 f8 ff 00 11 d0 a9 b2 00 c0 4f b6 e6 fc|
```

dce_iface オプションを使用する Snort 3 ルールでは、要求のエンディアンに関係なく、ビッグエンディアン順を使用して UUID を文字列で表す必要があります。

```
5a7b91f8-ff00-11d0-a9b2-00c04fb6e6fc
```

型：文字列

シンタックス：dce_iface: <UUID>;

有効な値： *UUID* は 32 の 16 進数で、ハイフンで区切られた 5 つの グループ（xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式）で表示されます。

例： dce_iface: 5a7b91f8-ff00-11d0-a9b2-00c04fb6e6fc;

dce_iface.version

サービスインターフェイスには、それに関連付けられたバージョンがあります。インターフェイスの一部のバージョンは、特定のエクスプロイトに対して脆弱ではない場合があります。そのため、dce_iface オプションで 1 つ以上のバージョン番号を指定して、特定のエクスプロイトを確認する必要があるかどうかを特定できます。

型：間隔

シンタックス：`dce_iface: <range_operator><positive integer>;` または `dce_iface: <positive integer><range_operator><positive integer>;`

有効な値：1 つ以上の一連の正のバージョン番号と、範囲形式の表で指定されている `range_operator` の1 つ。

例：`dce_iface: =6;`

dce_iface.any_frag

DCE/RPC 要求は、1 つ以上のフラグメントに分割できます。DCE/RPC ヘッダーにフラグが設定されて現在のフラグメントが要求の最初のフラグメントか、中間のフラグメントか、または最後のフラグメントであるかを示します。DCE/RPC 要求内のデータの確認の多くは、DCE/RPC 要求が最初のフラグメント（または完全な要求）である場合にのみ関連します。したがって、最初のフラグメントに続くフラグメントには、DCE/RPC 要求のより深いデータが含まれます。たとえば、要求の最初の 5 バイト（長さフィールドなど）内のデータを検索するルールでは、最初のフラグメント以外のフラグメントで誤ったデータを検出します。後続のフラグメントの開始は、要求の開始からある程度の長さでオフセットされます。これは、フラグメント化された DCE/RPC トラフィックの誤検知の原因となる可能性があります。

そのため、デフォルトでは、DCE_RPC インスペクタは要求の最初のフラグメントのみを照合します。インスペクタが一致の要求内のすべてのフラグメントを調べるようにするには、`dce_iface` ルールオプションに `any_frag` を追加します。最適化された DCE/RPC 要求は完全な要求と見なされることに注意してください。

シンタックス：`dec_iface: any_frag;`

例：`dce_iface: any_frag;`

dce_opnum

DCE RPC 操作番号、操作番号の範囲、または操作番号のリストと照合します。このオプションは、インターフェイスに対して実行できる 1 つ以上の特定の関数呼び出しを表します。クライアントが特定のサービスインターフェイスにバインドして要求を行った後、ルールは、クライアントがサービスに対して行っている関数呼び出しを特定して、関数呼び出し内に存在する可能性のあるエクスプロイトを確認する必要があります。関数呼び出しは、操作番号（opnums）の二重引用符で囲まれたリストとして指定されます。

型：文字列

シンタックス：`dce_opnum: <opnum_list>;`

`opnum_list` は次のいずれかです。

- 単一の整数。
- コンマ区切りの整数のリスト。
- 範囲内の最小数と最大数をハイフンで区切って指定した整数の範囲。
- 上記の組み合わせ。

有効な値：DCE/RPC 要求の opnum のリスト。

例：

```
dce_opnum: "15";
dce_opnum: "15-18";
dce_opnum: "15, 18-20";
dce_opnum: "15, 17, 20-22";
```

dce_stub_data

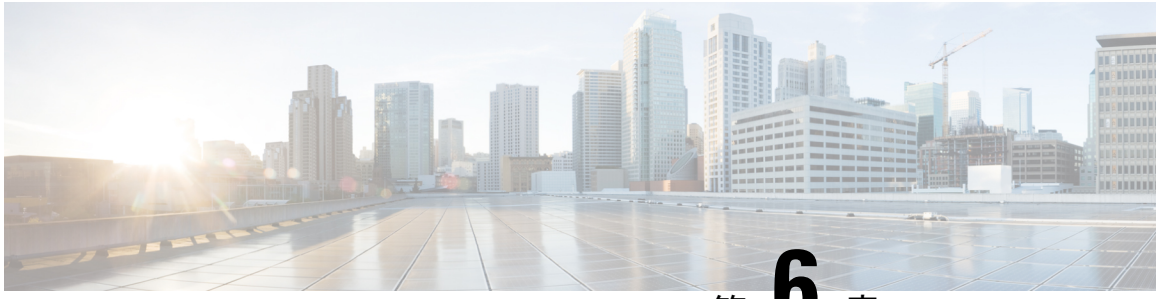
このオプションは、先行するルールオプションに関係なく、DCE/RPC スタブデータの先頭に検出カーソル（ルール処理でパケットペイロードを通過させるために使用）を配置します。このオプションは、DCE/RPC スタブデータが存在する場合に一致します。

シンタックス：`dce_stub_data;`

例：`dce_stub_data;`

表 *5:* 範囲の形式

範囲の形式	演算子	説明
operator i		
	<	より少ない
	>	右辺と比較して大きい
	=	等しい
	≠	等しくない
	≤	以下
	≥	以上
j operator k		
	<>	j よりも大きく、k よりも小さい
	<=>	j 以上で k 以下



第 6 章

DCE TCP インスペクタ

- DCE TCP インスペクタの概要 （43 ページ）
- DCE TCP インスペクタのパラメータ （45 ページ）
- DCE TCP インスペクタのルール （47 ページ）
- DCE インスペクタの侵入ルールのオプション （48 ページ）

DCE TCP インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	なし
有効	`true`

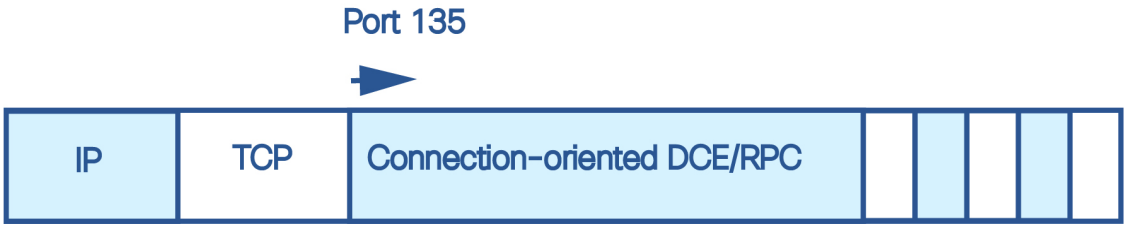
DCE/RPC プロトコルにより、別々のネットワーク ホスト上のプロセスが、同一ホストに配置されている場合と同様に通信できます。通常、このようなプロセス間通信はホスト間で TCP を介して転送されます。TCP 転送では、DCE/RPC が Windows Server Message Block（SMB）プロトコルまたは Samba でさらにカプセル化されることがあります。Samba は、Windows や UNIX または Linux のオペレーティングシステムから構成される混合環境でプロセス間通信に使用されるオープンソースの SMB 実装です。

ほとんどの DCE/RPC エクスプロイトは、DCE/RPC サーバ（ネットワーク上の Windows または Samba が稼働している任意のホスト）を対象とした DCE/RPC クライアント要求で発生します。またエクスプロイトはサーバ応答でも発生することがあります。

IP によりすべての DCE/RPC トランスポートがカプセル化されます。TCP は、すべてのコネクション型 DCE/RPC を伝送します。DCE TCP インスペクタは、コネクション型の DCE/RPC を検出し、プロトコル固有の特性（ヘッダー長やデータ フラグメントの順序など）を使用して次のことを行います。

- TCP トランスポートでカプセル化された DCE/RPC 要求を検出します。これには、RPC over HTTP バージョン 1 を使用して TCP で転送される DCE/RPC も含まれます。
- DCE/RPC データストリームを分析し、DCE/RPC トラフィック内の異常な動作と回避技術を検出します。
- DCE/RPC を最適化します。
- ルールエンジンで処理できるように DCE/RPC トラフィックを正規化します。

次の図に、DCE TCP インスペクタが TCP トランスポートのトラフィックの処理を開始するポイントを示します。



ウェルノウン TCP ポート 135 は、TCP トランスポートの DCE/RPC トラフィックを特定します。この図には RPC over HTTP は含まれていません。RPC over HTTP の場合、コネクション型 DCE/RPC は、図に示すように、HTTP を介した初期設定シーケンスの後、TCP 経由で直接伝送されます。

ターゲットベースのポリシー

Windows および Samba の DCE/RPC の実装は大きく異なります。たとえば、Windows のすべてのバージョンは、DCE/RPC トラフィックの最適化時に最初のフラグメントの DCE/RPC コンテキスト ID を使用しますが、Samba のすべてのバージョンは、最後のフラグメントのコンテキスト ID を使用します。また、特定の関数呼び出しを識別するために、Windows Vista では最初のフラグメントの opnum（操作番号）ヘッダーフィールドを使用しますが、Samba とその他のすべてのバージョンの Windows では最後のフラグメントの opnum フィールドを使用します。

そのため、`dce_tcp` インスペクタはターゲットベースのアプローチを使用します。`dce_tcp` インスペクタインスタンスを設定すると、`policy` パラメータは特定の DCE/RPC TCP プロトコルの実装を指定します。これをホスト情報と組み合わせることで、デフォルトのターゲットベースのサーバーポリシーが確立されます。必要に応じて、他のホストおよび DCE/RPC TCP の実装を対象とする追加のインスペクタを設定できます。デフォルトのターゲットベースのサーバーポリシーで指定されている DCE/RPC TCP の実装は、別の `dce_tcp` インスペクタインスタンスの対象になっていないすべてのホストに適用されます。

DCE TCP インスペクタが `policy` パラメータを使用してターゲットにできる DCE/RPC 実装は次のとおりです。

- WinXP（デフォルト）
- Win2000
- WinVista

- Win2003
- Win2008
- Win7
- Samba
- Samba-3.0.37
- Samba-3.0.22
- Samba-3.0.20

最適化

DCE TCP インスペクタでは、フラグメント化されたデータパケットを検出エンジンに送信する前に再設定することができます。この機能は、インラインモードで、完全な最適化が実行される前の検査プロセスの早い段階でエクスプロイトをキャッチしたり、フラグメント化を利用してそれ自体を隠すエクスプロイトをキャッチしたりするのに役立ちます。最適化を無効にすると、多数の誤検知が発生する可能性があることに注意してください。

DCE TCP インスペクタのパラメータ

DCE TCP ポートの設定

`binder` インスペクタは、DCE TCP ポートの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
   {
        "when": {
            "role": "any",
            "proto": "tcp",
            "service": "dcerpc",
            "ports": ""
        },
        "use": {
            "type": "dce_tcp"
        }
    }
]
```

max_frag_len

最適化のために許可される最大フラグメント長をバイト単位で指定します。より大きなフラグメントを処理する場合、インスペクタは、最適化する前にパケットコンテンツに対して指定されたサイズに考慮します。

（注） このパラメータで指定する値は、確実に検出するためにルールでデータを調べる必要がある深さ以上にする必要があります。すべてのデータの検出が確実に行われるようにするには、デフォルト値を使用します。

型：整数

有効な範囲：1514 ～ 65535

デフォルト値：65535

disable_defrag

フラグメント化された DCE/RPC トラフィックを最適化するかどうかを指定します。このパラメータが true の場合、インスペクタは引き続き異常を検出して DCE/RPC データをルールエンジンに送信しますが、フラグメント化された DCE/RPC データでのエクスプロイトを見落とすリスクがあります。

このパラメータには、トラフィックを最適化しないという柔軟性があり、処理を高速化できますが、ほとんどの DCE/RPC エクスプロイトでは、フラグメント化を利用してエクスプロイトを隠ぺいする試行が行われます。このパラメータを有効にすると、既知のエクスプロイトのほとんどがバイパスされ、誤検知が大量に発生します。

型：ブール値

有効な値：true、false

デフォルト値：false

limit_alerts

DCE アラートをフローごとの署名ごとに最大 1 つに制限するかどうかを指定します。

型：ブール値

有効な値：true、false

デフォルト値：true

reassemble_threshold

リアセンブルされたパケットをルールエンジンに送信する前にキューに入れるには、DCE/RPC の最適化と最適化バッファの最小バイト数を指定します。このパラメータは、完全な最適化が実行される前の早い段階でエクスプロイトを検出する可能性があるため、インラインモードで役立ちます。

低い値を指定すると、早期検出の可能性が高くなりますが、パフォーマンスに悪影響を及ぼす可能性があります。このパラメータを有効にした場合は、パフォーマンスへの影響をテストする必要があります。

値 0 は、リアセンブルを無効にします。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：0

policy

モニタ対象のネットワークセグメント上のターゲットホスト（複数可）が使用する Windows または Samba の DCE/RPC の実装を指定します。

型：列挙体

有効な値：Win2000、WinXP、WinVista、Win2003、Win2008、Win7、Samba、Samba-3.0.37、Samba-3.0.22、Samba-3.0.20 から選択した文字列

デフォルト値：WinXP

DCE TCP インスペクタのルール

dce_tcp インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *6 : DCE TCP* インスペクタのルール

GID:SID	ルール メッセージ
133:27	コネクション型 DCE/RPC：無効なメジャーバージョン（connection oriented DCE/RPC - invalid major version）
133:28	コネクション型 DCE/RPC：無効なマイナーバージョン（connection oriented DCE/RPC - invalid minor version）
133:29	コネクション型 DCE/RPC：無効な PDU タイプ（connection-oriented DCE/RPC - invalid PDU type）
133:30	コネクション型 DCE/RPC：フラグメント長がヘッダーサイズ未満（connection-oriented DCE/RPC - fragment length less than header size）
133:31	コネクション型 DCE/RPC：残りのフラグメント帳が必要なサイズ未満（connection-oriented DCE/RPC - remaining fragment length less than size needed）
133:32	コネクション型 DCE/RPC：コンテキスト項目が指定されていない（connection-oriented DCE/RPC - no context items specified）
133:33	コネクション型 DCE/RPC：転送シンタックスが指定されていない（connection-oriented DCE/RPC -no transfer syntaxes specified）

GID:SID	ルール メッセージ
133:34	コネクション型 DCE/RPC：最後のフラグメント以外のフラグメント長がクライアントのネゴシエートされた最大フラグメント送信サイズ未満（connection-oriented DCE/RPC - fragment length on non-last fragment less than maximum negotiated fragment transmit size for client）
133:35	コネクション型 DCE/RPC：フラグメント長がネゴシエートされた最大フラグメント送信サイズを超えている（connection-oriented DCE/RPC - fragment length greater than maximum negotiated fragment transmit size）
133:36	コネクション型 DCE/RPC：バインドとは異なるコンテキストのバイト順序を変更する（connection-oriented DCE/RPC - alter context byte order different from bind）
133:37	コネクション型 DCE/RPC：最初/最後のフラグメント以外の呼び出し ID がフラグメント化された要求に対して確立された呼び出し ID と異なる（call id of non first/last fragment different from call id established for fragmented request）
133:38	コネクション型 DCE/RPC：最初/最後のフラグメント以外の opnum がフラグメント化された要求に対して確立された opnum とは異なる（connection-oriented DCE/RPC - opnum of non first/last fragment different from opnum established for fragmented request）
133:39	コネクション型 DCE/RPC：最初/最後のフラグメント以外のコンテキスト ID がフラグメント化された要求に対して確立されたコンテキスト ID とは異なる（connection-oriented DCE/RPC - context id of non first/last fragment different from context id established for fragmented request）

DCE インスペクタの侵入ルールのオプション

dce_iface

次のコンマ区切りの要素を指定します。

- サービスインターフェイスの UUID。
- インターフェイスのバージョン（オプション）。デフォルト設定は、どのバージョンにも一致します。
- ルールが要求内のいずれかのフラグメントに一致する必要があるかどうかのインジケータ（オプション）。デフォルト設定は、最初のフラグメントのみに一致します。

DCE/RPC プロトコルでは、クライアントはサービスを呼び出す前にサービスにバインドする必要があります。クライアントは、バインド要求をサーバーに送信するときに、バインド先の1つ以上のサービスインターフェイスを指定できます。各インターフェイスはUUIDで表され、

各インターフェイスの UUID は一意のインデックス（またはコンテキスト ID）とペアになっており、このインデックスを使用して、クライアントが呼び出しているサービスを今後の要求で参照できます。サーバーは、有効なものとして受け入れるインターフェイス UUID で応答し、クライアントがそれらのサービスに要求を行うことを許可します。クライアントが要求を行うと、コンテキスト ID が指定されるため、サーバーはクライアントが要求を行っているサービスを認識します。

`dce_iface` ルールオプションを使用すると、ルールは、クライアントが特定のインターフェイス UUID にバインドされているかどうかと、このクライアント要求がそのインターフェイスへの要求を行っているかどうかをインスペクタに問い合わせることができます。これにより、インスペクタがバインド UUID を要求で使用されるコンテキスト ID に関連付けることができるため、複数のサービスが正常にバインドされている場合の誤検知を排除できます。

`dce_iface` オプションは、インスペクタでのコネクション型の DCE/RPC に対するサーバーの Bind Ack 応答と Alter Context 応答だけでなく、クライアントの Bind 要求と Alter Context 要求の追跡が必要です。Bind および Alter Context 要求ごとに、クライアントは、インターフェイスを参照するために DCE/RPC セッション中に使用される各インターフェイス UUID のハンドル（またはコンテキスト ID）とともに、インターフェイス UUID のリストを指定します。サーバー応答は、クライアントが要求を行うことを許可するインターフェイスを示します。これは、特定のインターフェイスにバインドするクライアントの要求を受け入れるか拒否します。この追跡は、要求が処理されるときに、要求で使用されるコンテキスト ID を、それがハンドルであるインターフェイス UUID と関連付けることができるようにするために必要です。

`dce_iface` ルールオプションは、次の場合に一致します。

- 指定したインターフェイス UUID が DCE/RPC 要求のインターフェイス UUID（コンテキスト ID によって参照される）と一致する

および

- `version` 引数が指定されていないか、または `version` 引数が指定されていて、DCE/RPC 要求のインターフェイス UUID と一致する

および

- `any_frag` 引数が指定されているか、`any_frag` 引数がなく、`dce_iface` オプションが最初の要求フラグメントの UUID とバージョン基準に一致する

例：

```
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188;
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188, <2;
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188,any_frag;
dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188, =1, any_frag;
```

dce_iface.uuid

DCE/RPC 要求では、UUID 番号がビッグエンディアンまたはリトルエンディアンのどちらで表されるかを指定できます。要求でのインターフェイス UUID の表現は、要求で指定したエンディアンに応じて異なります。`dce_rpc` インスペクタは UUID を正規化します。つまり、

dce_iface ルールオプションの UUID の指定は、要求のエンディアンに関係なく、同じように記述する必要があります。

たとえば、リトルエンディアンのバインド要求は、次のように UUID を表します。

```
|f8 91 7b 5a 00 ff d0 11 a9 b2 00 c0 4f b6 e6 fc|
```

ビッグエンディアンのバインド要求は、次のように同じ UUID を表します。

```
|5a 7b 91 f8 ff 00 11 d0 a9 b2 00 c0 4f b6 e6 fc|
```

dce_iface オプションを使用する Snort 3 ルールでは、要求のエンディアンに関係なく、ビッグエンディアン順を使用して UUID を文字列で表す必要があります。

```
5a7b91f8-ff00-11d0-a9b2-00c04fb6e6fc
```

型：文字列

シンタックス：`dce_iface: <UUID>;`

有効な値：*UUID* は 32 の 16 進数で、ハイフンで区切られた 5 つの グループ（xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式）で表示されます。

例：`dce_iface: 5a7b91f8-ff00-11d0-a9b2-00c04fb6e6fc;`

dce_iface.version

サービスインターフェイスには、それに関連付けられたバージョンがあります。インターフェイスの一部のバージョンは、特定のエクスプロイトに対して脆弱ではない場合があります。そのため、dce_iface オプションで 1 つ以上のバージョン番号を指定して、特定のエクスプロイトを確認する必要があるかどうかを特定できます。

型：間隔

シンタックス：`dce_iface: <range_operator><positive integer>;` または `dce_iface: <positive integer><range_operator><positive integer>;`

有効な値：1 つ以上の一連の正のバージョン番号と、範囲形式の表で指定されている range_operator の1 つ。

例：`dce_iface: =6;`

dce_iface.any_frag

DCE/RPC 要求は、1 つ以上のフラグメントに分割できます。DCE/RPC ヘッダーにフラグが設定されて現在のフラグメントが要求の最初のフラグメントか、中間のフラグメントか、または最後のフラグメントであるかを示します。DCE/RPC 要求内のデータの確認の多くは、DCE/RPC 要求が最初のフラグメント（または完全な要求）である場合にのみ関連します。したがって、最初のフラグメントに続くフラグメントには、DCE/RPC 要求のより深いデータが含まれます。たとえば、要求の最初の 5 バイト（長さフィールドなど）内のデータを検索するルールでは、最初のフラグメント以外のフラグメントで誤ったデータを検出します。後続のフラグメントの開始は、要求の開始からある程度の長さでオフセットされます。これは、フラグメント化された DCE/RPC トラフィックの誤検知の原因となる可能性があります。

そのため、デフォルトでは、DCE_RPC インスペクタは要求の最初のフラグメントのみを照合します。インスペクタが一致の要求内のすべてのフラグメントを調べるようにするには、dce_iface ルールオプションに any_frag を追加します。最適化された DCE/RPC 要求は完全な要求と見なされることに注意してください。

シンタックス：dec_iface: any_frag;

例：dce_iface: any_frag;

dce_opnum

DCE RPC 操作番号、操作番号の範囲、または操作番号のリストと照合します。このオプションは、インターフェイスに対して実行できる1つ以上の特定の関数呼び出しを表します。クライアントが特定のサービスインターフェイスにバインドして要求を行った後、ルールは、クライアントがサービスに対して行っている関数呼び出しを特定して、関数呼び出し内に存在する可能性のあるエクスプロイトを確認する必要があります。関数呼び出しは、操作番号（opnums）の二重引用符で囲まれたリストとして指定されます。

型：文字列

シンタックス：dce_opnum: <opnum_list>;

opnum_list は次のいずれかです。

- 単一の整数。
- コンマ区切りの整数のリスト。
- 範囲内の最小数と最大数をハイフンで区切って指定した整数の範囲。
- 上記の組み合わせ。

有効な値：DCE/RPC 要求の opnum のリスト。

例：

```
dce_opnum: "15";
dce_opnum: "15-18";
dce_opnum: "15, 18-20";
dce_opnum: "15, 17, 20-22";
```

dce_stub_data

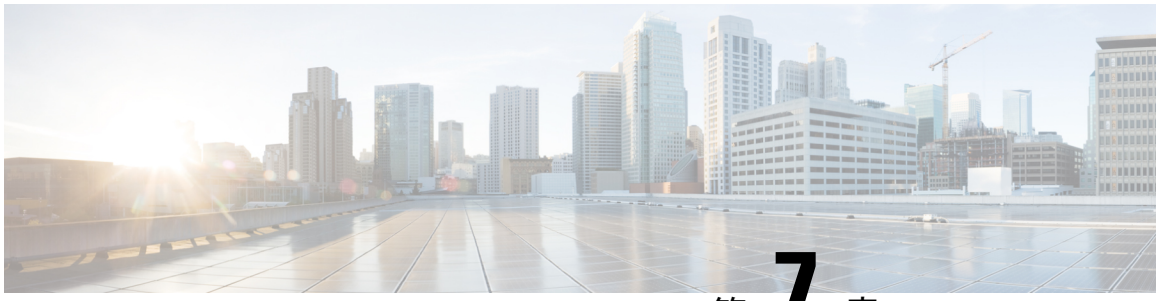
このオプションは、先行するルールオプションに関係なく、DCE/RPC スタブデータの先頭に検出カーソル（ルール処理でパケットペイロードを通過させるために使用）を配置します。このオプションは、DCE/RPC スタブデータが存在する場合に一致します。

シンタックス：dce_stub_data;

例：dce_stub_data;

表 *7:* 範囲の形式

範囲の形式	演算子	説明
operator i		
	<	より少ない
	>	右辺と比較して大きい
	=	等しい
	≠	等しくない
	≤	以下
	≥	以上
j operator k		
	<>	j よりも大きく、k よりも小さい
	<=>	j 以上で k 以下



第 7 章

DNP3 インスペクタ

- DNP3 インスペクタの概要 （53 ページ）
- DNP3 インスペクタのパラメータ （53 ページ）
- DNP3 インスペクタのルール （54 ページ）
- DNP3 インスペクタの侵入ルールのオプション （55 ページ）

DNP3 インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`、`stream_udp`
有効	`false`

Distributed Network Protocol（DNP3）は遠隔監視制御・情報取得（SCADA）プロトコルであり、当初は発電所間で一貫性のある通信を実現する目的で開発されました。DNP3 は水処理、廃棄物処理、輸送などの産業分野で幅広く使用されています。

`dnp3` インスペクタは、DNP3 トラフィックの異常を検出し、DNP3 プロトコルを分析します。`dnp3` 侵入ルールのオプションは、特定の DNP3 プロトコルフィールドにアクセスします。

DNP3 インスペクタのパラメータ

DNP3 TCP ポートの設定

`binder` インスペクタは、DNP3 TCP ポートの設定を定義します。詳細については、『バインダ インスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
   {
       "when": {
           "role": "any",
           "service": "dnp3"
    },
       "use": {
           "type": "dnp3"
       }
    }
]
```

check_crc

DNP3 リンク層フレームに含まれているチェックサムを検証するかどうかを指定します。dnp3 インスペクタは、チェックサムが無効なフレームを無視します。侵入ルール 145:1 が有効になっている場合、Snort は無効なチェックサムに対してアラートを生成します。

型：ブール値

有効な値：true、false

デフォルト値：false

DNP3 インスペクタのルール

dnp3インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *8 : DNP3* インスペクタのルール

GID:SID	ルール メッセージ
145:1	DNP3 リンク層フレームに不良 CRC が含まれている（DNP3 link-layer frame contains bad CRC）
145:2	DNP3 リンク層フレームがドロップされた（DNP3 link-layer frame was dropped）
145:3	DNP3 トランスポート層セグメントがリアセンブル中に削除された（DNP3 transport-layer segment was dropped during reassembly）
145:4	DNP3 リアセンブルバッファが、完全なメッセージをリアセンブルせずにクリアされた（DNP3 reassembly buffer was cleared without reassembling a complete message）
145:5	DNP3 リンク層フレームは予約済みアドレスを使用する（DNP3 link-layer frame uses a reserved address）
145:6	DNP3 アプリケーション層フラグメントは予約済み関数コードを使用する（DNP3 application-layer fragment uses a reserved function code）

DNP3 インスペクタの侵入ルールのオプション

dnp3_data

dnp3_data キーワードは、先行するルールのオプションに関係なく、アプリケーション層フラグメント内の DNP3 データの先頭に検出カーソルを配置します。このオプションを使用すると、データを分割して 16 バイトごとに CRC を追加することなく、フラグメント内のデータに基づいてルールを作成できます。

シンタックス：dnp3_data;

例：dnp3_data;

dnp3_func

このオプションは、DNP3 アプリケーション層の要求/応答のヘッダー内の関数コードと照合されます。コードは、以下のリストの 10 進数または文字列です。

型：文字列

シンタックス：dnp3_func: <DNP3_function>;

有効な値：*DNP3_function* は次のいずれかです。

- 0 ～ 255 の整数
- confirm（機能コード 0 に対応）
- read（機能コード 1 に対応）
- write（機能コード 2 に対応）
- select（機能コード 3 に対応）
- operate（機能コード 4 に対応）
- direct_operate（機能コード 5 に対応）
- direct_operat_nr（機能コード 6 に対応）
- immed_freeze（機能コード 7 に対応）
- immed_freeze_nr（機能コード 8 に対応）
- freeze_clear（機能コード 9 に対応）
- freeze_clear_nr（機能コード 10 に対応）
- freeze_at_time（機能コード 11 に対応）
- freeze_at_time_nr（機能コード 12 に対応）
- cold_restart（機能コード 13 に対応）

- warm_restart（機能コード 14 に対応）
- initialize_data（機能コード 15 に対応）
- initialize_appl（機能コード 16 に対応）
- initialize_appl（機能コード 17 に対応）
- initialize_appl（機能コード 18 に対応）
- save_config（機能コード 19 に対応）
- enable_unsolicited（機能コード 20 に対応）
- save_config（機能コード 21 に対応）
- assign_class（機能コード 22 に対応）
- assign_class（機能コード 23 に対応）
- record_current_time（機能コード 24 に対応）
- open_file（機能コード 25 に対応）
- close_file（機能コード 26 に対応）
- delete_file（機能コード 27 に対応）
- get_file_info（機能コード 28 に対応）
- authenticate_file（機能コード 29 に対応）
- abort_file（機能コード 30 に対応）
- activate_config（機能コード 31 に対応）
- authenticate_req（機能コード 32 に対応）
- authenticate_err（機能コード 33 に対応）
- response（機能コード 129 に対応）
- unsolicited_response（機能コード 130 に対応）
- authenticate_resp（機能コード 131 に対応）

例：

```
dnp3_func: 1;
dnp3_func: delete_file;
```

dnp3_ind

DNP3アプリケーション層の応答ヘッダーの内部インジケータフラグと照合する内部インジケータフラグのリストを表示します。1 つのオプションに複数のフラグを指定すると、いずれかの

フラグが設定されている場合にルールが起動します。複数のフラグにアラートを発行するには、複数のルールのオプションを使用します。

型：文字列

シンタックス：`dnp3_ind: "<flag> <flag>";`

有効な値：1 つ以上の DNP3 内部インジケータフラグ。ここで、`flag` は次のいずれかです。

- `all_stations`
- `class_1_events`
- `class_2_events`
- `class_3_events`
- `need_time`
- `local_control`
- `device_trouble`
- `device_restart`
- `no_func_code_support`
- `object_unknown`
- `parameter_error`
- `event_buffer_overflow`
- `already_executing`
- `config_corrupt`
- `reserved_2`
- `reserved_1`

例：

デバイスの再起動時または時刻同期の開始時のアラート：

```
dnp3_ind:"device_restart need_time";
```

class_1 イベントと class_2 AND class_3 イベントに関するアラート：

```
dnp3_ind:class_1_events; dnp3_ind:class_2_events; dnp3_ind:class_3_events;
```

dnp3_obj

DNP3 オブジェクト ヘッダー グループとバリエーションで照合します。

型：整数

構文： `dnp3_obj:<groupnum>,<varnum>;`

有効な値：DNP3 オブジェクトグループ ID とバリエーション ID。

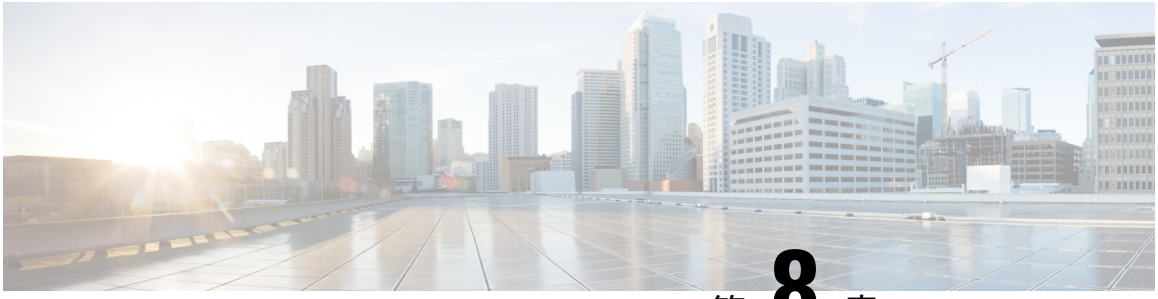
- *`groupnum`* は DNP3 オブジェクトグループ指定する `0` 〜 `255` の整数です。

- *varnum* はオブジェクトグループ内のバリエーションを指定する 0 ～ 255 の整数です。

例：

DNP3 `Date and Time` オブジェクトのアラート：

```
dnp3_obj:50,1;
```



第 8 章

FTP クライアントインスペクタ

- FTP クライアントインスペクタの概要 （59 ページ）
- FTP クライアントインスペクタのパラメータ （60 ページ）
- FTP クライアントインスペクタのルール （61 ページ）
- FTP クライアントインスペクタの侵入ルールのオプション （61 ページ）

FTP クライアントインスペクタの概要

タイプ	インスペクタ（パッシブ）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`ftp_server`、`stream_tcp`
有効	`true`

File Transfer Protocol（FTP）は、TCP/IP を介してクライアントとサーバー間でファイルを転送するために使用されるネットワークプロトコルです。クライアントとサーバーが接続を確立すると、クライアントはサーバーにコマンドを発行してファイルをサーバーにアップロードするか、またはサーバーからファイルをダウンロードし、サーバーからの応答を解釈します。

`ftp_client` インスペクタは、FTP コマンドチャネルの応答を確認して正規化します。

FTP コマンドチャネルバッファを指定すると、`ftp_client` インスペクタは FTP 応答コードとメッセージを解釈します。`ftp_client` インスペクタは、パラメータの正確性を適用し、FTP コマンド接続がいつ暗号化され、いつ FTP データチャネルが開かれるかを決定します。

FTP クライアントインスペクタのパラメータ

bounce

クライアントが発行した `ftp port` コマンド内のホスト情報を確認して FTP バウンスを確認するかどうかを指定します。`bounce` が `true` に設定されている場合、`ftp port` コマンド内のホスト情報が設定されたクライアント IP アドレス、またはホスト情報と一致しておらず、ルール 125:8 が有効になっているときは、システムがアラートを生成し、インライン展開で問題のあるパケットをドロップします。これは、FTP バウンス攻撃を防ぎ、FTP データ チャネルの接続先がクライアントとは異なる FTP 接続を許可するために使用できます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

ignore_telnet_erase_cmds

FTP コマンドチャネルを正規化するときに、消去文字（TNC EAC）と消去行文字（TNC EAL）の Telnet エスケープシーケンスを無視するかどうかを指定します。このパラメータは、FTP クライアントが Telnet 消去コマンドを処理する方法と一致するように設定する必要があります。通常、新しい FTP クライアントはこれらの Telnet エスケープシーケンスを無視しますが、レガシークライアントは通常、それらを処理します。`ignore_telnet_erase_cmds` パラメータが `false` の場合、インスペクタはルール 125:1 を使用してアラートを生成し、インライン展開で問題のあるパケットをドロップします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

max_resp_len

クライアントが受け入れるすべての応答メッセージの最大長をバイト単位で指定します。FTP 応答のメッセージ（3 桁のリターンコードの後のすべて）がその長さを超え、ルール 125:6 が有効になっている場合、システムはアラートを生成し、インライン展開で問題のあるパケットをドロップします。これは、FTP クライアント内のバッファのオーバーフローのエクスプロイトを確認するために使用されます。

型：整数

有効な範囲：`0 〜 4,294,967,295 (max32)`

デフォルト値：`4,294,967,295`

telnet_cmds

FTP コマンドチャネルで Telnet コマンドを確認するかどうかを指定します。このようなコマンドがある場合は、FTP コマンドチャネルでの回避試行を示している可能性があります。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 125:1 を有効にします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

FTP クライアントインスペクタのルール

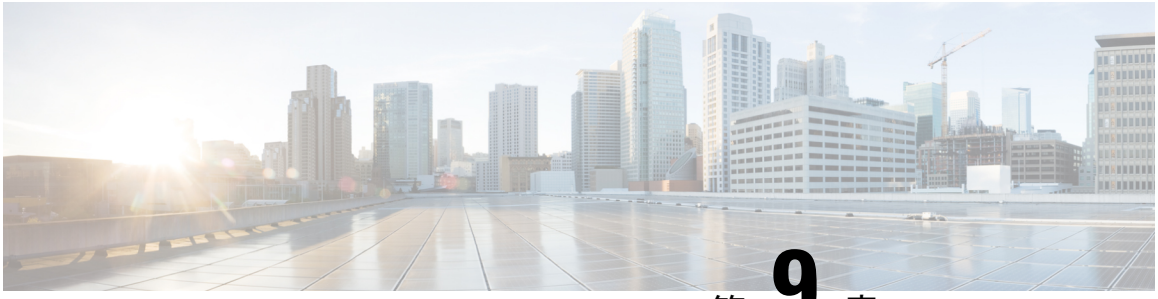
`ftp_client` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *9 : FTP* クライアントインスペクタのルール

GID:SID	ルール メッセージ
125:1	FTP コマンドチャネルの Telnet コマンド（TELNET cmd on FTP command channel）
125:6	FTP 応答メッセージが長すぎる（FTP response message was too long）
125:8	FTP バウンス試行（FTP bounce attempt）

FTP クライアントインスペクタの侵入ルールのオプション

`ftp_client` インスペクタには、侵入ルールのオプションはありません。



第 9 章

FTP サーバーインスペクタ

- FTP サーバーインスペクタの概要 （63 ページ）
- FTP サーバーインスペクタのパラメータ （64 ページ）
- FTP サーバーインスペクタのルール （69 ページ）
- FTP サーバーインスペクタの侵入ルールのオプション （70 ページ）

FTP サーバーインスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`ftp_client`、`stream_tcp`
有効	`true`

File Transfer Protocol（FTP）は、TCP/IP を介してクライアントとサーバー間でファイルを転送するために使用されるネットワークプロトコルです。クライアントとサーバーが接続を確立すると、クライアントはサーバーにコマンドを発行してファイルをサーバーにアップロードするか、またはサーバーからファイルをダウンロードし、サーバーからの応答を解釈します。

`ftp_server` インスペクタは、FTP コマンドチャネル上のコマンドを確認して正規化します。

FTP コマンドチャネルバッファを指定すると、`ftp_server` インスペクタは FTP コマンドとパラメータを識別し、パラメータの正確性を適用します。`ftp_server` は、FTP コマンド接続がいつ暗号化され、FTP データチャネルがいつ開かれるかを決定します。

FTP サーバーインスペクタのパラメータ

FTP サーバーポートの設定

binder インスペクタは、FTP サーバーの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "role":"any",
            "service":"ftp",
            "ports": ""
        },
        "use": {
            "type":"ftp_server"
        }
    }
]
```

chk_str_fmt

書式文字列攻撃を確認する FTP コマンドのリストを指定します。ルール 125:5 を有効にしてアラートを生成し、インライン展開で、インスペクタがこの状態を検出したときに問題のあるパケットをドロップすることができます。複数のコマンドは余白文字で区切ります。

型：文字列

有効な値：有効な FTP コマンドのリスト。

デフォルト値：なし

data_chan_cmds

正しい書式設定を確認する FTP コマンドのリストを指定します。複数のコマンドは余白文字で区切ります。

型：文字列

有効な値：PORT PASV LPRT LPSV EPRT EPSV の 1 つ以上のコマンドのリスト。

デフォルト値：なし

data_xfer_cmds

データ転送コマンドのリストを指定します。コマンドの正しい書式設定を確認します。複数のコマンドは余白文字で区切ります。

型：文字列

有効な値：RETR STOR STOU APPE LIST NLST の 1 つ以上のコマンドのリスト。

デフォルト値：なし

file_put_cmds

PUT コマンドのリストを指定します。コマンドの正しい書式設定を確認します。複数のコマンドは余白文字で区切ります。

型：文字列

有効な値：STOR STOU APPE の 1 つ以上のコマンドのリスト。

デフォルト値：なし

⚠

注意 サポートからの指示がない限り、file_put_cmds パラメータは変更しないでください。

file_get_cmds

GET コマンドのリストを指定します。コマンドの正しい書式設定を確認します。複数のコマンドは余白文字で区切ります。

型：文字列

有効な値：GET コマンド（RETR など）のリスト。

デフォルト値：なし

⚠

注意 サポートからの指示がない限り、file_get_cmds パラメータは変更しないでください。

encr_cmds

セキュア接続に関連するコマンドのリストを指定します。コマンドの正しい書式設定を確認します。複数のコマンドは余白文字で区切ります。

型：文字列

有効な値：セキュア接続に関連するコマンド（AUTH など）のリスト。

デフォルト値：なし

login_cmds

ログインプロセスに関連するコマンドのリストを指定します。コマンドの正しい書式設定を確認します。複数のコマンドは余白文字で区切ります。

型：文字列

有効な値：USER、PASS など 1 つ以上のコマンドのリストを指定します。

デフォルト値：なし

check_encrypted

暗暗号化されたセッションで暗号化を終了するコマンドを確認するかどうかを指定します。`encrypted_traffic` パラメータとともに使用します。

このパラメータに対してルール 125:7 を有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

cmd_validity[]

FTP コマンドの配列と、インスペクタがそれらを検証するために使用する基準。これらの妥当性検査は、`ftp_server` インスペクタ（RFC 959）によって実行されるデフォルトの検査をオーバーライドします。

イベントを生成し、ライン展開ではこのパラメータの問題のあるパケットをドロップするには、ルール 125:2 と 125:4 を有効にします。

型：配列（オブジェクト）

例：

```
{
   "cmd_validity": [
       {
           "command": "CWD",
           "format": "abc",
           "length": 250
       }
   ]
}
```

cmd_validity[].command

検証する FTP コマンドの名前を指定します。

型：文字列

有効な値：二重引用符で囲まれた有効な FTP コマンド。

デフォルト値：なし

cmd_validity[].format

`cmd_validity[].command` の有効な形式について説明します。

型：文字列

有効な値：次のいずれかの形式。

- `int`：このパラメータは整数にする必要があります。
- `number`：このパラメータは 1 ～ 255 の整数にする必要があります。

- char *chars*：このパラメータは *chars* の 1 文字にする必要があり、1 文字以上のリストには文字間に区切り文字は使用しません。
- date *datefmt*：このパラメータは指定した形式に従います。*datefmt* は次の要素を使用して構成します。
  - # = 数字
  - c = 文字
  - [] = オプションの形式を囲みます
  - | = または
  - {} = 囲まれた形式の選択
  - .+- リテラル文字
- string：このパラメータは制限のない文字列です。
- host_port：このパラメータは、RFC 959 に従って、ホストポートの指定子にする必要があります。
- long_host_port：このパラメータは、RFC 1639 に従って、ホストポートの長い指定子にする必要があります。
- extended_host_port：このパラメータは、RFC 2428 に従い、拡張ホストポート指定子にする必要があります。
- {},|：このパラメータは、中括弧で囲まれ、| で区切られた選択肢の 1 つにする必要があります。
- {}, []：このパラメータは、中括弧で囲まれた選択肢の 1 つにする必要があります。オプションの値は角括弧で囲みます。

デフォルト値：なし

cmd_validity[].length

cmd_validity[].command パラメータの最大長をバイト単位で指定し、def_max_param_len で定義されているデフォルト値をオーバーライドします。FTP コマンドのパラメータが cmd_validity[].length を超え、ルール 125:3 が有効になっている場合、Snort はアラートを生成します。cmd_validity[].length を使用して、特定のコマンドを小さなパラメータ値に制限します。

長さに制限がないことを示すには、0 を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

def_max_param_len

サーバーが処理するすべてのFTPコマンドに対してインスペクタが許可するデフォルトの最大長をバイト単位で指定します。基本的なバッファオーバーフロー検出には `def_max_param_len` を使用します。（これは、`cmd_validity[].length` を使用すると個々のコマンドでオーバーライドすることができます。）このパラメータに対してルール 125:3 を有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

長さに制限がないことを示すには、`0` を指定します。

型：整数

有効な範囲：`0` 〜 `4,294,967,295 (max32)`

デフォルト値：`100`

encrypted_traffic

暗号化された FTP トラフィックを確認するかどうかを指定します。`check_encrypted` パラメータとともに使用します。このパラメータに対してルール 125:7 を有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

ftp_cmds

RFC 959 で説明されている FTP コマンド以外にサーバーがサポートする FTP コマンドのリスト。（たとえば、インストールで RFC 775 で指定されている「X」コマンドを使用している場合は、このパラメータを使用してそれらをインスペクタに追加できます。）

型：文字列

有効な値：二重引用符で囲まれ、スペースで区切られた有効な FTP コマンドのリスト。

デフォルト値：なし

ignore_data_chan

FTP データ チャネルを無視するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

ignore_telnet_erase_cmds

FTP コマンドチャネルを正規化するときに、消去文字（TNC EAC）と消去行文字（TNC EAL）の Telnet エスケープシーケンスを無視するかどうかを指定します。`ignore_telnet_erase_cmds` を設定して、FTP サーバーが Telnet 消去コマンドを処理する方法を照合します。通常、新しい

FTP クライアントはこれらの Telnet エスケープシーケンスを無視しますが、レガシークライアントは通常、それらを処理します。

telnet 消去コマンドが無視されず、ルール 125:1 が有効になっている場合、Snort はイベントを生成し、インライン展開では問題のあるパケットをドロップします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

print_cmds

初期化時にこのサーバーの各 FTP コマンドの設定を出力するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

telnet_cmds

FTP コマンドチャネルで Telnet コマンドを確認するかどうかを指定します。このようなコマンドがある場合は、FTP コマンドチャネルでの回避試行を示している可能性があります。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

FTP サーバーインスペクタのルール

`ftp_server` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

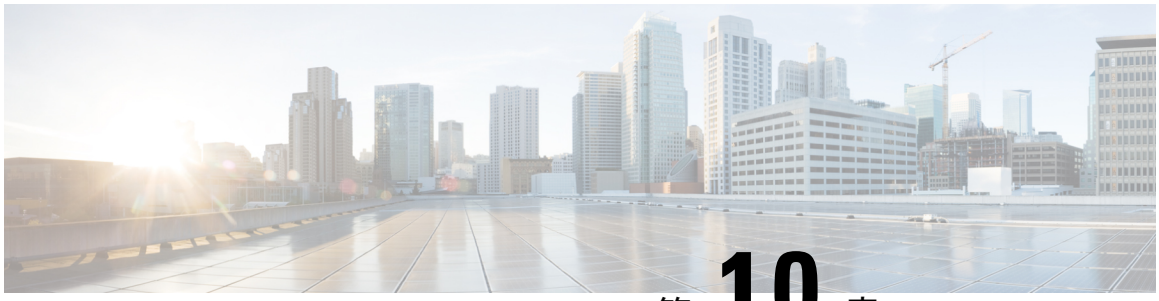
表 *10 : FTP* サーバーインスペクタのルール

GID:SID	ルール メッセージ
125:1	FTP コマンドチャネルの TELNET コマンド（TELNET command on FTP command channel）
125:2	無効な FTP コマンド（invalid FTP command）
125:3	FTPコマンドパラメータが長すぎる（FTP command parameters were too long）
125:4	FTP コマンドパラメータの形式が正しくない（FTP command parameters were malformed）

GID:SID	ルール メッセージ
125:5	FTP コマンドパラメータに文字列形式が含まれている可能性がある（FTP command parameters contained potential string format）
125:7	FTP トラフィックが暗号化されている（FTP traffic encrypted）
125:9	FTP コマンドチャネルに回避（不完全）Telnet コマンドがある（evasive (incomplete) TELNET cmd on FTP command channel）

FTP サーバーインスペクタの侵入ルールのオプション

`ftp_server` インスペクタには侵入ルールのオプションはありません。



第 10 章

GTP 検査インスペクタ

- GTP 検査インスペクタの概要 （71 ページ）
- GTP 検査インスペクタのパラメータ （71 ページ）
- GTP 検査インスペクタのルール （74 ページ）
- GTP 検査インスペクタの侵入ルールのオプション （74 ページ）

GTP 検査インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_udp`
有効	`false`

General Packet Radio Service（GPRS）Tunneling Protocol（GTP）により、GTP コア ネットワークを介した通信が実現します。

`gtp_inspect` インスペクタは、GTP トラフィックの異常を検出し、コマンド チャネル シグナリング メッセージを検査するためにルールエンジンに転送します。

GTP 検査インスペクタのパラメータ

GTP 検査サービスとポートの設定

`binder` インスペクタは GTP 検査のサービスとポートの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
   {
       "when": {
           "service": "gtp_inspect",
           "role": any
       },
       "use": {
           "type": "gtp_inspect"
       }
   },
   {
       "when": {
           "proto": "tcp",
           "role": "server",
           "ports": "2123 2152 3386"
       },
       "use": {
           "type": "gtp_inspect"
       }
   }
]
```

version

有効な GTP バージョンを指定します。

型：整数

有効な値：0、1、2

デフォルト値：2

messages[]

有効な GTP メッセージに関する情報の配列を指定します。

型：配列（オブジェクト）

例：

```
{
    messages: [
        {
            "type": 0,
            "name": ""
        }
    ]
}
```

messages[].type

有効な GTP メッセージタイプを指定します。「GTP メッセージタイプ」の表を参照してください。

型：整数

有効な範囲：0 ～ 255

デフォルト値：なし

messages[].name

有効なGTPメッセージ名を指定します。「GTPメッセージタイプ」の表を参照してください。

型：文字列

有効な値：有効な GTP メッセージ名

デフォルト値：なし

infos[]

GTP 情報要素の配列を指定します。

型：配列（オブジェクト）

例：

```
{
    infos: [
        {
            "type": 0,
            "name": "echo_request",
            "length": 0
        }
    ]
}
```

infos[].type

有効な GTP 要素タイプコードを指定します。「GTP 情報要素」の表を参照してください。

型：整数

有効な範囲：0 ～ 255

デフォルト値：0

infos[].name

有効な GTP 要素名を指定します。

型：文字列

有効な値：有効な GTP 情報要素の名前。「GTP 情報要素」の表を参照してください。

infos[].length

有効な GTP 情報要素の長さを指定します。

型：整数

有効な範囲：0 ～ 255

デフォルト値：0

GTP 検査インスペクタのルール

gtp_inspect インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *11 : GTP* インスペクタのルール

GID:SID	ルール メッセージ
143:1	メッセージ長が無効（message length is invalid）
143:2	情報要素長が無効（information element length is invalid）
143:3	情報要素の順序が正しくない（information elements are out of order）
143:4	TEID がない（TEID is missing）

GTP 検査インスペクタの侵入ルールのオプション

gtp_inspect インスペクタの侵入ルールのオプションを使用すると、GTP コマンドチャネルの GTP バージョン、メッセージタイプ、および情報要素を検査できます。

GTP オプションは、content または byte_jump と組み合わせて使用することはできません。gtp_info または gtp_type を使用するルールそれぞれで gtp_version を使用する必要があります。

gtp_version

指定した GTP バージョンを GTP 制御メッセージのバージョンと照合します。

型：整数

シンタックス：`gtp_version: <version>;`

有効な値：0、1、2

例：`gtp_version: 1;`

gtp_type

それぞれの GTP メッセージは、数値と文字列で構成されるメッセージ タイプによって識別されます。指定した GTP タイプを GTP メッセージのタイプと照合します。

次の例に示すように、メッセージ タイプとして定義済みの 10 進数値、定義済み文字列、あるいはどちらか（または両方）を任意に組み合わたカンマ区切りリストを指定できます。

型：文字列

シンタックス：`gtp_type: <message_type>;`

有効な値：GTP メッセージタイプの表に示します。「GTP メッセージタイプ」の表を参照してください。

例：`gtp_type: "10, 11, echo_request";`

リスト内のそれぞれの値または文字列を照合するとき、システムは OR 演算を使用します。値と文字列を列挙する順序は重要ではありません。リスト内のいずれか1つの値または文字列の一致により、キーワードが一致します。認識されない文字列または範囲外の値を含むルールを保存しようとすると、エラーが生成されます。

GTP バージョンに応じて、同じメッセージタイプの値が異なる場合があります。たとえば `sgsn_context_request` メッセージタイプの値は GTPv0 と GTPv1 では `50` ですが、GTPv2 では `130` です。

パケット内のバージョン番号に応じて、`gtp_type` オプションは異なる値と一致します。たとえば、`sgsn_context_request` メッセージは GTPv0 パケットまたは GTPv1 パケットでは値 `50` と一致し、GTPv2 パケットでは値 `130` と一致します。パケット内のメッセージタイプの値が、パケットで指定されたバージョンの既知の値でない場合は、オプションはパケットと一致しません。

メッセージタイプに整数を指定した場合、パケット内に指定されたバージョンとは無関係に、メッセージタイプが GTP パケット内の値と一致すればオプションは一致します。

`gtp_message_type` は、GTP メッセージタイプ の表の数値またはキーワードです。

表 *12 : GTP* メッセージ タイプ

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
1	echo_request	echo_request	echo_request
2	echo_response	echo_response	echo_response
3	version_not_supported	version_not_supported	version_not_supported
4	node_alive_request	node_alive_request	該当なし
5	node_alive_response	node_alive_response	該当なし
6	redirection_request	redirection_request	該当なし
7	redirection_response	redirection_response	該当なし
16	create_pdp_context_request	create_pdp_context_request	該当なし
17	create_pdp_context_response	create_pdp_context_response	該当なし
18	update_pdp_context_request	update_pdp_context_request	該当なし
19	update_pdp_context_response	update_pdp_context_response	該当なし
20	delete_pdp_context_request	delete_pdp_context_request	該当なし

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
21	delete_pdp_context_response	delete_pdp_context_response	該当なし
22	create_aa_pdp_context_request	init_pdp_context_activation_request	該当なし
23	create_aa_pdp_context_response	init_pdp_context_activation_response	該当なし
24	delete_aa_pdp_context_request	該当なし	該当なし
25	delete_aa_pdp_context_response	該当なし	該当なし
26	error_indication	error_indication	該当なし
27	pdu_notification_request	pdu_notification_request	該当なし
28	pdu_notification_response	pdu_notification_response	該当なし
29	pdu_notification_reject_request	pdu_notification_reject_request	該当なし
30	pdu_notification_reject_response	pdu_notification_reject_response	該当なし
31	該当なし	supported_ext_header_notification	該当なし
32	send_routing_info_request	send_routing_info_request	create_session_request
33	send_routing_info_response	send_routing_info_response	create_session_response
34	failure_report_request	failure_report_request	modify_bearer_request
35	failure_report_response	failure_report_response	modify_bearer_response
36	note_ms_present_request	note_ms_present_request	delete_session_request
37	note_ms_present_response	note_ms_present_response	delete_session_response
38	該当なし	該当なし	change_notification_request
39	該当なし	該当なし	change_notification_response
48	identification_request	identification_request	該当なし
49	identification_response	identification_response	該当なし
50	sgsn_context_request	sgsn_context_request	該当なし
51	sgsn_context_response	sgsn_context_response	該当なし
52	sgsn_context_ack	sgsn_context_ack	該当なし
53	該当なし	forward_relocation_request	該当なし
54	該当なし	forward_relocation_response	該当なし

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
55	該当なし	forward_relocation_complete	該当なし
72	該当なし	relocation_cancel_request	該当なし
57	該当なし	relocation_cancel_response	該当なし
58	該当なし	forward_srns_contex	該当なし
59	該当なし	forward_relocation_complete_ack	該当なし
60	該当なし	forward_srns_contex_ack	該当なし
64	該当なし	該当なし	modify_bearer_command
65	該当なし	該当なし	modify_bearer_failure_indication
66	該当なし	該当なし	delete_bearer_command
67	該当なし	該当なし	delete_bearer_failure_indication
68	該当なし	該当なし	bearer_resource_command
69	該当なし	該当なし	bearer_resource_failure_indication
70	該当なし	ran_info_relay	downlink_failure_indication
71	該当なし	該当なし	trace_session_activation
72	該当なし	該当なし	trace_session_deactivation
73	該当なし	該当なし	stop_paging_indication
95	該当なし	該当なし	create_bearer_request
96	該当なし	mbms_notification_request	create_bearer_response
97	該当なし	mbms_notification_response	update_bearer_request
98	該当なし	mbms_notification_reject_request	update_bearer_response
99	該当なし	mbms_notification_reject_response	delete_bearer_request
100	該当なし	create_mbms_context_request	delete_bearer_response
101	該当なし	create_mbms_context_response	delete_pdn_request
102	該当なし	update_mbms_context_request	delete_pdn_response
103	該当なし	update_mbms_context_response	該当なし
104	該当なし	delete_mbms_context_request	該当なし

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
105	該当なし	delete_mbms_context_response	該当なし
112	該当なし	mbms_register_request	該当なし
113	該当なし	mbms_register_response	該当なし
114	該当なし	mbms_deregister_request	該当なし
115	該当なし	mbms_deregister_response	該当なし
116	該当なし	mbms_session_start_request	該当なし
117	該当なし	mbms_session_start_response	該当なし
118	該当なし	mbms_session_stop_request	該当なし
119	該当なし	mbms_session_stop_response	該当なし
120	該当なし	mbms_session_update_request	該当なし
121	該当なし	mbms_session_update_response	該当なし
128	該当なし	ms_info_change_request	identification_request
129	該当なし	ms_info_change_response	identification_response
130	該当なし	該当なし	sgsn_context_request
131	該当なし	該当なし	sgsn_context_response
132	該当なし	該当なし	sgsn_context_ack
133	該当なし	該当なし	forward_relocation_request
134	該当なし	該当なし	forward_relocation_response
135	該当なし	該当なし	forward_relocation_complete
136	該当なし	該当なし	forward_relocation_complete_ack
137	該当なし	該当なし	forward_access
138	該当なし	該当なし	forward_access_ack
139	該当なし	該当なし	relocation_cancel_request
140	該当なし	該当なし	relocation_cancel_response
141	該当なし	該当なし	configuration_transfer_tunnel
149	該当なし	該当なし	detach

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
150	該当なし	該当なし	detach_ack
151	該当なし	該当なし	cs_paging
152	該当なし	該当なし	ran_info_relay
153	該当なし	該当なし	alert_mme
154	該当なし	該当なし	alert_mme_ack
155	該当なし	該当なし	ue_activity
156	該当なし	該当なし	ue_activity_ack
160	該当なし	該当なし	create_forward_tunnel_request
161	該当なし	該当なし	create_forward_tunnel_response
162	該当なし	該当なし	suspend
163	該当なし	該当なし	suspend_ack
164	該当なし	該当なし	resume
165	該当なし	該当なし	resume_ack
166	該当なし	該当なし	create_indirect_forward_tunnel_request
167	該当なし	該当なし	create_indirect_forward_tunnel_response
168	該当なし	該当なし	delete_indirect_forward_tunnel_request
169	該当なし	該当なし	delete_indirect_forward_tunnel_response
170	該当なし	該当なし	release_access_bearer_request
171	該当なし	該当なし	release_access_bearer_response
176	該当なし	該当なし	downlink_data
177	該当なし	該当なし	downlink_data_ack
179	該当なし	該当なし	pgw_restart
180	該当なし	該当なし	pgw_restart_ack
200	該当なし	該当なし	update_pdn_request
201	該当なし	該当なし	update_pdn_response
211	該当なし	該当なし	modify_access_bearer_request

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
212	該当なし	該当なし	modify_access_bearer_response
231	該当なし	該当なし	mbms_session_start_request
232	該当なし	該当なし	mbms_session_start_response
233	該当なし	該当なし	mbms_session_update_request
234	該当なし	該当なし	mbms_session_update_response
235	該当なし	該当なし	mbms_session_stop_request
236	該当なし	該当なし	mbms_session_stop_response
240	data_record_transfer_request	data_record_transfer_request	該当なし
241	data_record_transfer_response	data_record_transfer_response	該当なし
254	該当なし	end_marker	該当なし
255	pdu	pdu	該当なし

gtp_info

1 つの GTP メッセージには多数の情報要素が含まれることがあり、それぞれの要素は定義済み数値および定義済み文字列によって識別されます。`gtp_info` オプションを使用すると、指定した情報要素の先頭から検査を開始し、その情報要素に検査を限定することができます。

情報要素に対して定義された 10 進数値と定義された文字列のどちらでも指定できます。単一の値または文字列を指定することも、1 つのルール内で複数の `gtp_info` オプションを使って複数の情報要素を検査することもできます。

1 つのメッセージに同じタイプの複数の情報要素が含まれている場合は、すべてが照合対象として検査されます。情報要素が無効な順序で出現する場合は、最後のインスタンスだけが検査されます。

バージョンに応じて、GTP メッセージは同じ情報要素に対して異なる値を使用できます。たとえば `cause` 情報要素の値は GTPv0 と GTPv1 では `1` ですが、GTPv2 では `2` です。

パケット内のバージョン番号に応じて、`gtp_info` オプションは異なる値と一致します。上記の例の場合、GTPv0 または GTPv1 パケットではキーワードが情報要素値 `1` と一致しますが、GTPv2 パケットでは値 `2` と一致します。パケット内の情報要素値が、パケットで指定されたバージョンの既知の値でない場合は、オプションがパケットと一致しません。

情報要素に整数を指定した場合、パケット内に指定されたバージョンとは無関係に、メッセージタイプが GTP パケット内の値と一致すればオプションが一致します。

型：文字列

シンタックス：`gtp_info: <identifier>;`

有効な値：GTP 情報要素の表に示します。

例： `gtp_info: "qos";`

表 *13 : GTP* 情報要素

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
1	cause	cause	imsi
2	imsi	imsi	cause
3	rai	rai	recovery
4	tlli	tlli	該当なし
5	p_tmsi	p_tmsi	該当なし
6	qos	該当なし	該当なし
8	recording_required	recording_required	該当なし
9	authentication	authentication	該当なし
10	該当なし	該当なし	該当なし
11	map_cause	map_cause	該当なし
12	p_tmsi_sig	p_tmsi_sig	該当なし
13	ms_validated	ms_validated	該当なし
14	recovery	recovery	該当なし
15	selection_mode	selection_mode	該当なし
16	flow_label_data_1	teid_1	該当なし
17	flow_label_signalling	teid_control	該当なし
18	flow_label_data_2	teid_2	該当なし
19	ms_unreachable	teardown_ind	該当なし
20	該当なし	nsapi	該当なし
21	該当なし	ranap	該当なし
22	該当なし	rab_context	該当なし
23	該当なし	radio_priority_sms	該当なし
24	該当なし	radio_priority	該当なし

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
25	該当なし	packet_flow_id	該当なし
26	該当なし	charging_char	該当なし
27	該当なし	trace_ref	該当なし
28	該当なし	trace_type	該当なし
29	該当なし	ms_unreachable	該当なし
71	該当なし	該当なし	apn
72	該当なし	該当なし	ambr
73	該当なし	該当なし	ebi
74	該当なし	該当なし	ip_addr
75	該当なし	該当なし	mei
76	該当なし	該当なし	msisdn
77	該当なし	該当なし	indication
78	該当なし	該当なし	pco
79	該当なし	該当なし	paa
80	該当なし	該当なし	bearer_qos
80	該当なし	該当なし	flow_qos
82	該当なし	該当なし	rat_type
83	該当なし	該当なし	serving_network
84	該当なし	該当なし	bearer_tft
85	該当なし	該当なし	tad
86	該当なし	該当なし	uli
87	該当なし	該当なし	f_teid
88	該当なし	該当なし	tmsi
89	該当なし	該当なし	cn_id
90	該当なし	該当なし	s103pdf
91	該当なし	該当なし	s1udf

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
92	該当なし	該当なし	delay_value
93	該当なし	該当なし	bearer_context
94	該当なし	該当なし	charging_id
95	該当なし	該当なし	charging_char
96	該当なし	該当なし	trace_info
97	該当なし	該当なし	bearer_flag
99	該当なし	該当なし	pdn_type
100	該当なし	該当なし	pti
101	該当なし	該当なし	drx_parameter
103	該当なし	該当なし	gsm_key_tri
104	該当なし	該当なし	umts_key_cipher_quin
105	該当なし	該当なし	gsm_key_cipher_quin
106	該当なし	該当なし	umts_key_quin
107	該当なし	該当なし	eps_quad
108	該当なし	該当なし	umts_key_quad_quin
109	該当なし	該当なし	pdn_connection
110	該当なし	該当なし	pdn_number
111	該当なし	該当なし	p_tmsi
112	該当なし	該当なし	p_tmsi_sig
113	該当なし	該当なし	hop_counter
114	該当なし	該当なし	ue_time_zone
115	該当なし	該当なし	trace_ref
116	該当なし	該当なし	complete_request_msg
117	該当なし	該当なし	guti
118	該当なし	該当なし	f_container
119	該当なし	該当なし	f_cause

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
120	該当なし	該当なし	plmn_id
121	該当なし	該当なし	target_id
123	該当なし	該当なし	packet_flow_id
124	該当なし	該当なし	rab_contex
125	該当なし	該当なし	src_rnc_pdcp
126	該当なし	該当なし	udp_src_port
127	charge_id	charge_id	apn_restriction
128	end_user_address	end_user_address	selection_mode
129	mm_context	mm_context	src_id
130	pdp_context	pdp_context	該当なし
131	apn	apn	change_report_action
132	protocol_config	protocol_config	fq_csid
133	gsn	gsn	channel
134	msisdn	msisdn	emlpp_pri
135	該当なし	qos	node_type
136	該当なし	authentication_qu	fqdn
137	該当なし	tft	ti
138	該当なし	target_id	mbms_session_duration
139	該当なし	utran_trans	mbms_service_area
140	該当なし	rab_setup	mbms_session_id
141	該当なし	ext_header	mbms_flow_id
142	該当なし	trigger_id	mbms_ip_multicast
143	該当なし	omc_id	mbms_distribution_ack
144	該当なし	ran_trans	rfsp_index
145	該当なし	pdp_context_pri	uci
146	該当なし	addi_rab_setup	csg_info

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
147	該当なし	sgsn_number	csg_id
148	該当なし	common_flag	cmi
149	該当なし	apn_restriction	service_indicator
150	該当なし	radio_priority_lcs	detach_type
151	該当なし	rat_type	ldn
152	該当なし	user_loc_info	node_feature
153	該当なし	ms_time_zone	mbms_time_to_transfer
154	該当なし	imei_sv	throttling
155	該当なし	camel	arp
156	該当なし	mbms_ue_context	epc_timer
157	該当なし	tmp_mobile_group_id	signalling_priority_indication
158	該当なし	rim_routing_addr	tmgi
159	該当なし	mbms_config	mm_srvcc
160	該当なし	mbms_service_area	flags_srvcc
161	該当なし	src_rnc_pdcp	nmbr
162	該当なし	addi_trace_info	該当なし
163	該当なし	hop_counter	該当なし
164	該当なし	plmn_id	該当なし
165	該当なし	mbms_session_id	該当なし
166	該当なし	mbms_2g3g_indicator	該当なし
167	該当なし	enhanced_nsapi	該当なし
168	該当なし	mbms_session_duration	該当なし
169	該当なし	addi_mbms_trace_info	該当なし
170	該当なし	mbms_session_repetition_num	該当なし
171	該当なし	mbms_time_to_data	該当なし
173	該当なし	bss	該当なし

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
174	該当なし	cell_id	該当なし
175	該当なし	pdu_num	該当なし
177	該当なし	mbms_bearer_capab	該当なし
178	該当なし	rim_routing_disc	該当なし
179	該当なし	list_pfc	該当なし
180	該当なし	ps_xid	該当なし
181	該当なし	ms_info_change_report	該当なし
182	該当なし	direct_tunnel_flags	該当なし
183	該当なし	correlation_id	該当なし
184	該当なし	bearer_control_mode	該当なし
185	該当なし	mbms_flow_id	該当なし
186	該当なし	mbms_ip_multicast	該当なし
187	該当なし	mbms_distribution_ack	該当なし
188	該当なし	reliable_inter_rat_handover	該当なし
189	該当なし	rfsp_index	該当なし
190	該当なし	fqdn	該当なし
191	該当なし	evolved_allocation1	該当なし
192	該当なし	evolved_allocation2	該当なし
193	該当なし	extended_flags	該当なし
194	該当なし	uci	該当なし
195	該当なし	csg_info	該当なし
196	該当なし	csg_id	該当なし
197	該当なし	cmi	該当なし
198	該当なし	apn_ambr	該当なし
199	該当なし	ue_network	該当なし
200	該当なし	ue_ambr	該当なし

タイプ	バージョン 0 の名前	バージョン 1 の名前	バージョン 2 の名前
201	該当なし	apn_ambr_nsapi	該当なし
202	該当なし	ggsn_backoff_timer	該当なし
203	該当なし	signalling_priority_indication	該当なし
204	該当なし	signalling_priority_indication_nsapi	該当なし
205	該当なし	high_bitrate	該当なし
206	該当なし	max_mbr	該当なし
251	charging_gateway_addr	charging_gateway_addr	該当なし
255	private_extension	private_extension	private_extension

第 11 章

HTTP 検査インスペクタ

- HTTP 検査インスペクタの概要 （89 ページ）
- HTTP 検査インスペクタを設定するためのベストプラクティス （91 ページ）
- HTTP 検査インスペクタのパラメータ （91 ページ）
- HTTP 検査インスペクタのルール （100 ページ）
- HTTP 検査インスペクタの侵入ルールのオプション （107 ページ）

HTTP 検査インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`true`

Hypertext Transfer Protocol （HTTP）は、クライアントとサーバー間でハイパーメディア（オーディオ、ビデオ、画像、およびテキスト）の交換を可能にするアプリケーション層プロトコルです。HTTP は、信頼性の高いメッセージ送信を必要とするステートレスプロトコルです。クライアントとサーバー間の通信は、HTTP 要求と応答の形式で行われます。

HTTP/1.1 サーバーは通常、TCP/IP を介したポート 80 を使用します。セキュアバージョンの HTTP（HTTP/TLS または HTTPS）はポート 443 を使用します。HTTP は、プロトコルでアクセス制御と認証メカニズムを定義します。

HTTP/2 が改善され、速度が増加し、クライアントが要求したよりも多くの情報がプッシュされるようになりましたが、HTTP/1.1 と同じポートとプロトコルで動作します。HTTP/2 固有のルールは `service:http2` で設定されます。

HTTP/3 は TCP ではなく QUIC（Quick UDP Internet Connections）プロトコルを使用するコネクションレス型であるため、より優れた損失回復率でより多くのアクティブストリームをサポー

トできます。HTTP/3 は、以前のバージョンの HTTP と同じメッセージングを使用します。HTTP/3 固有のルールは `service:http3` で設定されます。

HTTP インスペクタは、3 つのバージョンの HTTP をすべて同じ方法でサポートします。

`http_inspect` インスペクタは、HTTP メッセージのプロトコルデータユニット（PDU）を検出して分析します。`http_inspect` は、TCP ストリームから TCP ペイロードを受信し、カプセル化された HTTP メッセージを確認します。

HTTP インスペクタは、次の HTTP メッセージセクションを検出できます。

- 要求行
- ステータス行
- ヘッダー
- Content-Length メッセージ本文（Content-length ヘッダーで定義されるメッセージ本文）
- チャンクされたメッセージ本文
- 前のメッセージ本文（Content-Length ヘッダーのないメッセージ本文）
- Trailers

`http_inspect` インスペクタは、すべての HTTP ヘッダーフィールドと HTTP URI のコンポーネントを検出して正規化します。`http_inspect` インスペクタは TCP ポートを正規化しません。

`http_inspect` インスペクタは、4 種類の URI を検出できます。

- アスタリスク（*）：正規化されていません
- 権限（Authority）： HTTP CONNECT メソッドで使用される URI
- 発信元（Origin）：スラッシュで始まる URI （スキームや権限の存在なし）
- 絶対（Absolute）：スキーム、ホスト、および絶対パスを含む URI。

HTTP URI には、次のものを含めることができます。

- スキーム（Scheme）（ftp、http、または https）
- ホスト（Host ）（サーバーのドメイン名）
- TCP ポート（TCP port）
- パス （Path）（ディレクトリとファイル）
- クエリ（Query）（要求パラメータ）
- フラグメント （Fragment）（ファイルの一部）

HTTP メッセージのセクションでアラートを発生させるように `http_inspect` インスペクタを設定できます。次に例を示します。

- HTTP 要求または応答の本文から読み取るバイト数を指定する

- JavaScript の検出と正規化を有効にする
- さまざまな種類のファイルの圧縮解除に対応する
- HTTP URI の復号化をカスタマイズする

（注） `http_inspect` インスペクタは、ストリーム TCP ペイロードを部分的に検査できます。

HTTP 検査インスペクタを設定するためのベストプラクティス

http_inspect インスペクタを構成するときは、次のベストプラクティスを考慮してください。

- HTTP トラフィックに大きなビデオファイルが含まれている場合は、`request_depth` パラメータと `response_depth` パラメータを設定します。
- HTTP URI 検査パラメータのデフォルト設定を使用します。

```
"utf8": "true"
"plus_to_space": "true"
"percent_u": "true"
"utf8_bare_byte": "true"
"iis_unicode": "true"
"iis_double_decode": "true"
```

HTTP 検査インスペクタのパラメータ

HTTP サービスの設定

`binder` インスペクタは、HTTP サービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
             "service": "http",
             "role": any
        },
        "use": {
            "type": "http_inspect"
        }
    }
]
```

request_depth

HTTP メッセージの要求本文から読み取るバイト数を指定します。

検査するバイト数に制限を設定しない場合は、-1 を指定します。分析する HTTP 本文データの量を制限する場合は、request_depth パラメータと response_depth パラメータを指定することをお勧めします。

HTTP ヘッダーのみを検査するには、request_depth を 0 に設定します。

型：整数

有効な範囲：-1 ～ 9,007,199,254,740,992 (max53)

デフォルト値：-1

response_depth

HTTP メッセージの応答本文から読み取るバイト数を指定します。

検査するバイト数に制限を設定しない場合は、-1 を指定します。分析する HTTP 本文データの量を制限する場合は、request_depth パラメータと response_depth パラメータを指定することをお勧めします。

HTTP ヘッダーのみを検査するには、response_depth を 0 に設定します。

型：整数

有効な範囲：-1 ～ 9,007,199,254,740,992 (max53)

デフォルト値：-1

unzip

gzip ファイルの圧縮を解除し、メッセージ本文を検査する前にデフレートするかどうかを指定します。圧縮解除を無効にすると、HTTP インスペクタは HTTP メッセージ本文のすべての部分を処理できなくなります。http_inspect インスペクタは HTTP ヘッダーを処理できます。

型：ブール値

有効な値：true、false

デフォルト値：true

maximum_host_length

Host HTTP ヘッダーフィールドで許可されるバイトの最大数を指定します。

ヘッダー値の長さに制限を設定しない場合は、-1 を指定します。

型：整数

有効な範囲：-1 ～ 9,007,199,254,740,992 (max53)

デフォルト値：-1

maximum_chunk_length

HTTP メッセージの本文チャンクで許可される最大バイト数を指定します。

HTTP チャンクのバイト数に制限を設定しない場合は、`-1` を指定します。

型：整数

有効な範囲：`-1` 〜 `9,007,199,254,740,992 (max53)`

デフォルト値：`-1`

normalize_utf

HTTP応答本文で検出されたUTFエンコーディング（UTF-8、UTF-7、UTF-16LE、UTF-16BE、UTF-32LE、およびUTF-32BE）を正規化するかどうかを指定します。`http_inspect` インスペクタは、HTTP `Content-Type` ヘッダーからの UTF 文字エンコードを決定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`true`

decompress_pdf

HTTP応答本文で検出された `application/pdf`（PDF）ファイルのデフレート互換の圧縮部分の圧縮を解除するかどうかを指定します。 `http_inspect` インスペクタは、 `/FlateDecode` ストリームフィルタを使用して PDF ファイルの圧縮を解除します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_swf

HTTP応答本文で検出された `application/vnd.adobe.flash-movie`（SWF）ファイルの圧縮を解除するかどうかを指定します。

（注） HTTP GET 応答で検出されたファイルの圧縮部分のみを圧縮解除できます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_vba

HTTP 応答本文で検出された Microsoft Office Visual Basic for Applications のマクロファイルの圧縮を解除するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_zip

HTTP 応答本文で検出された`application/zip`（ZIP）ファイルの圧縮を解除するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

script_detection

スクリプトの末尾要素（<\script>）を検出した後、JavaScript の内容を検査するかどうかを指定します。`http_inspect` は、スクリプトの末尾を検出すると、早期検出のために部分的に読み取られたメッセージ本文をすぐに転送します。スクリプト検出により、Snort は悪意のある JavaScript を含む可能性のある応答メッセージをすばやくブロックできます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

normalize_javascript

HTTP 応答本文で JavaScript を正規化するためにレガシーメカニズムを使用するかどうかを指定します。このオプションは、レガシー JavaScript ノーマライザを構成します。`http_inspect` インスペクタは `unescape` 関数や `decodeURI` 関数、`String.fromCharCode` メソッドなどの難読化 JavaScript データを正規化します。HTTP インスペクタは、unescape 関数、decodeURI 関数、および decodeURIComponent 関数内のエンコード（%XX、%uXXXX、XX、および uXXXXi）を正規化します。

`http_inspect` インスペクタは連続する空白文字を検出し、1 つのスペースに正規化します。`normalize_javascript` が有効になっている場合、`max_javascript_whitespaces` を設定して、難読化された JavaScript の連続する空白文字の数を制限できます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

js_norm_bytes_depth

正規化する入力 JavaScript のバイト数を指定します。このオプションは、拡張 JavaScript ノーマライザに固有です。

（注） 拡張 JavaScript ノーマライザを使用する場合、Lightweight Security Package（LSP）および Snort 3 のデフォルト設定が使用されます。JavaScript 固有の構成は、ネットワーク分析ポリシー（NAP）ユーザーインターフェイスからブロックされます。デフォルト設定を上書きしてノーマライザ設定をカスタマイズするには、`/ftd/app_data/Volume/root1/ngfw/var/cisco/deploy` にある `NAPOverride.lua` ファイルを変更します。

`http_inspect` インスペクタは連続する空白文字を検出し、1 つのスペースに正規化します。インスペクタは、トラフィックを効果的に正規化するために、開始 <script> is in one PDU and the end </script> が別の PDU にあるさまざまな PDU のスクリプトを追跡します。新しいバッファ `js_data` は、ジャストインタイム（JIT）アプローチを使用して JavaScript コードを検出および正規化する Snort 3 IPS バッファに追加され、このオプションがルールで使用されている場合にのみノーマライザが呼び出されます。

`http_inspect` インスペクタは、JavaScript コードに関連付けられた関数名、変数名、およびラベル名を正規化します。さらに、インスペクタは、application/javascript または類似の MIME タイプを使用して、外部スクリプトの形式で転送される JavaScript コードを正規化します。ノーマライザは、JavaScript 機能がクライアント側からの元の入力から変更されない自動セミコロン挿入を実行します。

`http_inspect` インスペクタは、`unescape` 関数、`decodeURI` 関数、および `decodeURIComponent` 関数と、`String.fromCharCode` メソッドおよび `String.fromCodePoint` メソッドなどの難読化 JavaScript データを正規化します。HTTP インスペクタは、unescape 関数、decodeURI 関数、および decodeURIComponent 関数内のエンコード（%XX、%uXXXX、\uXX、\u{XXXX}\xXX、10 進数コードポイント、16 進数コードポイント）を正規化します。

また、`http_inspect` インスペクタは Javascript のプラス（+）演算子を正規化し、その演算子を使用して文字列を連結します。

JavaScript のバイト数に制限を設定しない場合は、`-1` を指定します。

型：整数

有効な範囲：`-1` 〜 `9,007,199,254,740,992 (max53)`

デフォルト値：`-1`

js_norm_identifier_depth

正規化する一意の JavaScript 識別子の最大数を指定します。このオプションは、拡張 JavaScript ノーマライザに固有です。

（注） 拡張 JavaScript ノーマライザを使用する場合、Lightweight Security Package（LSP）および Snort 3 のデフォルト設定が使用されます。JavaScript 固有の構成は、ネットワーク分析ポリシー（NAP）ユーザーインターフェイスからブロックされます。デフォルト設定を上書きしてノーマライザ設定をカスタマイズするには、`/ftd/app_data/Volume/root1/ngfw/var/cisco/deploy` にある `NAPOverride.lua` ファイルを変更します。

型：整数

有効な範囲：`0` 〜 `65536`

デフォルト値：`65536`

js_norm_max_bracket_depth

正規化する JavaScript ブラケットのネストの最大深度を指定します。このオプションは、拡張 JavaScript ノーマライザに固有です。

（注） 拡張 JavaScript ノーマライザを使用する場合、Lightweight Security Package（LSP）および Snort 3 のデフォルト設定が使用されます。JavaScript 固有の構成は、ネットワーク分析ポリシー（NAP）ユーザーインターフェイスからブロックされます。デフォルト設定を上書きしてノーマライザ設定をカスタマイズするには、`/ftd/app_data/Volume/root1/ngfw/var/cisco/deploy` にある `NAPOverride.lua` ファイルを変更します。

型：整数

有効な範囲：`1` 〜 `65535`

デフォルト値：`256`

js_norm_max_scope_depth

正規化する JavaScript スコープのネストの最大深度を指定します。このオプションは、拡張 JavaScript ノーマライザに固有です。

（注） 拡張 JavaScript ノーマライザを使用する場合、Lightweight Security Package（LSP）および Snort 3 のデフォルト設定が使用されます。JavaScript 固有の構成は、ネットワーク分析ポリシー（NAP）ユーザーインターフェイスからブロックされます。デフォルト設定を上書きしてノーマライザ設定をカスタマイズするには、`/ftd/app_data/Volume/root1/ngfw/var/cisco/deploy` にある `NAPOverride.lua` ファイルを変更します。

型：整数

有効な範囲：1 ～ 65535

デフォルト値：256

js_norm_max_tmpl_nest

正規化する JavaScript テンプレートリテラルのネストの最大深度を指定します。このオプションは、拡張 JavaScript ノーマライザに固有です。

（注） 拡張 JavaScript ノーマライザを使用する場合、Lightweight Security Package（LSP）および Snort 3 のデフォルト設定が使用されます。JavaScript 固有の構成は、ネットワーク分析ポリシー（NAP）ユーザーインターフェイスからブロックされます。デフォルト設定を上書きしてノーマライザ設定をカスタマイズするには、/ftd/app_data/Volume/root1/ngfw/var/cisco/deployにある NAPOverride.lua ファイルを変更します。

型：整数

有効な範囲：0 ～ 255

デフォルト値：32

max_javascript_whitespaces

JavaScript 難読化データ内で許可される連続する空白文字の最大数を指定します。

型：整数

有効な範囲：1 ～ 65535

デフォルト値：200

percent_u

%uNNNN エンコーディングと %UNNNN エンコーディングを正規化するかどうかを指定します。4 つの N 文字は、Microsoft インターネット インフォメーション サービス（IIS）の Unicode コードポイントに関連する 16 進数でエンコードされた値を表します。正規のクライアントが %u エンコーディングを使用することはほとんどないため、%u エンコーディングによってエンコードされている HTTP トラフィックを復号化することをお勧めします。

型：ブール値

有効な値：true、false

デフォルト値：false

utf8

URI の標準 UTF-8 Unicode シーケンスを正規化するかどうかを指定します。http_inspect インスペクタは、2 バイトまたは 3 バイトの UTF-8 文字を 1 バイトに正規化できます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`true`

utf8_bare_byte

URL またはパーセントエンコードされていないバイトを含む UTF-8 文字を正規化するかどうかを指定します。`utf8_bare_byte` パラメータを有効にすることをお勧めします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

iis_unicode

HTTP メッセージ内の文字を Unicode コードポイントで正規化するかどうかを指定します。

（注） `iis_unicode` パラメータを有効にすることをお勧めします。Unicode は、攻撃試行や回避試行によく見られます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

iis_unicode_code_page

IIS Unicode マップファイルのコードページを使用するかどうかを指定します。

型：整数

有効な範囲：`1` ～ `65535`

デフォルト値：`1252`

iis_double_decode

URL でエンコードされた文字を二重デコードして文字を正規化するかどうかを指定します。要求 URI を介して 2 回通過して IIS 二重エンコードトラフィックを復号化します。`iis_double_decode` パラメータを有効にすることをお勧めします。通常、二重エンコーディングは攻撃シナリオでのみ見られます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`true`

oversize_dir_length

URL ディレクトリで許可される最大バイト数を指定します。

型：整数

有効な範囲：1 ～ 65535

デフォルト値：300

backslash_to_slash

URI でバックスラッシュ（\）をスラッシュ（/）に置き換えるかどうかを指定します。

型：ブール値

有効な値：true、false

デフォルト値：true

plus_to_space

URI の プラス記号（+）を <sp> に置き換えるかどうかを指定します。

型：ブール値

有効な値：true、false

デフォルト値：true

simplify_path

URIディレクトリパスを最も単純な形式に減らすかどうかを指定します。余分なトラバーサルを含む URI ディレクトリ パスには、. 、..、/ が含まれる場合があります。

型：ブール値

有効な値：true、false

デフォルト値：true

xff_headers

調べる X-Forwarded-For HTTP ヘッダーのタイプを指定します。xff_headers パラメータで、X-Forwarded-For ヘッダーを優先順位の高いものから順にリストします。

カスタム X-Forwarded-For タイプのヘッダーを定義できます。元のクライアント IP アドレスを伝送する HTTP ヘッダーには、ベンダー固有のヘッダー名を付けることができます。このシナリオでは、xff_headers パラメータは、カスタムヘッダーを HTTP インスペクタに導入する方法を提供します。

xff_headers のデフォルト値は、2 つの一般的に知られているヘッダーである x-forwarded-for true-client-ip です。ストリーム内に両方のデフォルトヘッダーが存在する場合、true-client-ip よりも x-forwarded-for が優先されます。X-Forwarded-For HTTP ヘッダーを複数指定する場合は、ヘッダー名をスペースで区切ります。

型：文字列

有効な値：`x-forwarded-for、true-client-ip`

デフォルト値：`x-forwarded-for true-client-ip`

HTTP 検査インスペクタのルール

`http_inspect` インスペクタルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *14 : HTTP* 検査インスペクタのルール

GID:SID	ルール メッセージ
119:1	URIに予約されていない文字のパーセントエンコーディングがある（URI has percent-encoding of an unreserved character）
119:2	URI はパーセントエンコーディングされており、結果は再びパーセントエンコーディングされる（URI is percent encoded and the result is percent encoded again）
119:3	URI に標準以外の %u スタイルの Unicode エンコーディングがある（URI has non-standard %u-style Unicode encoding）
119:4	URIに、パーセントエンコーディングされていないバイトを含むUnicodeエンコーディングがある（URI has Unicode encodings containing bytes that were not percent-encoded）
119:6	URI に 2 バイトまたは 3 バイトの UTF-8 エンコーディングがある（URI has two-byte or three-byte UTF-8 encoding）
119:7	URI に Unicode マップ コード ポイント エンコーディングがある（URI has unicode map code point encoding）
119:8	URI パスに連続したスラッシュ文字が含まれている（URI path contains consecutive slash characters）
119:9	URIのパス部分にバックスラッシュ文字が表示される（backslash character appears in the path portion of a URI）
119:10	URI パスに現在のディレクトリを繰り返す /./ パターンが含まれている（URI path contains /./ pattern repeating the current directory）
119:11	URIパスにディレクトリを上に移動する/../パターンが含まれている（URI path contains /../ pattern moving up a directory）
119:12	HTTP 開始行内にタブ文字がある（Tab character in HTTP start line）

GID:SID	ルール メッセージ
119:13	HTTP 開始行またはヘッダー行が CR なしに LF で終了する（HTTP start line or header line terminated by LF without a CR）
119:14	正規化された URI に bad_characters リストの文字が含まれている（Normalized URI includes character from bad_characters list）
119:15	URI パスに oversize_dir_length パラメータよりも長いセグメントが含まれている（URI path contains a segment that is longer than the oversize_dir_length parametery）
119:16	チャンク長が、設定された maximum_chunk_length を超えている（chunk length exceeds configured maximum_chunk_length）
119:18	URI パスにルートディレクトリを超える /../ が含まれている（URI path includes /../ that goes above the root directory）
119:19	HTTP ヘッダー行が 4096 バイトを超えている（HTTP header line exceeds 4096 bytes）
119:20	HTTP メッセージに 200 を超えるヘッダーフィールドがある（HTTP message has more than 200 header fields）
119:21	HTTP メッセージに複数の Content-Length ヘッダー値がある（HTTP message has more than one Content-Length header value）
119:24	ホストヘッダーフィールドが複数回表示されるか、複数の値が含まれている（Host header field appears more than once or has multiple values）
119:25	HTTP ホストヘッダーフィールド値の長さが maximum_host_length オプションを超えている（length of HTTP Host header field value exceeds maximum_host_length option）
119:28	content-length またはチャンクのない HTTP POST 要求または PUT 要求（HTTP POST or PUT request without content-length or chunks）
119:31	Snort が HTTP 要求メソッドを認識していない（HTTP request method is not known to Snort）
119:32	HTTP 要求は HTTP/0.9 と呼ばれるプリミティブ HTTP フォーマットを使用する（HTTP request uses primitive HTTP format known as HTTP/0.9）
119:33	HTTP 要求 URI にパーセントエンコーディングされていない余白文字が含まれている（HTTP request URI has space character that is not percent-encoded）

GID:SID	ルール メッセージ
119:34	HTTP 接続に応答されていない 100 を超える同時パイプライン要求がある（HTTP connection has more than 100 simultaneous pipelined requests that have not been answered）
119:102	HTTP 応答の無効なステータスコード（invalid status code in HTTP response）
119:104	HTTP 応答に正規化に失敗した UTF 文字セットが含まれている（HTTP response has UTF character set that failed to normalize）
119:105	HTTP 応答に UTF-7 文字セットが含まれている（HTTP response has UTF-7 character set）
119:109	JavaScript の難読化レベルの幅がある（more than one level of JavaScript obfuscation）
119:110	連続した JavaScript の空白が最大許容数を超えている（consecutive JavaScript whitespaces exceed maximum allowed）
119:111	JavaScript 難読化データ内にエンコーディングが複数ある（multiple encodings within JavaScript obfuscated data）
119:112	SWF ファイルの zlib の圧縮解除に失敗した（SWF file zlib decompression failure）
119:113	SWF ファイル LZMA の圧縮解除に失敗した（SWF file LZMA decompression failure）
119:114	PDF ファイルのデフレートに失敗した（PDF file deflate decompression failure）
119:115	PDF ファイルのサポート対象外の圧縮タイプ（PDF file unsupported compression type）
119:116	複数の圧縮が PDF ファイルに適用されている（PDF file with more than one compression applied）
119:117	PDF ファイルの解析に失敗した（PDF file parse failure）
119:201	HTTP トラフィックではないか、または回復不能な HTTP プロトコルエラー（not HTTP traffic or unrecoverable HTTP protocol error）
119:202	チャンクの長さの先行ゼロが多すぎる（chunk length has excessive leading zeros）
119:203	HTTP メッセージの前または間の空白文字（white space before or between HTTP messages）
119:204	URI のない要求メッセージ（request message without URI）

GID:SID	ルール メッセージ
119:205	HTTP 応答の理由句内の制御文字（control character in HTTP response reason phrase）
119:206	開始行の不正で余分な空白文字（illegal extra whitespace in start line）
119:207	HTTP バージョンの破損（corrupted HTTP version）
119:209	HTTP ヘッダーでのフォーマットエラー（format error in HTTP header）
119:210	チャンクヘッダーのオプションが存在する（chunk header options present）
119:211	URI の形式が正しくない（URI badly formatted）
119:212	URI のパーセントエンコーディングのタイプが認識されない（unrecognized type of percent encoding in URI）
119:213	HTTP チャンクのフォーマットが正しくない（HTTP chunk misformatted）
119:214	チャンク長に隣接する空白文字がある（white space adjacent to chunk length）
119:215	ヘッダー名内に空白文字がある（white space within header name）
119:216	過度な gzip 圧縮（excessive gzip compression）
119:217	gzip 圧縮解除に失敗した（gzip decompression failed）
119:218	HTTP 0.9 が要求され、その後に別の要求が続いている（HTTP 0.9 requested followed by another request）
119:219	通常の要求の後に HTTP 0.9 要求が続いている（HTTP 0.9 request following a normal request）
119:220	メッセージに Content-Length と Transfer-Encoding の両方がある（message has both Content-Length and Transfer-Encoding）
119:221	Transfer-Encoding またはゼロ以外の Content-Length と組み合わされた本文がないことを示すステータスコード（status code implying no body combined with Transfer-Encoding or nonzero Content-Length）
119:222	Transfer-Encoding の末尾がチャンクされていない（Transfer-Encoding not ending with chunked）
119:223	チャンク前のエンコーディングによる Transfer-Encoding（Transfer-Encoding with encodings before chunked）
119:224	HTTP トラフィックの形式が誤っている（misformatted HTTP traffic）
119:225	サポート対象外の Content-Encoding が使用されている（unsupported Content-Encoding used）

GID:SID	ルール メッセージ
119:226	不明な Content-Encoding が使用されている（unknown Content-Encoding used）
119:227	複数の Content-Encodings が適用されている（multiple Content-Encodings applied）
119:228	クライアント要求前のサーバー応答（server response before client request）
119:229	サーバー応答の PDF/SWF/ZIP 圧縮解除が大きすぎる（PDF/SWF/ZIP decompression of server response too big）
119:230	HTTP メッセージヘッダー名の非表示文字（nonprinting character in HTTP message header name）
119:231	HTTP ヘッダーの Content-Length 値が正しくない（bad Content-Length value in HTTP header）
119:232	HTTP ヘッダー行の折り返し（HTTP header line wrapped）
119:233	HTTP ヘッダー行が LF のない CR で終了した（HTTP header line terminated by CR without a LF）
119:234	チャンクが非標準セパレータで終了した（chunk terminated by nonstandard separator）
119:235	チャンクの長さが CR なしの LF で終了した（chunk length terminated by LF without CR）
119:236	複数の応答に 100 番台のステータスコードがある（chunk length terminated by LF without CR）
119:237	100 番台のステータスコードが Expect ヘッダーに対応していない（100 status code not in response to Expect header）
119:238	100 または 101 以外の 1XX ステータスコード（1XX status code other than 100 or 101）
119:239	メッセージの本文なしで送信されるヘッダーを予測（Expect header sent without a message body）
119:240	HTTP 1.0 メッセージに Transfer-Encoding ヘッダーが含まれている（HTTP 1.0 message with Transfer-Encoding header）
119:241	Content-Transfer-Encoding が HTTP ヘッダーとして使用されている（Content-Transfer-Encoding used as HTTP header）
119:242	チャンクされたメッセージトレーラに不正なフィールドがある（illegal field in chunked message trailers）

GID:SID	ルール メッセージ
119:243	ヘッダーフィールドが不適切に2回表示されるか、2つの値を持つ（header field inappropriately appears twice or has two values）
119:244	Content-Encoding ヘッダーでチャンクされた無効な値
119:245	Range ヘッダーのない要求に 206 応答が送信された（206 response sent to a request without a Range header）
119:246	バージョンフィールドの HTTP がすべて大文字ではない（HTTP in version field not all upper case）
119:247	重要なヘッダー値に空白文字が埋め込まれている（white space embedded in critical header value）
119:248	gzip 圧縮データの後に予期しない非 gzip データが続く（gzip compressed data followed by unexpected non-gzip data）
119:249	過剰な HTTP パラメータキーが繰り返される（excessive HTTP parameter key repeats）
119:253	メッセージの本文に HTTP CONNECT 要求が含まれている（HTTP CONNECT request with a message body）
119:254	CONNECT 要求後、CONNECT 応答前の HTTP クライアントからサーバーへのトラフィック（HTTP client-to-server traffic after CONNECT request but before CONNECT response）
119:255	HTTP CONNECT 2XX 応答にContent-Length ヘッダーが含まれている（HTTP CONNECT 2XX response with Content-Length header）
119:256	HTTP CONNECT 2XX 応答に Transfer-Encoding ヘッダーが含まれている（HTTP CONNECT 2XX response with Transfer-Encoding header）
119:257	HTTP CONNECT 応答に 1XX ステータスコードが含まれている（HTTP CONNECT response with 1XX status code）
119:258	要求メッセージが完了する前の HTTP CONNECT 応答（HTTP CONNECT response before request message completed）
119:259	HTTP Content-Disposition ファイル名パラメータの形式が間違っている（malformed HTTP Content-Disposition filename parameter）
119:260	HTTP Content-Length メッセージの本文が切り捨てられた（HTTP Content-Length message body was truncated）
119:261	HTTP チャンクメッセージの本文が切り捨てられた（HTTP chunked message body was truncated）

GID:SID	ルール メッセージ
119:262	HTTP URI スキーム長が 10 文字を超えている（HTTP URI scheme longer than 10 characters）
119:263	HTTP/1 クライアントが HTTP/2 へのアップグレードを要求した（HTTP/1 client requested HTTP/2 upgrade）
119:264	HTTP/1 サーバーが HTTP/2 へのアップグレードを許可した（HTTP/1 server granted HTTP/2 upgrade）
119:265	JavaScript に不正なトークンがある（bad token in JavaScript）
119:266	JavaScript に予期しないスクリプト開始タグがある（unexpected script opening tag in JavaScript）
119:267	JavaScript に予期しないスクリプト終了タグがある（unexpected script closing tag in JavaScript）
119:268	外部スクリプトタグの下に JavaScript コードがある（JavaScript code under the external script tags）
119:269	短い形式のスクリプト開始タグ（script opening tag in a short form）
119:270	一意の JavaScript 識別子の最大数（max number of unique JavaScript identifiers reached）
119:271	JavaScript ブラケットのネストがキャパシティを超えている（JavaScript bracket nesting is over capacity）
119:272	HTTP Accept-Encoding ヘッダーに連続したコンマがある（Consecutive commas in HTTP Accept-Encoding header）
119:273	JavaScript の正規化中に PDU が欠落した（missed PDUs during JavaScript normalization）
119:274	JavaScript スコープのネストがキャパシティを超えている（JavaScript scope nesting is over capacity）
119:275	HTTP/1 バージョンが 1.0 または 1.1e 以外（HTTP/1 version other than 1.0 or 1.1e）
119:276	開始行の HTTP バージョンが 0 になっている（HTTP version in start line is 0）
119:277	開始行の HTTP バージョンが 1 より大きい（HTTP version in start line is higher than 1）

HTTP 検査インスペクタの侵入ルールのオプション

http_client_body

検出カーソルを HTTP リクエストの本文に設定します。HTTP メッセージで HTTP ヘッダーを指定しなかった場合、Snort は URI 正規化を使用して `http_client_body` を正規化します。URI 正規化は通常、`http_header` に適用されます。

シンタックス：`http_client_body;`

例：`http_client_body;`

http_cookie

抽出された HTTP `Cookie` ヘッダーフィールドに検出カーソルを設定します。`http_cookie` ルールオプションには、パラメータの `http_cookie.request`、`http_cookie.with_header`、`http_cookie.with_body`、および `http_cookie.with_trailer` が含まれます。

シンタックス：http_cookie: <parameter>, <parameter>

例：`http_cookie: request;`

http_cookie.request

HTTP 要求メッセージで検出された HTTP Cookie を照合します。HTTP 応答を確認する場合は、HTTP 要求 Cookie を使用します。`http_cookie.request` パラメータはオプションです。

シンタックス：`http_cookie: request;`

例：`http_cookie: request;`

http_cookie.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_cookie.with_header` パラメータはオプションです。

シンタックス：`http_cookie: with_header;`

例：`http_cookie: with_header;`

http_cookie.with_body

ルールの別の部分で確認するのは `http_cookie` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_cookie.with_body` パラメータはオプションです。

シンタックス：`http_cookie: with_body;`

例：`http_cookie: with_body;`

http_cookie.with_trailer

ルールの別の部分で確認するのは `http_cookie` ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_cookie.with_trailer` パラメータはオプションです。

シンタックス：`http_cookie: with_trailer;`

例：`http_cookie: with_trailer;`

http_header

検出カーソルを正規化された HTTP ヘッダーに設定します。個々のヘッダー名を指定するには、`field` オプションを使用します。

`http_header` ルールオプションには、パラメータの `http_header.field`、`http_header.request`、`http_header.with_header`、`http_header.with_body`、および `http_header.with_trailer` が含まれます。

シンタックス：`http_header: field <field_name>,<parameter>, <parameter>`

例：`http_header: field Content-Type, with_trailer;`

http_header.field

指定したヘッダー名を正規化された HTTP ヘッダーと照合します。ヘッダー名では大文字と小文字が区別されません。ヘッダー名を指定しなかった場合、HTTP インスペクタは、HTTP Cookie のヘッダー（`Cookie` と `Set-Cookie`）を除くすべてのヘッダーを確認します。

型：文字列

シンタックス：`http_header: field <field_name>;`

有効な値：HTTP ヘッダー名。

例：`http_header: field Content-Type;`

http_header.request

HTTP 要求で検出されたヘッダーを照合します。HTTP 応答を確認する場合は、HTTP 要求ヘッダーを使用します。`http_header.request` パラメータはオプションです。

シンタックス：`http_header: request;`

例：`http_header: request;`

http_header.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_header.with_header` パラメータはオプションです。

シンタックス：`http_header: with_header;`

例：`http_header: with_header;`

http_header.with_body

ルールの別の部分で確認するのは http_header ルールオプションではなく、HTTP メッセージの本文であることを指定します。http_header.with_body パラメータはオプションです。

シンタックス：`http_header: with_body;`

例：`http_header: with_body;`

http_header.with_trailer

ルールの別の部分で確認するのは http_header ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。http_header.with_trailer パラメータはオプションです。

シンタックス：`http_header: with_trailer;`

例：`http_header: with_trailer;`

http_method

検出カーソルを HTTP 要求のメソッドに設定します。一般的な HTTP 要求メソッドの値は、GET、POST、OPTIONS、HEAD、DELETE、PUT、TRACE、および CONNECT です。

http_method ルールオプションには、パラメータの http_method.with_header、http_method.with_body、および http_method.with_trailer が含まれます。

シンタックス：`http_method: <parameter>, <parameter>;`

例：`http_method; content:"GET";`

http_method.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。http_method.with_header パラメータはオプションです。

シンタックス：`http_method: with_header;`

例：`http_method: with_header;`

http_method.with_body

ルールの別の部分で確認するのは http_header ルールオプションではなく、HTTP メッセージの本文であることを指定します。http_method.with_body パラメータはオプションです。

シンタックス：`http_method: with_body;`

例：`http_method: with_body;`

http_method.with_trailer

ルールの別の部分で確認するのは http_header ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。http_method.with_trailer パラメータはオプションです。

シンタックス：`http_method: with_trailer;`

例：`http_method: with_trailer;`

http_param

検出カーソルを指定した HTTP パラメータキーに設定します。HTTP パラメータキーは、クエリまたは要求の本文に表示される場合があります。

`http_param` ルールオプションには、`http_param.param` パラメータと `http_method.nocase` が含まれます。

シンタックス：`http_param: <parameter_key>, nocase;`

例：`http_param: offset, nocase;`

http_param.param

指定したパラメータを照合します。

型：文字列

シンタックス：`http_param: <http_parameter>;`

有効な値：要求クエリパラメータまたは要求の本文フィールド。

例：`http_param: offset;`

http_param.nocase

指定したパラメータを照合しますが、大文字小文字は考慮されません。`http_param.nocase` パラメータはオプションです。

シンタックス：`http_param: nocase;`

例：`http_param: nocase;`

http_raw_body

検出カーソルを正規化されていない要求または応答メッセージの本文に設定します。

シンタックス：`http_raw_body;`

例： `http_raw_body;`

http_raw_cookie

検出カーソルを正規化されていない HTTP `Cookie` ヘッダーに設定します。`http_raw_cookie` ルールオプションには、パラメータの `http_raw_cookie.request`、`http_raw_cookie.with_header`、`http_raw_cookie.with_body`、および `http_raw_cookie.with_trailer` が含まれます。

シンタックス：`http_raw_cookie: <parameter>, <parameter>;`

例：`http_raw_cookie: request;`

http_raw_cookie.request

HTTP 要求で検出された Cookie を照合します。応答メッセージを確認する場合は、HTTP 要求の Cookie を使用します。`http_raw_cookie.request` パラメータはオプションです。

シンタックス：`http_raw_cookie: request;`

例：`http_raw_cookie: request;`

http_raw_cookie.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_raw_cookie.with_header` パラメータはオプションです。

シンタックス：`http_raw_cookie: with_header;`

例：`http_raw_cookie: with_header;`

http_raw_cookie.with_body

ルールの別の部分で確認するのは`http_raw_cookie`ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_raw_cookie.with_body`パラメータはオプションです。

シンタックス：`http_raw_cookie: with_body;`

例：`http_raw_cookie: with_body;`

http_raw_cookie.with_trailer

ルールの別の部分で確認するのは`http_raw_cookie`ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_raw_cookie.with_trailer` パラメータはオプションです。

シンタックス：`http_raw_cookie: with_trailer;`

例：`http_raw_cookie: with_trailer;`

http_raw_header

検出カーソルを正規化されていないヘッダーに設定します。`http_raw_header` には、元のメッセージの変更されていないすべてのヘッダー名と値が含まれます。

`http_raw_header` ルールオプションには、パラメータの `http_raw_header.field`、`http_raw_header.request`、`http_raw_header.with_header`、`http_raw_header.with_body`、および `http_raw_header.with_trailer` が含まれます。

シンタックス：`http_raw_header: field <field_name>, <parameter>, <parameter>;`

例：`http_raw_header: field Content-Type, with_trailer;`

http_raw_header.field

指定したヘッダー名を正規化されていないHTTPヘッダーと照合します。ヘッダー名では大文字と小文字が区別されません。ヘッダー名を指定しなかった場合、HTTP インスペクタは、HTTP Cookie のヘッダー（Cookie と Set-Cookie）を除くすべてのヘッダーを確認します。

型：文字列

シンタックス：`http_raw_header: field <field_name>`

有効な値：HTTP ヘッダー名。

例：`http_raw_header: field Content-Type;`

http_raw_header.request

HTTP要求メッセージで検出されたヘッダーを照合します。応答メッセージを確認する場合は、HTTP 要求ヘッダーを使用します。`http_raw_header.request` パラメータはオプションです。

シンタックス：`http_raw_header: request;`

例：`http_raw_header: request;`

http_raw_header.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_raw_header.with_header` パラメータはオプションです。

シンタックス：`http_raw_header: with_header;`

例：`http_raw_header: with_header;`

http_raw_header.with_body

ルールの別の部分で確認するのは`http_raw_header`ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_raw_header.with_body`パラメータはオプションです。

シンタックス：`http_raw_header: with_body;`

例：`http_raw_header: with_body;`

http_raw_header.with_trailer

ルールの別の部分で確認するのは`http_raw_header`ルールオプションではなく、HTTP メッセージトレーラであることを指定します。`http_raw_header.with_trailer` パラメータはオプションです。

シンタックス：`http_raw_header: with_trailer;`

例：`http_raw_header: with_trailer;`

http_raw_request

検出カーソルを正規化されていない要求行に設定します。最初のヘッダー行の特定の部分を調べるには、ルールオプションの `http_method`、`http_raw_uri`、または `http_version` のいずれかを使用します。

`http_raw_request` ルールオプションには、パラメータの `http_raw_request.with_header`、`http_raw_request.with_body`、および `http_raw_request.with_trailer` が含まれます。

シンタックス：`http_raw_request: <parameter>, <parameter>;`

例：`http_raw_request: with_header;`

http_raw_request.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_raw_request.with_header` パラメータはオプションです。

シンタックス：`http_raw_request: with_header;`

例：`http_raw_request: with_header;`

http_raw_request.with_body

ルールの別の部分で確認するのは `http_raw_request` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_raw_request.with_body` パラメータはオプションです。

シンタックス：`http_raw_request: with_body;`

例： `http_raw_request: with_body;`

http_raw_request.with_trailer

ルールの別の部分で確認するのは `http_raw_request` ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_raw_request.with_trailer` パラメータはオプションです。

シンタックス：`http_raw_request: with_trailer;`

例：`http_raw_request: with_trailer;`

http_raw_status

検出カーソルを正規化されていないステータス行に設定します。ステータス行の特定の部分を調べるには、ルールオプションの `http_version`、`http_stat_code`、または `http_stat_msg` のいずれかのを使用します。

`http_raw_status` ルールオプションには、パラメータの `http_raw_status.with_body` と `http_raw_status.with_trailer` が含まれます。

シンタックス：`http_raw_status: <parameter>, <parameter>;`

例：`http_raw_status: with_body;`

http_raw_status.with_body

ルールの別の部分で確認するのは `http_raw_status` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_raw_status.with_body` パラメータはオプションです。

シンタックス：`http_raw_status: with_body;`

例：`http_raw_status: with_body;`

http_raw_status.with_trailer

ルールの別の部分で確認するのは http_raw_status ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。http_raw_status.with_trailer パラメータはオプションです。

シンタックス：http_raw_status: with_trailer;

例：http_raw_status: with_trailer;

http_raw_trailer

検出カーソルを正規化されていない HTTP トレーラに設定します。トレーラにはメッセージの内容に関する情報が含まれています。クライアント要求で HTTP ヘッダーを作成する場合、トレーラは使用できません。

http_raw_trailer は、終了ヘッダーに適用されることを除いて、http_raw_header と同じです。HTTP のヘッダーとトレーラを検査するには、個別のルールを作成する必要があります。

http_raw_trailer ルールオプションには、パラメータの http_raw_trailer.field、http_raw_trailer.request、http_raw_trailer.with_header、http_raw_trailer.with_body が含まれます。

シンタックス：http_raw_trailer: field <field_name>, <parameter>, <parameter>;

例：http_raw_trailer: field <field_name>, request;

http_raw_trailer.field

指定したトレーラ名を正規化されていない HTTP トレーラと照合します。トレーラ名では大文字と小文字が区別されません。

型：文字列

シンタックス：http_raw_trailer: field <field_name>;

有効な値：HTTP トレーラ名。

例： http_raw_trailer: field trailer-timestamp;

http_raw_trailer.request

HTTP 要求メッセージで検出されたトレーラを照合します。応答メッセージを確認する場合は、HTTP 要求トレーラを使用します。http_raw_trailer.request パラメータはオプションです。

シンタックス：http_raw_trailer: request;

例：http_raw_trailer: request;

http_raw_trailer.with_header

ルールで確認できるのは HTTP 応答ヘッダーのみであることを指定します。http_raw_trailer.with_header パラメータはオプションです。

シンタックス：http_raw_trailer: with_header;

例：`http_raw_trailer: with_header;`

http_raw_trailer.with_body

ルールの別の部分で確認するのは `http_raw_trailer` ルールオプションではなく、HTTP メッセージの本文を検査することを指定します。`http_raw_trailer.with_body` パラメータはオプションです。

シンタックス：`http_raw_trailer: with_body;`

例：`http_raw_trailer: with_body;`

http_raw_uri

検出カーソルを正規化されていない URI に設定します。

`http_raw_uri` ルールオプションには次のものが含まれます。

- `http_raw_uri.with_header`
- `http_raw_uri.with_body`
- `http_raw_uri.with_trailer`
- `http_raw_uri.scheme`
- `http_raw_uri.host`
- `http_raw_uri.port`
- `http_raw_uri.path`
- `http_raw_uri.query`
- `http_raw_uri.fragment`

シンタックス：`http_raw_uri: <parameter>, <parameter>;`

例：`http_raw_uri: with_header, path, query;`

http_raw_uri.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_raw_uri.with_header` パラメータはオプションです。

シンタックス：`http_raw_uri: with_header;`

例：`http_raw_uri: with_header;`

http_raw_uri.with_body

ルールの別の部分で確認するのは `http_raw_uri` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_raw_uri.with_body` パラメータはオプションです。

シンタックス：`http_raw_uri: with_body;`

例：`http_raw_uri: with_body;`

http_raw_uri.with_trailer

ルールの別の部分で確認するのは `http_raw_uri` ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_raw_uri.with_trailer` パラメータはオプションです。

シンタックス：`http_raw_uri: with_trailer;`

例：`http_raw_uri: with_trailer;`

http_raw_uri.scheme

URI のスキームに対してのみ照合します。`http_raw_uri.scheme` パラメータはオプションです。

シンタックス：`http_raw_uri: scheme;`

例：`http_raw_uri: scheme;`

http_raw_uri.host

URI のホスト（ドメイン名）に対してのみ照合します。`http_raw_uri.host` パラメータはオプションです。

シンタックス：`http_raw_uri: host;`

例：`http_raw_uri: host;`

http_raw_uri.port

URI のポート（TCP ポート）に対してのみ照合します。`http_raw_uri.port` パラメータはオプションです。

シンタックス：`http_raw_uri: port;`

例：`http_raw_uri: port;`

http_raw_uri.path

URI のパスセクション（ディレクトリとファイル）に対してのみ照合します。`http_raw_uri.path` パラメータはオプションです。

シンタックス：`http_raw_uri: path;`

例： `http_raw_uri: path;`

http_raw_uri.query

URI のクエリパラメータに対してのみ照合します。`http_raw_uri.query` パラメータはオプションです。

シンタックス：`http_raw_uri: query;`

例：`http_raw_uri: query;`

http_raw_uri.fragment

URI のフラグメントセクションに対してのみ照合します。フラグメントは、要求したファイルの一部であり、通常はブラウザ内でのみ検出され、ネットワーク経由では送信されません。`http_raw_uri.fragment` パラメータはオプションです。

シンタックス：`http_raw_uri: fragment;`

例：`http_raw_uri: fragment;`

http_stat_code

検出カーソルを HTTP ステータスコードに設定します。HTTP ステータスコードは、100 ～ 599 の範囲の 3 桁の数字です。

`http_stat_code` ルールオプションには、パラメータの `http_stat_code.with_body` と `http_stat_code.with_trailer` が含まれます。

シンタックス：`http_stat_code: <parameter>, <parameter>;`

例：`http_stat_code: with_trailer;`

http_stat_code.with_body

ルールの別の部分で確認するのは `http_stat_code` ルールオプションではなく、HTTP メッセージの本文を検査することを指定します。`http_stat_code.with_body` パラメータはオプションです。

シンタックス：`http_stat_code: with_body;`

例：`http_stat_code: with_body;`

http_stat_code.with_trailer

ルールの別の部分で確認するのは `http_stat_code` ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_stat_code.with_trailer` パラメータはオプションです。

シンタックス：`http_stat_code: with_trailer;`

例：`http_stat_code: with_trailer;`

http_stat_msg

検出カーソルを HTTP ステータスメッセージに設定します。HTTP ステータス メッセージは、HTTP ステータスコードをプレーンテキストで記述します（例：`OK`）。

`http_stat_msg` ルールオプションには、パラメータの `http_stat_msg.with_body` と `http_stat_msg.with_trailer` が含まれます。

シンタックス：`http_stat_msg: <parameter>, <parameter>;`

例：`http_stat_msg: with_body;`

http_stat_msg.with_body

ルールの別の部分で確認するのは `http_stat_msg` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_stat_msg.with_body` パラメータはオプションです。

シンタックス：`http_stat_msg: with_body;`

例：`http_stat_msg: with_body;`

http_stat_msg.with_trailer

ルールの別の部分で確認するのは `http_stat_msg` ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_stat_msg.with_trailer` パラメータはオプションです。

シンタックス：`http_stat_msg: with_trailer;`

例：`http_stat_msg: with_trailer;`

http_trailer

検出カーソルを正規化されたトレーラに設定します。トレーラにはメッセージの内容に関する情報が含まれています。クライアント要求でHTTPヘッダーを作成する場合、トレーラは使用できません。

`http_trailer` は、終了ヘッダーに適用されることを除いて、`http_header` と同じです。HTTP のヘッダーとトレーラを検査するには、個別のルールを作成する必要があります。

`http_trailer` ルールオプションには、パラメータの `http_trailer.field`、`http_trailer.request`、`http_trailer.with_header`、`http_trailer.with_body` が含まれます。

シンタックス：`http_trailer: field <field_name>, <parameter>, <parameter>;`

例：`http_trailer: field trailer-timestamp, with_body;`

http_trailer.field

指定したトレーラ名を正規化されたHTTPトレーラと照合します。トレーラ名では大文字と小文字が区別されません。

型：文字列

シンタックス：`http_trailer: field <field_name>;`

有効な値：HTTP トレーラ名。

例：`http_trailer: field trailer-timestamp;`

http_trailer.request

HTTP要求メッセージで検出されたトレーラを照合します。応答メッセージを確認する場合は、HTTP 要求トレーラを使用します。`http_trailer.request` パラメータはオプションです。

シンタックス：`http_trailer: request;`

例：`http_trailer: request;`

http_trailer.with_header

ルールの別の部分で確認するのは `http_trailer` ルールオプションではなく、HTTP メッセージのヘッダーであることを指定します。`http_trailer.with_header` パラメータはオプションです。

シンタックス：`http_trailer: with_header;`

例：`http_trailer: with_header;`

http_trailer.with_body

ルールの別の部分で確認するのは `http_trailer` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_trailer.with_body` パラメータはオプションです。

シンタックス：`http_trailer: with_body;`

例：`http_trailer: with_body;`

http_true_ip

検出カーソルを最終的なクライアント IP アドレスに設定します。クライアントが要求を送信すると、プロキシサーバーは最終的なクライアント IP アドレスを保存します。クライアント IP アドレスは、`X-Forwarded-For`、`True-Client-IP`、またはその他のカスタム `X-Forwarded-For` タイプのヘッダーにリストされている最後の IP アドレスです。複数のヘッダーが存在する場合、Snort は `xff_headers` で定義されたヘッダーを考慮します。

`http_true_ip` ルールオプションには、パラメータの `http_true_ip.with_header`、`http_true_ip.with_body`、および `http_true_ip.with_trailer` が含まれます。

シンタックス：`http_true_ip: <parameter>, <parameter>;`

例：`http_true_ip: with_header;`

http_true_ip.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_true_ip.with_header` パラメータはオプションです。

シンタックス：`http_true_ip: with_header;`

例：`http_true_ip: with_header;`

http_true_ip.with_body

ルールの別の部分で確認するのは `http_true_ip` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_true_ip.with_body` パラメータはオプションです。

シンタックス：`http_true_ip: with_body;`

例：`http_true_ip: with_body;`

http_true_ip.with_trailer

ルールの別の部分で確認するのは `http_true_ip` ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_true_ip.with_trailer` パラメータはオプションです。

シンタックス：`http_true_ip: with_trailer;`

例： `http_true_ip: with_trailer;`

http_uri

検出カーソルを正規化された URI バッファに設定します。

- `http_uri.with_header`
- `http_uri.with_body`
- `http_uri.with_trailer`
- `http_uri.scheme`
- `http_uri.host`
- `http_uri.port`
- `http_uri.path`
- `http_uri.query`
- `http_uri.fragment`

シンタックス：`http_uri: <parameter>, <parameter>;`

例：`http_uri: with_trailer, path, query;`

http_uri.with_header

ルールで確認できるのは HTTP メッセージのヘッダーのみであることを指定します。`http_uri.with_header` パラメータはオプションです。

シンタックス：`http_uri: with_header;`

例：`http_uri: with_header;`

http_uri.with_body

ルールの別の部分で確認するのは `http_uri` ルールオプションではなく、HTTP メッセージの本文であることを指定します。`http_uri.with_body` パラメータはオプションです。

シンタックス：`http_uri: with_body;`

例：`http_uri: with_body;`

http_uri.with_trailer

ルールの別の部分で確認するのは`http_uri`ルールオプションではなく、HTTP メッセージのトレーラであることを指定します。`http_uri.with_trailer` パラメータはオプションです。

シンタックス：`http_uri: with_trailer;`

例：`http_uri: with_trailer;`

http_uri.scheme

URI のスキームに対してのみ照合します。`http_uri.scheme` パラメータはオプションです。

シンタックス：`http_uri: scheme;`

例：`http_uri: scheme;`

http_uri.host

URI のホスト（ドメイン名）に対してのみ照合します。`http_uri.host` パラメータはオプションです。

シンタックス：`http_uri: host;`

例：`http_uri: host;`

http_uri.port

URI のポート（TCP ポート）に対してのみ照合します。`http_uri.port` パラメータはオプションです。

シンタックス：`http_uri: port;`

例：`http_uri: port;`

http_uri.path

URI のパス（ディレクトリとファイル）に対してのみ照合します。`http_uri.path` パラメータはオプションです。

シンタックス：`http_uri: path;`

例：`http_uri: path;`

http_uri.query

URI のクエリパラメータに対してのみ照合します。`http_uri.query` パラメータはオプションです。

シンタックス：`http_uri: uri;`

例：`http_uri: query;`

http_uri.fragment

URIのフラグメントセクションに対してのみ照合します。フラグメントは、要求したファイルの一部であり、通常はブラウザ内でのみ検出され、ネットワーク経由では送信されません。`http_uri.fragment` パラメータはオプションです。

シンタックス：`http_uri: fragment;`

例：`http_uri: fragment;`

http_version

検出カーソルをHTTPバージョンバッファの先頭に設定します。`http_version`は、さまざまなHTTPバージョンを受け入れます。最も一般的に見られるバージョンは `HTTP/1.0` と `HTTP/1.1` です。`http_version` ルールオプションには、パラメータの `http_version.request`、`http_version.with_header`、`http_version.with_body`、および `http_version.with_trailer` が含まれます。

シンタックス：`http_version: <parameter>, <parameter>;`

例：`http_version; content:"HTTP/1.1";`

http_version.request

HTTP要求で検出されたバージョンを照合します。応答メッセージを確認する場合は、要求バージョンを使用します。`http_version.request` パラメータはオプションです。

シンタックス：`http_version: request;`

例：`http_version: request;`

http_version.with_header

ルールで確認できるのはHTTPメッセージのヘッダーのみであることを指定します。`http_version.with_header` パラメータはオプションです。

シンタックス：`http_version: with_header;`

例：`http_version: with_header;`

http_version.with_body

ルールの別の部分で確認するのは`http_version`ルールオプションではなく、HTTPメッセージの本文であることを指定します。`http_version.with_body` パラメータはオプションです。

シンタックス：`http_version: with_body;`

例：`http_version: with_body;`

http_version.with_trailer

ルールの別の部分で確認するのは`http_version`ルールオプションではなく、HTTPメッセージのトレーラであることを指定します。`http_version.with_trailer` パラメータはオプションです。

シンタックス：`http_version: with_trailer;`

例：`http_version: with_trailer;`

http_version_match

標準の HTTP バージョンと照合する HTTP バージョンのリストを指定します。複数のバージョンは余白文字で区切ります。HTTP 要求またはステータス行には、バージョンが含まれる場合があります。そのバージョンが存在する場合、Snort はこのバージョンを `http_version_match` で指定されたリストと比較します。

そのバージョンの形式の `[0-9].[0-9]` がない場合は、不正な形式と見なされます。`1.0` または `1.1` ではない `[0-9].[0-9]` の形式のバージョンは、その他と見なされます。

型：文字列

シンタックス：`http_version_match: <version_list>`

有効な値：`1.0`、`1.1`、`2.0`、`0.9`、`other`、`malformed`

例：`http_version_match: "1.0 1.1";`

js_data

検出カーソルを正規化された JavaScript データに設定します。このオプションは、拡張 JavaScript ノーマライザに固有です。

シンタックス：`js_data;`

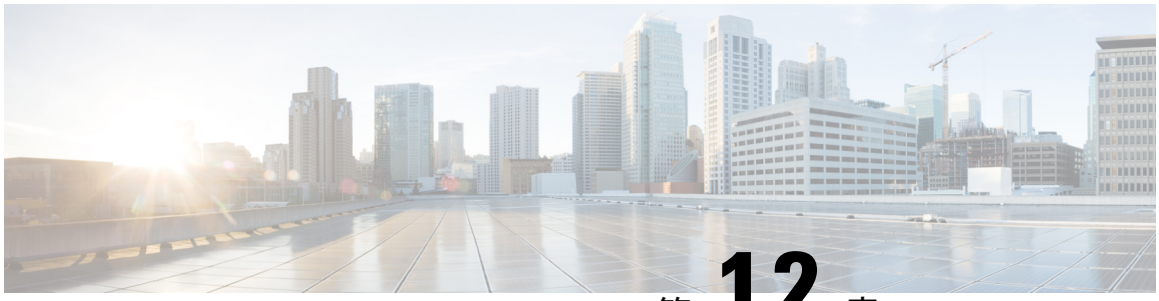
例：`js_data;`

vba_data

検出カーソルを Microsoft Office Visual Basic for Applications マクロバッファに設定します。

シンタックス：`vba_data;`

例：`vba_data;`



第 12 章

IEC104 インスペクタ

- IEC104 インスペクタの概要 （125 ページ）
- IEC104 インスペクタのパラメータ （126 ページ）
- IEC104 インスペクタのルール （126 ページ）
- IEC104 インスペクタの侵入ルールのオプション （130 ページ）

IEC104 インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`false`

IEC 60870-5-104（IEC104）プロトコルは、電力システム間で遠隔制御メッセージを交換するための通信規格について説明します。IEC104 プロトコルは TCP ポート 2404 を使用します。

`iec104` インスペクタは、ネットワークトラフィック内の IEC104 メッセージを検出します。`iec104` インスペクタは、複数のフレームにまたがるメッセージを組み合わせるか、または 1 つのフレーム内で複数のメッセージを分割することで、IEC104 メッセージを分析し、正規化します。

有効にすると、侵入ルールのオプションは、IEC104 アプリケーションプロトコル制御情報（APCI）タイプとアプリケーション サービス データ ユニット（ASDU）機能コードにアクセスできるようになります。

IEC104 インスペクタのパラメータ

IEC104 TCP ポートの設定

`binder` インスペクタは、IEC104 TCP ポートの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "role": "server",
            "proto": "tcp",
            "ports": "2404"
         },
        "use": {
            "type": "iec104"
        }
    }
]
```

（注） `iec104` インスペクタはパラメータを提供しません。

IEC104 インスペクタのルール

`iec104` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *15 : IEC104* インスペクタのルール

GID:SID	ルール メッセージ
151:1	IEC104 APCI ヘッダーの長さが、指定された IEC104 ASDU タイプ ID に必要な長さと一致しない（Length in IEC104 APCI header does not match the length needed for the given IEC104 ASDU type id）
151:2	IEC104 開始バイトが 0x68 と一致しない（IEC104 Start byte does not match 0x68）
151:3	予約済みの IEC104 ASDU タイプ ID は使用中になっている（Reserved IEC104 ASDU type id in use）
151:4	IEC104 APCI U 予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI U Reserved field contains a non-default value）

GID:SID	ルール メッセージ
151:5	IEC104 APCI U メッセージタイプが無効な値に設定された（IEC104 APCI U message type was set to an invalid value）
151:6	IEC104 APCI S 予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI S Reserved field contains a non-default value）
151:7	IEC104 APCI I 要素数がゼロに設定されている（IEC104 APCI I number of elements set to zero）
151:8	IEC104 APCI I 機能をサポートしていない ASDU に SQ ビットが設定されている（IEC104 APCI I SQ bit set on an ASDU that does not support the feature）
151:9	IEC104 APCI I 機能をサポートしていない ASDU で要素の数が 2 以上に設定されている（IEC104 APCI I number of elements set to greater than one on an ASDU that does not support the feature）
151:10	IEC104 APCI I 初期化の原因が予約値に設定されている（IEC104 APCI I Cause of Initialization set to a reserved value）
151:11	IEC104 APCI I 問い合わせコマンドの修飾子が予約済みの値に設定されている（IEC104 APCI I Qualifier of Interrogation Command set to a reserved value）
151:12	IEC104 APCI I カウンタ問い合わせコマンドの修飾子の要求パラメータが予約済みの値に設定されている（IEC104 APCI I Qualifier of Counter Interrogation Command request parameter set to a reserved value）
151:13	IEC104 APCI I 測定値のパラメータの修飾子のパラメータの種類が予約済みの値に設定されている（Qualifier of Parameter of Measured Values kind of parameter set to a reserved value）
151:14	IEC104 APCI I 測定値のパラメータの修飾子のローカルパラメータの変更が技術的に有効だが未使用の値に設定されている（Qualifier of Parameter of Measured Values local parameter change set to a technically valid but unused value）
151:15	IEC104 APCI I 測定値のパラメータの修飾子のパラメータオプションが技術的に有効だが未使用の値に設定されている（IEC104 APCI I Qualifier of Parameter of Measured Values parameter option set to a technically valid but unused value）
151:16	IEC104 APCI I パラメータアクティベーションの修飾子が予約済みの値に設定されている（IEC104 APCI I Qualifier of Parameter Activation set to a reserved value）

GID:SID	ルール メッセージ
151:17	IEC104 APCI I コマンドの修飾子が予約値に設定されている（Qualifier of Command set to a reserved value）
151:18	IEC104 APCI I リセットプロセスの修飾子が予約値に設定されている（Qualifier of Reset Process set to a reserved value）
151:19	IEC104 APCI I ファイルの準備完了修飾子が予約値に設定されている（File Ready Qualifier set to a reserved value）
151:20	IEC104 APCI I セクションの準備完了修飾子が予約済みの値に設定されている（Section Ready Qualifier set to a reserved value）
151:21	IEC104 APCI I 選択して呼び出しの修飾子が予約値に設定されている（Select and Call Qualifier set to a reserved value）
151:22	IEC104 APCI I 最終セクションまたはセグメントの修飾子が予約値に設定されている（Last Section or Segment Qualifier set to a reserved value）
151:23	IEC104 APCI I 応答確認のファイルまたはセクションの修飾子が予約済みの値に設定されている（Acknowledge File or Section Qualifier set to a reserved value）
151:24	IEC104 APCI I 構造修飾子が効力がないメッセージに設定されている（Structure Qualifier set on a message where it should have no effect）
151:25	IEC104 APCI I シングルポイント情報予約済みフィールドにデフォルト以外の値が含まれている（Single Point Information Reserved field contains a non-default value）
151:26	IEC104 APCI I ダブルポイント情報予約済みフィールドにデフォルト以外の値が含まれている（Double Point Information Reserved field contains a non-default value）
151:27	IEC104 APCI I 送信原因が予約値に設定されている（Cause of Transmission set to a reserved value）
151:28	IEC104 APCI I 送信の原因がASDU に許可されていない値に設定されている（Cause of Transmission set to a value not allowed for the ASDU）
151:29	IEC104 APCI I 無効な 2 オクテットの共通アドレス値が検出された（IEC104 APCI I invalid two octet common address value detected）
151:30	IEC104 APCI I 品質記述子構造の予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI I Quality Descriptor Structure Reserved field contains a non-default value）

GID:SID	ルール メッセージ
151:31	IEC104 APCI I 保護装置構造のイベントの品質記述子の予約済みフィールドにデフォルト以外の値が含まれている（Quality Descriptor for Events of Protection Equipment Structure Reserved field contains a non-default value）
151:32	IEC104 APCI I IEEE STD 754 値が NaN になる（IEEE STD 754 value results in NaN）
151:33	IEC104 APCI I IEEE STD 754 値が無限大になる（IEC104 APCI I IEEE STD 754 value results in infinity）
151:34	IEC104 APCI I 保護装置構造の単一イベントの予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI I Single Event of Protection Equipment Structure Reserved field contains a non-default value）
151:35	IEC104 APCI I 保護装置構造の開始イベントの予約フィールドにデフォルト以外の値が含まれている（IEC104 APCI I Start Event of Protection Equipment Structure Reserved field contains a non-default value）
151:36	IEC104 APCI I 出力回路情報構造の予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI I Output Circuit Information Structure Reserved field contains a non-default value）
151:37	IEC104 APCI I 異常な固定テストビットパターンが検出された（IEC104 APCI I Abnormal Fixed Test Bit Pattern detected）
151:38	IEC104 APCI I 単一コマンド構造の予約済みフィールドにデフォルト以外の値が含まれている（Single Command Structure Reserved field contains a non-default value）
151:39	IEC104 APCI I ダブルコマンド構造に無効な値が含まれている（Double Command Structure contains an invalid value）
151:40	IEC104 APCI I 規制ステップコマンド構造の予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI I Regulating Step Command Structure Reserved field contains a non-default value）
151:41	IEC104 APCI I Time2a ミリ秒が許容範囲外に設定されている（IEC104 APCI I Time2a Millisecond set outside of the allowable range）
151:42	IEC104 APCI I Time2a の分が許容範囲外に設定されている（IEC104 APCI I Time2a Minute set outside of the allowable range）
151:43	IEC104 APCI I Time2a の分の予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI I Time2a Minute Reserved field contains a non-default value）

GID:SID	ルール メッセージ
151:44	IEC104 APCI I Time2a の時間が許容範囲外に設定されている（IEC104 APCI I Time2a Hours set outside of the allowable range）
151:45	IEC104 APCI I Time2a の時間の予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI I Time2a Hours Reserved field contains a non-default value）
151:46	IEC104 APCI I Time2a の日が許容範囲外に設定されている（IEC104 APCI I Time2a Day of Month set outside of the allowable range）
151:47	IEC104 APCI I Time2a の月が許容範囲外に設定されている（IEC104 APCI I Time2a Month set outside of the allowable range）
151:48	IEC104 APCI I Time2a の月の予約済みフィールドにデフォルト以外の値が含まれている（IEC104 APCI I Time2a Month Reserved field contains a non-default value）
151:49	IEC104 APCI I Time2a の年が許容範囲外に設定されている（IEC104 APCI I Time2a Year set outside of the allowable range）
151:50	IEC104 APCI I Time2a 年の予約フィールドにデフォルト以外の値が含まれている（Time2a Year Reserved field contains a non-default value）
151:51	IEC104 APCI I セグメント長に null 値が検出された（IEC104 APCI I a null Length of Segment value has been detected）
151:52	IEC104 APCI I セグメント長に無効な値が検出された（IEC104 APCI I an invalid Length of Segment value has been detected）
151:53	IEC104 APCI I ファイルのステータスが予約済みの値に設定されている（IEC104 APCI I Status of File set to a reserved value）
151:54	IEC104 APCI I セットポイントコマンド ql フィールドの修飾子が予約済みの値に設定されている（IEC104 APCI I Qualifier of Set Point Command ql field set to a reserved value）

IEC104 インスペクタの侵入ルールのオプション

iec104_apci_type

IEC104 メッセージが、オプションで設定されている IEC104 アプリケーションプロトコル情報制御（APIC）タイプと一致することを確認します。

`iec104_apci_type` 侵入ルールオプションは、完全な APIC タイプ名、あるいは大文字または小文字の APIC タイプの省略形を使用して指定された文字列を受け入れます。

型：文字列

シンタックス： `iec104_apci_type: <apic_type>;`

例：

```
iec104_apci_type: unnumbered_control_function;
iec104_apci_type: S;
iec104_apci_type: I;
iec104_apci_type: i;
```

iec104_asdu_func

IEC104 メッセージが、オプションで設定された IEC104 アプリケーション サービス データ ユニット（ASDU）機能コードと一致していることを確認します。

`iec104_asdu_func` 侵入ルールオプションでは、大文字または小文字の ASDU 機能コードを使用して指定された文字列を使用できます。

型：文字列

シンタックス： `iec104_asdu_func: <asdu_func>;`

例：

```
iec104_asdu_func: M_SP_NA_1;
iec104_asdu_func: m_sp_na_1;
```

第 13 章

IMAP インスペクタ

- IMAP インスペクタの概要 （133 ページ）
- IMAP インスペクタのパラメータ （134 ページ）
- IMAP インスペクタのルール （136 ページ）
- IMAP インスペクタの侵入ルールのオプション （137 ページ）

IMAP インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`true`

Internet Message Application Protocol（IMAP）は、電子メールクライアントを有効にし、リモート IMAP3 サーバーからメッセージを取得するようにします。IMAP3 サーバーは、安全でないセッションに TCP ポート 143 を使用するか、SSL/TLS を介した IMAP に TCP ポート 993 を使用します。

`imap` インスペクタは、IMAP トラフィックを検出し、IMAP コマンドと応答を分析します。

`imap` インスペクタは、IMAP メッセージのコマンド、ヘッダー、および本文のセクションを識別し、多目的のインターネットメール拡張（MIME）添付ファイルを抽出して復号化することができます。MIME 添付ファイルには、複数の添付ファイルや複数のパケットにまたがる大きな添付ファイルが含まれる場合があります。

`imap` インスペクタは、IMAP トラフィックを識別し、Snort 許可リストに追加します。有効にすると、侵入ルールは異常な IMAP トラフィックの発生時にイベントを生成します。

IMAP インスペクタのパラメータ

IMAP サービスの設定

binder インスペクタは、IMAP サービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "service": "imap",
            "role": any
        },
        "use": {
            "type": "imap"
        }
    }
]
```

b_64_decode_depth

各 Base64 エンコード MIME 電子メール添付ファイルから抽出してデコードできる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 141:4 を有効にしてこのパラメータのイベントを生成し、インライン展開で、（エンコードが正しくないか、またはデータが壊れているため）復号化が失敗した場合に問題のあるパケットをドロップすることができます。

型：整数

有効な範囲：-1 ～ 65535

デフォルト値：-1

bitenc_decode_depth

バイトの最大数を指定し、エンコードされていない各 MIME 電子メールの添付ファイルから抽出します。65535 未満の整数を指定するか、または 0 を指定して、エンコードされていない MIME 添付ファイルの抽出を無効にすることができます。抽出するバイト数に制限を設定しない場合は、-1 を指定します。これらの添付ファイルのタイプには、7 ビット、8 ビット、バイナリ、ならびにプレーンテキスト、JPEG イメージと PNG イメージ、および MP4 ファイルなど、マルチパートのコンテンツタイプが含まれます。

型：整数

有効な範囲：-1 ～ 65535

デフォルト値：-1

decompress_pdf

MIME 添付ファイルの `application/pdf` （PDF） ファイルを圧縮解除するかどうかを指定します。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 141:8 を有効にします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_swf

MIME 添付ファイルの `application/vnd.adobe.flash-movie` （SWF）の圧縮を解除するかどうかを指定します。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 141:8 を有効にします。

型：整数

有効な値：`true`、`false`

デフォルト値：`false`

decompress_vba

MIME 添付ファイルの Microsoft Office Visual Basic for Applications のマクロファイルの圧縮を解除するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_zip

MIME 添付ファイルのアプリケーション/zip（ZIP）ファイルを解凍するかどうかを指定します。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 141:8 を有効にします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

qp_decode_depth

各 quoted-printable（QP）エンコード MIME 電子メール添付ファイルから抽出してデコードできる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 141:5 を有効にしてこのパラメータのイベントを生成し、インライン展開で、（エンコードが正しくないか、またはデータが壊れていることが原因で）復号化が失敗した場合に問題のあるパケットをドロップすることができます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

uu_decode_depth

各 Unix-to-Unix エンコード（UU エンコード）MIME 電子メール添付ファイルから抽出して復号化できる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 141:7 を有効にしてこのパラメータのイベントを生成し、インライン展開で、（エンコードが正しくないか、またはデータが破損していることが原因で）復号化が失敗したときに問題のあるパケットをドロップできます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

IMAP インスペクタのルール

`imap` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *16 : IMAP* インスペクタのルール

GID:SID	ルール メッセージ
141:1	不明な IMAP3 コマンド（unknown IMAP3 command）
141:2	不明な IMAP3 応答（unknown IMAP3 response）
141:4	Base64 の復号化に失敗した（base64 decoding failed）
141:5	Quoted-Printable の復号化に失敗した（quoted-printable decoding failed）
141:7	Unix-to-Unix の復号化に失敗した（Unix-to-Unix decoding failed）

GID:SID	ルール メッセージ
141:8	ファイルの圧縮解除に失敗した（file decompression failed）

IMAP インスペクタの侵入ルールのオプション

vba_data

検出カーソルを Microsoft Office Visual Basic for Applications マクロバッファに設定します。

シンタックス：`vba_data;`

例：`vba_data;`

第 14 章

MMS インスペクタ

- MMS インスペクタの概要 （139 ページ）
- MMS インスペクタのパラメータ （140 ページ）
- MMS インスペクタのルール （140 ページ）
- MMS インスペクタの侵入ルールのオプション （140 ページ）

MMS インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`false`

IEC 61850 は、電力システムの通信プロトコルを定義する国際規格です。Manufacturing Message Specification（MMS）プロトコルは、IEC 61850 プロトコルの 1 つです。MMS を使用すると、さまざまな製造およびプロセス制御のデバイス間で遠隔監視制御・情報取得（SCADA）データをリアルタイムで転送できます。MMS プロトコルは、TCP ポート 102 を使用して、クライアントデバイスとサーバーデバイス間でメッセージを交換します。

`mms` インスペクタは、MMS トラフィックを検出および分析します。MMS メッセージには、1 つの TCP パケット内に複数のプロトコルデータユニット（PDU）、複数の TCP パケットに分割された 1 つの PDU、または 2 つのメッセージ設定の組み合わせが含まれる場合があります。`mms` インスペクタは、MMS トラフィックを正規化して完全な MMS メッセージをデバイスに提示します。

MMS プロトコルをデコードせずに、MMS メッセージの Snort 3 ルールを作成します。`mms` インスペクタは、MMS プロトコルをカプセル化する OSI レイヤーを分析し、ルールオプションを通じて特定の MMS プロトコルフィールドとデータコンテンツへのアクセスを提供します。

MMS ルールのオプションの詳細については、MMS インスペクタの侵入ルールのオプション（140 ページ） を参照してください。

MMS インスペクタのパラメータ

MMS サービスの設定

binder インスペクタは MMS サービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "service": "mms"
        },
        "use": {
            "type": "mms"
        }
    }
]
```

MMS インスペクタのルール

mms インスペクタには関連付けられたルールがありません。

MMS インスペクタの侵入ルールのオプション

mms_data

検出カーソルの位置を MMS プロトコルデータユニット（PDU）の先頭に配置し、すべての OSI カプセル化層をバイパスします。侵入ルールに mms_data が含まれている場合、ルールの次のルールオプションは MMS PDU から処理を開始します。

シンタックス：mms_data;

例：

次のサンプル侵入ルールは、mms_data ルールオプションを設定します。mms_data ルールオプションは、検出カーソルを MMS PDU の先頭に配置し、その位置のバイトで Initiate-Request メッセージの値をチェックします。

```
alert tcp ( \
msg: "PROTOCOL-SCADA MMS Initiate-Request"; \
flow: to_server, established; \
mms_data; \
content:"|A8|", depth 1; \
```

```
sid:1000000; \
)
```

mms_func

提供された関数名または番号を、MMS要求または応答の`Confirmed Service`フィールドと比較します。MMS 機能の名前または番号が `Confirmed Service` と一致したときに警告します。

タイプ：文字列

シンタックス：`mms_func <function>;`

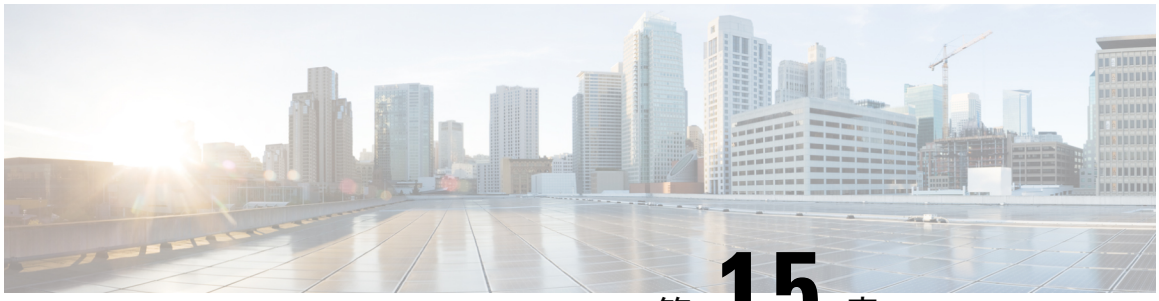
例：

次の侵入ルールの例では、`mms_func` ルールオプションを設定し、`Confirmed Service Request` サービスが提供された関数名と一致した場合に警告します。さらに、`mms_func` は、`Confirmed Service Request (0xA0)` メッセージで一致する高速パターンマッチング機能を有効にします。

```
alert tcp ( \
msg: "PROTOCOL-SCADA MMS svc get_name_list"; \
flow: to_server, established; \
content:"|A0|"; \
mms_func: get_name_list; \
sid:1000000; \
)
```

次のサンプル侵入ルールは、`mms_func` ルールオプションを設定し、`GetNameList` メッセージが関数番号と一致したときに警告します。

```
alert tcp ( \
msg: "PROTOCOL-SCADA MMS svc get_name_list"; \
flow: to_server, established; \
content:"|A0|"; \
mms_func:1; \
sid:1000001; \
)
```



第 15 章

Modbus インスペクタ

- Modbus インスペクタの概要 （143 ページ）
- Modbus インスペクタを設定するためのベストプラクティス （144 ページ）
- Modbus インスペクタのパラメータ （144 ページ）
- Modbus インスペクタのルール （144 ページ）
- Modbus インスペクタの侵入ルールのオプション （145 ページ）

Modbus インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`false`

Modbus プロトコルは、遠隔監視制御・情報取得（SCADA）システムとプログラマブル自動化コントローラ（PLC）の間でメッセージを交換するための通信規格を定義します。Modbus プロトコルは TCP ポート 502 を使用します。

`modbus` インスペクタは、ネットワークトラフィック内の Modbus メッセージを検出して分析します。有効にすると、Modbus 侵入ルールのオプションを通じて特定の Modbus プロトコル フィールドにアクセスできます。

Modbusインスペクタを設定するためのベストプラクティス

ネットワークに有効になっている Modbus デバイスが含まれていない場合は、トラフィックに適用するネットワーク分析ポリシーの modbus インスペクタを有効にする必要があります。

Modbus インスペクタのパラメータ

Modbus TCP ポートの設定

binder インスペクタは、Modbus TCP ポートの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "role": "server",
            "proto": "tcp",
            "ports": "502"
        },
        "use": {
            "type": "modbus"
        }
    },
    {
        "when": {
            "role": "any",
            "service:" "modbus"
        },
        "use": {
            "type":"modbus"
        }
    }
]
```

（注） modbus インスペクタはパラメータを提供しません。

Modbus インスペクタのルール

modbus インスペクタ ルールを有効に、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *17 : Modbus* インスペクタのルール

GID:SID	ルール メッセージ
144:1	Modbus MBAP ヘッダーの長さが、指定した機能に必要な長さと一致しない（length in Modbus MBAP header does not match the length needed for the given function）
144:2	Modbus プロトコル ID がゼロでない（Modbus protocol ID is non-zero）
144:3	予約済み Modbus 機能コードは使用中になっている（reserved Modbus function code in use）

Modbus インスペクタの侵入ルールのオプション

`modbus` オプションは、単独で使用することも、`content` および `byte_jump` 侵入ルールのオプションと組み合わせて使用することもできます。

modbus_data

データカーソルを Modbus `Data` フィールドの先頭に設定します。

シンタックス：`modbus_data;`

例：`modbus_data;`

modbus_func

Modbus `Function` フィールドが指定した Modbus 機能コードと一致していることを確認します。Modbus 関数コードを表す正の整数または文字列リテラルを設定できます。

型：文字列

シンタックス：`modbus_func: <function>;`

有効な値は、次のとおりです。

表 *18 : Modbus* 関数コード値

コード	文字列
1	`read_coils`
2	`read_discrete_inputs`
3	`read_holding_registers`
4	`read_input_registers`
5	`write_single_coil`
6	`write_single_register`

コード	文字列
7	read_exception_status
8	diagnostics
11	get_comm_event_counter
12	get_comm_event_log
15	write_multiple_coils
16	write_multiple_registers
17	report_slave_id
20	read_file_record
21	write_file_record
22	mask_write_register
23	read_write_multiple_registers
24	read_fifo_queue
43	encapsulated_interface_transport

例：

```
modbus_func: read_coils;
modbus_func: 8;
```

modbus_unit

メッセージ内の Modbus ユニット ID が指定されたユニット ID と一致することを確認します。Modbus ユニット ID を表す数値を設定できます。

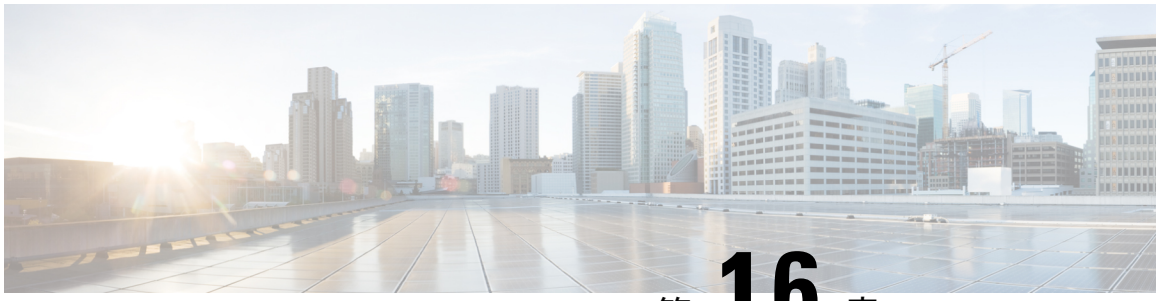
型：整数

シンタックス：`modbus_unit: <unit_id>;`

有効な範囲：`0` 〜 `255`

例：

```
modbus_unit: 1;
```



第 16 章

Normalizer インスペクタ

- ノーマライザインスペクタの概要 （147 ページ）
- Normalizer インスペクタのパラメータ （148 ページ）
- Normalizer インスペクタのルール （153 ページ）
- Normalizer インスペクタの侵入ルールのオプション （153 ページ）

ノーマライザインスペクタの概要

タイプ	インスペクタ（パケット）
使用方法	コンテキスト
インスタンス タイプ	ネットワーク
その他のインスペクタが必要	なし
有効	`true`

`normalizer` インスペクタは、パケット内のプロトコルの異常を検出して削除します。`normalizer` インスペクタを使用することで、インライン展開での検出を回避するために攻撃者がパケットを作成する可能性を最小限に抑えることができます。

（注） トラフィックをネットワークから送信する前に、ルーテッド、スイッチド、または透過的なインターフェイスあるいはインライン インターフェイス ペアを使用して、関連する設定を管理対象デバイスに展開する必要があります。

パケット内に IPv4、IPv6、ICMPv4、ICMPv6、および TCP プロトコルの組み合わせの正規化を指定できます。`normalizer` インスペクタは、パケットごとの正規化を実行し、ほとんどの正規化を処理します。`stream_tcp` インスペクタは TCP ペイロードの正規化を含む TCP の状態関連のパケットとストリームの正規化を処理します。

インライン正規化は、復号化の直後で他のインスペクタによる処理の直前に実行されます。正規化は、パケット層の内部から外部への方向で行われます。

normalizer インスペクタはイベントを生成しません。normalizer インスペクタは、他のインスペクタやインライン展開で使用できるようにパケットを作成します。インスペクタは、システムが処理するパケットがネットワーク上のホストで受信されるパケットと同じものになるようにします。

Normalizer インスペクタのパラメータ

設定内で normalizer の範囲を見つけ、normalizer インスペクタのパラメータを設定します。

ip6

IPv6 トラフィックの Reserved フラグをクリアします。

型：ブール値

有効な値：true、false

デフォルト値：false

icmp4

ICMPv4 トラフィックの Reserved フラグをクリアします。

型：ブール値

有効な値：true、false

デフォルト値：false

icmp6

ICMPv6 トラフィックの Reserved フラグをクリアします。

型：ブール値

有効な値：true、false

デフォルト値：false

ip4.base

[IPv4 フラグ（IPv4 Flags）] ヘッダーフィールドの単一ビットの [予約済み（Reserved）] サブフィールドとともにパラメータパディングをクリアします。緊急のポインター/フラグの問題を修正します。ip4.base を有効にすることをお勧めします。

型：ブール値

有効な値：true、false

デフォルト値：false

ip4.df

[IPv4 フラグ（IPv4 Flags）] ヘッダーフィールドの単一ビットの [フラグメント禁止（Don't Fragment）] サブフィールドをクリアします。`ip4.df` を有効にして、ダウンストリームルーターがパケットをドロップするのではなくフラグメント化できるようにします。`ip4.df` パラメータは、ドロップするパケットを作成する回避が行われないようにすることができます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

ip4.rf

着信パケットの予約ビットをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

ip4.tos

1 バイトの [差別化サービス（Differentiated Services）]（旧称 [タイプオブサービス（Type of Service）]）フィールドをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

ip4.trim

過剰なペイロードを持つパケットを、IP ヘッダーに指定されたデータグラム長にレイヤ 2（たとえば、イーサネット）ヘッダーを合計した長さにまで切り捨てます。ただし、最小フレーム長より小さく切り捨てることはしません。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.base

TCP ヘッダーの単一ビットの [予約済み（Reserved）] サブフィールドとオプションのパディングバイトをクリアします。緊急ポインタまたは緊急フラグの問題を修正します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.block

TCP 正規化中にパケットをドロップするかどうかを指定します。

有効にすると、Snortは無効になり受信ホストによってブロックされる可能性が高い異常なTCPパケット（正規化されている場合）をブロックします。たとえば、Snortは確立されたセッションの後に送信された SYN パケットをブロックします。

ルールが有効にされているかどうかに関係なく、Snort は次の TCP ストリームインスペクタのいずれかのルールに一致するすべてのパケットをドロップします。

- 129:1
- 129:3
- 129:4
- 129:6
- 129:8
- 129:11
- 129:14 ～129:19

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.ecn

`Explicit Congestion Notification (ECN)` フラグのパケット単位またはストリーム単位の正規化を次のように有効にします。

- [パケット（Packet）] を指定すると、ネゴシエーションに関係なく、パケット単位でECNフラグがクリアされます。
- [ストリーム（Stream）] を指定すると、ECNの使用がネゴシエートされていない場合、ストリーム単位で ECN フラグがクリアされます。[ストリーム（Stream）] を指定した場合に正規化を行うには、TCP ストリーム インスペクタで `tcp.require_3whs` を有効にする必要があります。
- `tcp.ecn` パラメータを無効にするには、`off` を指定します。

型：列挙体

有効な値：`off`、`packet`、`stream`

デフォルト値：`off`

tcp.ips

再送信されるデータの一貫性が確保されるように[TCPデータ（TCP Data）]フィールドの正規化を有効にします。正しく再構成できないセグメントはすべてドロップされます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`true`

tcp.opts

トラフィックで許可する特定の TCP オプションを正規化するかどうかを指定します。Snort では、明示的に許可したオプションは正規化されません。Snort では、明示的に許可していないオプションが正規化されます。

Snort では、最適な TCP パフォーマンスを実現するために一般的に使用されている次の TCP のオプションが常に許可されます。

- 最大セグメント サイズ（MSS）（Maximum Segment Size (MSS)）
- ウィンドウ スケール（Window Scale）
- タイム スタンプ TCP（Time Stamp TCP）

他のそれほど一般的に使用されないオプションについては、Snort は自動的に許可しません。

`tcp.opts` が有効になっている場合、TCP トラフィックの正規化には次のものが含まれます。

- MSS、ウィンドウスケール、タイムスタンプ、および明示的に許可されたすべてのオプションを除き、すべてのオプションのバイトを [操作なし（No Operation）]（TCP オプション 1）に設定します。
- タイムスタンプは存在していても無効な場合、あるいは有効であってもネゴシエートされない場合、タイムスタンプオクテットを [操作なし（No Operation）] に設定します。
- タイムスタンプがネゴシエートされるものの、存在しない場合、パケットをブロックします。
- 確認応答（ACK）制御ビットが設定されていない場合、[タイムスタンプエコー応答（TSecr）（Time Stamp Echo Reply (TSecr)）] オプションフィールドをクリアします。
- SYN 制御ビットが設定されていない場合、[MSS] オプションと [ウィンドウスケール（Window Scale）] オプションを [操作なし（No Operation）]（TCP オプション 1）に設定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.pad

オプションのパディングバイトをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.req_pay

ペイロードがない場合、TCP ヘッダー [緊急ポインタ（Urgent Pointer）] フィールドと緊急（URG）制御ビットをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.req_urg

TCP ヘッダーの緊急（URG）制御ビットが設定されていない場合は、16 ビットの TCP ヘッダーの [緊急ポインタ（Urgent Pointer）] フィールドをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.req_urp

TCP ヘッダーの [緊急ポインタ（Urgent Pointer）] フィールドが設定されていない場合は、TCP ヘッダーの `urgent (URG)` 制御ビットをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.resv

TCP ヘッダーの `Reserved` ビットをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.trim_mss

ペイロードが MSS より長い場合、TCP の [データ（Data）] フィールドを最大セグメントサイズ（MSS）にまで切り捨てます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.trim_rst

RST パケットからデータをクリアします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.trim_syn

TCP 同期（SYN）パケット内のデータを削除します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.trim_win

TCP の [データ（Data）] フィールドを [ウィンドウ（Window）] フィールドに指定されたサイズにまで切り捨てます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp.urp

ポインタがペイロード長を上回る場合、2 バイトの TCP ヘッダーの [緊急ポインタ（Urgent Pointer）] フィールドをペイロード長に設定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

Normalizer インスペクタのルール

`normalizer` インスペクタには、関連付けられたルールがありません。

Normalizer インスペクタの侵入ルールのオプション

`normalizer` インスペクタには、侵入ルールのオプションはありません。

第 17 章

POP インスペクタ

- POP インスペクタの概要 （155 ページ）
- POP インスペクタのパラメータ （156 ページ）
- POP インスペクタのルール （158 ページ）
- POP インスペクタの侵入ルールのオプション （159 ページ）

POP インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`true`

Post Office Protocol バージョン 3（POP3）を使用すると、電子メールクライアントはリモートの POP3 サーバーからメッセージを取得できるようになります。POP3 サーバーは、安全でないセッションには TCP ポート 110 を使用し、POP over SSL/TLS には TCP ポート 995 を使用します。

`pop` インスペクタは、POP トラフィックを検出し、POP コマンドと応答を分析します。

`pop` インスペクタは、POP メッセージのコマンド、ヘッダー、および本文のセクションを識別し、Multi-purpose Internet Mail Extension（MIME）添付ファイルを抽出し、復号化することができます。`pop` インスペクタは、MIME 添付ファイルを処理します。これには、複数の添付ファイルや、複数のパケットにまたがる大きな添付ファイルが含まれます。

`pop` インスペクタは、POP メッセージを識別し、Snort 許可リストに追加します。有効にすると、侵入ルールは異常な POP トラフィックの発生時にイベントを生成します。

POP インスペクタのパラメータ

（注） 復号化またはMIME電子メール添付ファイルの復号化が不要な場合の抽出の場合は、複数の添付ファイルや複数のパケットにまたがる大きな添付ファイルが含まれている可能性があります。

最大値は、`b_64_decode_depth`、`bitnc_decode_depth`、`qp_decode_depth`、または`uu_decode_depth`パラメータの値が次の点で異なる場合に使用されます。

- デフォルトのネットワーク分析ポリシー
- 同じアクセス コントロール ポリシーのネットワーク分析ルールによって呼び出される、他のカスタム ネットワーク分析ポリシー

POP サービスの設定

`binder`インスペクタは、POPサービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
           "service": "pop",
           "role": any
        },
        "use": {
           "type": "pop"
        }
    }
]
```

b_64_decode_depth

各 Base64 エンコード MIME 電子メール添付ファイルから抽出してデコードできる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 142:4 を有効にして、このパラメーターのイベントを生成し、インライン展開で、デコードが失敗したときに問題のあるパケットをドロップすることができます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

bitenc_decode_depth

バイトの最大数を指定し、エンコードされていない各MIME電子メールの添付ファイルから抽出します。65535 未満の整数を指定するか、または 0 を指定して、エンコードされていない MIME添付ファイルの抽出を無効にすることができます。抽出するバイト数に制限を設定しない場合は、-1 を指定します。これらの添付ファイルのタイプには、7 ビット、8 ビット、バイナリ、ならびにプレーンテキスト、JPEG イメージと PNG イメージ、および MP4 ファイルなど、マルチパートのコンテンツタイプが含まれます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

decompress_pdf

MIME 添付ファイルの `application/pdf` （PDF） ファイルを圧縮解除するかどうかを指定します。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 142:8 を有効にします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_swf

MIME 添付ファイルの `application/vnd.adobe.flash-movie` （SWF）の圧縮を解除するかどうかを指定します。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 142:8 を有効にします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_vba

MIME 添付ファイルの Microsoft Office Visual Basic for Applications のマクロファイルの圧縮を解除するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_zip

MIME 添付ファイルのアプリケーション/zip（ZIP）ファイルを解凍するかどうかを指定します。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 142:8 を有効にします。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

qp_decode_depth

各 quoted-printable（QP）エンコード MIME 電子メール添付ファイルから抽出してデコードできる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 142:5 を有効にしてこのパラメータのイベントを生成し、インライン展開で、（エンコードが正しくないか、またはデータが壊れていることが原因で）復号化が失敗した場合に問題のあるパケットをドロップすることができます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

uu_decode_depth

各 Unix-to-Unix エンコード（UU エンコード）MIME 電子メール添付ファイルから抽出して復号化できる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 142:7 を有効にしてこのパラメータのイベントを生成し、インライン展開で、（エンコードが正しくないか、またはデータが壊れていることが原因で）復号化が失敗した場合に問題のあるパケットをドロップすることができます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

POP インスペクタのルール

`pop` インスペクタ ルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *19 : POP* インスペクタのルール

GID:SID	ルール メッセージ
142:1	不明な POP3 コマンド（unknown POP3 command）
142:2	不明な POP3 応答（unknown POP3 response）
142:4	Base64 の復号化に失敗した（base64 decoding failed）
142:5	Quoted-Printable の復号化に失敗した（quoted-printable decoding failed）
142:7	Unix-to-Unix の復号化に失敗した（Unix-to-Unix decoding failed）
142:8	ファイルの圧縮解除に失敗した（file decompression failed）

POP インスペクタの侵入ルールのオプション

vba_data

検出カーソルを Microsoft Office Visual Basic for Applications マクロバッファに設定します。

シンタックス：`vba_data;`

例：`vba_data;`

第 18 章

ポートスキャンインスペクタ

- ポートスキャンインスペクタの概要 （161 ページ）
- ポートスキャンインスペクタを設定するためのベストプラクティス （164 ページ）
- ポートスキャンインスペクタのパラメータ （165 ページ）
- ポートスキャンインスペクタのルール （175 ページ）
- ポートスキャンインスペクタの侵入ルールのオプション （177 ページ）

ポートスキャンインスペクタの概要

タイプ	インスペクタ（プローブ）
使用方法	グローバル
インスタンス タイプ	グローバル
その他のインスペクタが必要	なし
有効	false

ポートスキャンとは、攻撃者が攻撃の準備段階としてよく使用する、ネットワーク調査の形式です。ポートスキャンでは、攻撃者はターゲットホスト上のネットワークプロトコルとサービスをプローブするように設計されたパケットを送信します。攻撃者は、ホストが応答で送信したパケットを確認することで、ホスト上のどのポートが開かれているか、または開かれているポートでどのアプリケーションプロトコルが実行されているかを直接あるいは推論によって判断できます。

ポートスキャン自体は攻撃の証拠になりません。ネットワーク上の正当なユーザーが、攻撃者が使用するのと同様のポートスキャン技術を使用している可能性があります。

`port_scan` インスペクタは、4 種類のポートスキャンを検出し、TCP、UDP、ICMP、および IP プロトコルでの接続試行をモニタします。`port_scan` インスペクタは、アクティビティのパターンを検出することで、どのポートが悪意のあるポートであるかを判断するのに役立ちます。

表 *20* : ポートスキャンプロトコルのタイプ

プロトコル	説明
TCP	TCP プローブを検出します。たとえば、SYN スキャン、ACK スキャン、TCP connect() スキャン、および（Xmas tree、FIN、NULL）といった異常なフラグを組み合わせたスキャンなどです。
UDP	ゼロバイト UDP パケットなどの UDP プローブを検出します。
ICMP	ICMP エコー要求（ping）を検出します。
IP	IP プロトコル スキャンを検出します。Snort は、開いているポートを探すのではなく、ターゲットホストでサポートされている IP プロトコルを検索します。

一般に、ターゲットホストの数、スキャン側ホストの数、およびスキャン対象のポートの数に応じて、ポートスキャンは 4 つのタイプに分けられます。

表 *21* : ポートスキャン タイプ

タイプ	説明
ポートスキャン	攻撃者が少数のホストを使用して、1 つの対象ホスト上で複数のポートをスキャンする 1 対 1 ポートスキャン。 1 対 1 ポートスキャンには次のような特徴があります。 • 少数のホストを使用してスキャン • 単一のホストをスキャン • 多数のポートをスキャン ポートスキャンでは、TCP、UDP、および IP のポートスキャンが検出されます。
ポートスイープ	攻撃者が少数のホストを使用して、複数の対象ホスト上で 1 つのポートをスキャンする 1 対多のポートスイープ。 ポートスイープには次のような特徴があります。 • 少数のホストを使用してスキャン • 多数のホストをスキャン • 少数の固有のポートをスキャン ポートスイープでは、TCP、UDP、ICMP、および IP のポートスイープが検出されます。

タイプ	説明
デコイ ポートスキャン	攻撃者がスプーフィングされた送信元IPアドレスと実際にスキャンされたIPアドレスとを組み合わせた1対1ポートスキャン。 デコイポートスキャンには次のような特徴があります。 • 多数のホストを使用してスキャン • 少数のポートを一度だけスキャン • 単一（または少数）のホストをスキャン デコイポートスイープでは、TCP、UDP、およびIPのプロトコルポートスキャンが検出されます。
分散型ポートスキャン	複数のホストが開いているポストに対して1つのホストをクエリする多対1のポートスキャン。 分散型ポートスキャンには次のような特徴があります。 • 多数のホストを使用してスキャン • 多数のポートを一度だけスキャン • 単一（または少数）のホストをスキャン 分散型ポートスキャンでは、TCP、UDP、およびIPのプロトコルポートスキャンが検出されます。

ポートスキャン感度のレベル

`port_scan` インスペクタは、3つのレベルのデフォルトのスキャン感度を備えています。

- default_low_port_scan
- default_med_port_scan
- default_high_port_scan

さまざまなフィルタを使用して、追加のスキャン感度レベルを構成できます。

- scans
- rejects
- nets
- ports

`port_scan` インスペクタは、プローブされたホストから否定応答を収集することで、プローブについて学習します。たとえば、Webクライアントが TCP を使用して Web サーバーに接続している場合、そのクライアントは Web サーバーがポート 80 でリスニングしていると想定できます。ただし、攻撃者がサーバーをプローブする場合、そのサーバーが Web サービスを提供

するかどうかを攻撃者は事前に知っているわけではありません。port_scan インスペクタは否定応答（つまり、ICMP 到達不能または TCP RST パケット）を検出すると、その応答を潜在的ポートスキャンとして記録します。否定応答をフィルタリングするデバイス（ファイアウォールやルータなど）の向こう側にターゲットホストがある場合、このプロセスはさらに困難になります。この場合、port_scan インスペクタは、選択した機密レベルに基づいてフィルタ処理されたポートスキャンイベントを生成することができます。

ポートスキャンインスペクタを設定するためのベストプラクティス

ポートスキャンの検出を最適化するには、port_scan インスペクタをネットワークに合わせて調整することをお勧めします。

- watch_ip パラメータは慎重に設定してください。watch_ip パラメータは、port_scan インスペクタがネットワーク上で非常にアクティブな正当なホストをフィルタ処理するのに役立ちます。最も一般的な例には、NAT IP、DNS キャッシュサーバー、syslog サーバー、および nfs サーバーがあります。
- port_scan インスペクタが生成する可能性のある誤検知のほとんどは、フィルター処理済みスキャンの alert タイプです。alert タイプは、特定の期間にホストが過度にアクティブであったことを示している場合があります。ホストがフィルタ処理済みスキャンの alert タイプを継続的に生成する場合は、ホストを ignore_scanners リストに追加するか、スキャン感度のレベルを低くします。
- プライオリティカウント、接続数、IP 数、ポート数、IP 範囲、およびポート範囲を使用して、誤検知を特定します。 誤検知を判断する最も簡単な方法は、単純な比率の推定です。次に、推定する比率と、誤検知ではなく正当なスキャンを示す関連値のリストを示します。
  - 接続数/IP 数：この比率は、IP ごとの接続の推定平均を示します。ポートスキャンの場合、この比率は高くなります。ポートスイープの場合、この比率は低くなります。
  - ポート数/IP 数：この比率は、IP ごとに接続されているポートの推定平均を示します。ポートスキャンの場合はこの比率が高くなり、スキャンされたホストのポートは少数の IP で接続されていたことを示します。ポートスイープの場合はこの比率が低くなり、スキャンするホストは少数のポートに接続していても、多くのホスト上で接続していることを示します。
  - 接続数/ポート数：この比率は、ポートごとの接続の推定平均を示します。ポートスキャンの場合、この比率は低くなります。これは、各接続が異なるポートに対して行われたことを示しています。ポートスイープの場合、この比率は高くなります。これは、同じポートに多くの接続があったことを示しています。

  プライオリティカウントが高いほど、実際のポートスキャンまたはポートスイープである可能性が高くなります（ホストがファイアウォールで管理されている場合を除く）。

- ポートスキャンを検出できない場合は、スキャン感度レベルを下げることができます。スキャン感度レベルを高くすることで、最適な保護が得られます。スキャン感度レベルが低いと、エラー応答に基づいてアラートのみが生成され、フィルタ処理済みのスキャンは捕捉されません。スキャン感度レベルが低い場合のエラー応答は、ポートスキャンを示している可能性があり、感度レベルが低いことによって生成されるアラートは非常に正確であり、必要とする調整は最小限ですみます。フィルタ処理されたスキャンまたは感度レベルの高いスキャンには、誤検知の傾向があります。

ポートスキャンインスペクタのパラメータ

memcap

最大トラッカーメモリをバイト単位で指定します。

型：整数

有効な範囲：`1024 〜 9,007,199,254,740,992 (maxSZ)`

デフォルト値：`10,485,760`

protos

モニタするプロトコルを指定します。プロトコルの省略形の文字列を入力します。複数のプロトコルを指定するには、各プロトコルの省略形をスペースで区切ります。

型：文字列

有効な値：`tcp、udp、icmp、ip、all`

デフォルト値：`all`

scan_types

調べるポートスキャンのタイプを指定します。プロトコルの省略形の文字列を入力します。複数のプロトコルを指定するには、各プロトコル文字列をスペースで区切ります。

型：文字列

有効な値：`portscan、portsweep、decoy_portscan、distributed_portscan、all`

デフォルト値：`all`

watch_ip

監視するオプションのポートを持つ CIDR ブロックと IP のリストを指定します。

`watch_ip` が定義されていない場合、`port_scan` インスペクタはすべてのネットワークトラフィックを確認します。

型：文字列

有効な値：CIDR または IP アドレス、CIDR または IP アドレスのリスト

デフォルト値：なし

alert_all

確立したウィンドウ内のしきい値を超えるすべてのイベントについてアラートを生成するかどうかを指定します。`alert_all` が `false` に設定されている場合、`port_scan` インスペクタは、ウィンドウ内のしきい値を超える最初のイベントについてのみアラートを生成します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

include_midstream

オプションのポートが含まれた CIDR をリストするかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

tcp_decoy.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：`0` 〜 `65535`

デフォルト値：`15`

tcp_decoy.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：`0` 〜 `65535`

デフォルト値：`25`

tcp_decoy.scan

スキャンの試行回数を指定します。

型：整数

有効な範囲：`0` 〜 `65535`

デフォルト値：`100`

tcp_decoy.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

tcp_dist.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

tcp_dist.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

tcp_dist.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

tcp_dist.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

tcp_ports.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

tcp_ports.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

tcp_ports.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

tcp_ports.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

tcp_sweep.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

tcp_sweep.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

tcp_sweep.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

tcp_sweep.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_decoy.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

udp_decoy.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_decoy.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

udp_decoy.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_dist.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

udp_dist.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_dist.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

udp_dist.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_ports.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

udp_ports.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_ports.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

udp_ports.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_sweep.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

udp_sweep.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

udp_sweep.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

udp_sweep.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_decoy.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

ip_decoy.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_decoy.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

ip_decoy.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_dist.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

ip_dist.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_dist.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

ip_dist.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_sweep.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

ip_sweep.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_sweep.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

ip_sweep.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_proto.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

ip_proto.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

ip_proto.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

ip_proto.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

icmp_sweep.rejects

否定応答のスキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：15

icmp_sweep.ports

以前の試行からポート（またはプロトコル）が変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

icmp_sweep.scans

スキャンの試行回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：100

icmp_sweep.nets

以前の試行からアドレスが変更された回数を指定します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：25

tcp_window

Transmission Control Protocol（TCP）スキャンの検出間隔を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

udp_window

User Datagram Protocol（UDP）スキャンの検出間隔を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

ip_window

Internet Protocol（IP）スキャンの検出間隔を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

icmp_window

Internet Control Message Protocol（ICMP）スキャンの検出間隔を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

ポートスキャンインスペクタのルール

port_scan インスペクタ ルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

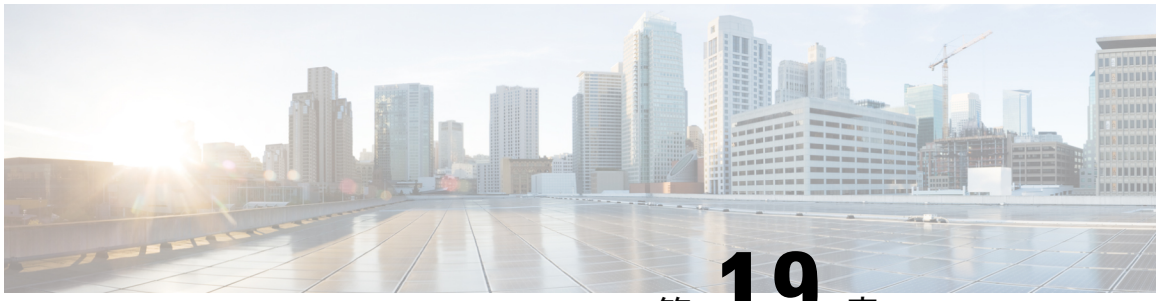
表 *22*：ポートスキャンインスペクタのルール

GID:SID	ルール メッセージ
122:1	TCP ポートスキャン（TCP portscan）
122:2	TCP デコイポートスキャン（TCP decoy portscan）
122:3	TCP ポートスイープ（TCP portsweep）
122:4	TCP 分散型ポートスキャン（TCP distributed portscan）
122:5	TCP フィルタ処理済みポートスキャン（TCP filtered portscan）
122:6	TCP フィルタ処理済みデコイポートスキャン（TCP filtered decoy portscan）
122:7	TCP フィルタ処理済みポートスイープ（TCP filtered portsweep）
122:8	TCP フィルタ処理済み分散型ポートスキャン（TCP filtered distributed portscan）
122:9	IP プロトコルスキャン（IP protocol scan）
122:10	IP デコイプロトコルスキャン（IP decoy protocol scan）
122:11	IP プロトコルスイープ（IP protocol sweep）
122:12	IP 分散型プロトコルスキャン（IP distributed protocol scan）
122:13	IP フィルタ処理済みプロトコルスキャン（IP filtered protocol scan）
122:14	IP フィルタ処理済みデコイプロトコルスキャン（IP filtered decoy protocol scan）
122:15	IP フィルタ処理済みプロトコルスイープ（IP filtered protocol sweep）
122:16	IP フィルタ処理済み分散型プロトコルスキャン（IP filtered distributed protocol scan）
122:17	UDP ポートスキャン（UDP portscan）
122:18	UDP デコイポートスキャン（UDP decoy portscan）
122:19	UDP ポートスイープ（UDP portsweep）
122:20	UDP 分散型ポートスキャン（UDP distributed portscan）
122:21	UDP フィルタ処理済みポートスキャン（UDP filtered portscan）
122:22	UDP フィルタ処理済みデコイポートスキャン（UDP filtered decoy portscan）
122:23	UDP フィルタ処理済みポートスイープ（UDP filtered portsweep）

GID:SID	ルール メッセージ
122:24	UDP フィルタ処理済み分散型ポートスキャン（UDP filtered distributed portscan）
122:25	ICMP スイープ（ICMP sweep）
122:26	ICMP フィルタ処理済みスイープ（ICMP filtered sweep）
122:27	オープンポート（open port）

ポートスキャンインスペクタの侵入ルールのオプション

`port_scan` インスペクタには、侵入ルールのオプションはありません。



第 19 章

レートフィルタ

- レートフィルタの概要 （179 ページ）
- レートフィルタのパラメータ （181 ページ）
- レートフィルタのルール （183 ページ）
- レートフィルタの侵入ルールのオプション （183 ページ）
- レートフィルタ操作の確認 （183 ページ）

レートフィルタの概要

タイプ	モジュール（基本）
使用方法	コンテキスト
インスタンス タイプ	単一
有効	`false`

レートベースの攻撃は、ネットワークまたはホストに過剰なトラフィックを送信することで低速化または正規の要求の拒否を引き起こし、ネットワークまたはホストを混乱させようとします。レートベースの防御を使用し、そのルールの過剰な一致に応じて侵入のアクションを変更することができます。

`rate_filter` は、指定した間隔内にルールの一致が多すぎる場合にその状態を検出します。インライン展開された管理対象デバイス上でこの機能を使用して、指定された時刻のレートベースの攻撃をブロックしてから、ルール一致がイベントを生成するだけでトラフィックをドロップしないルール状態に戻すことができます。

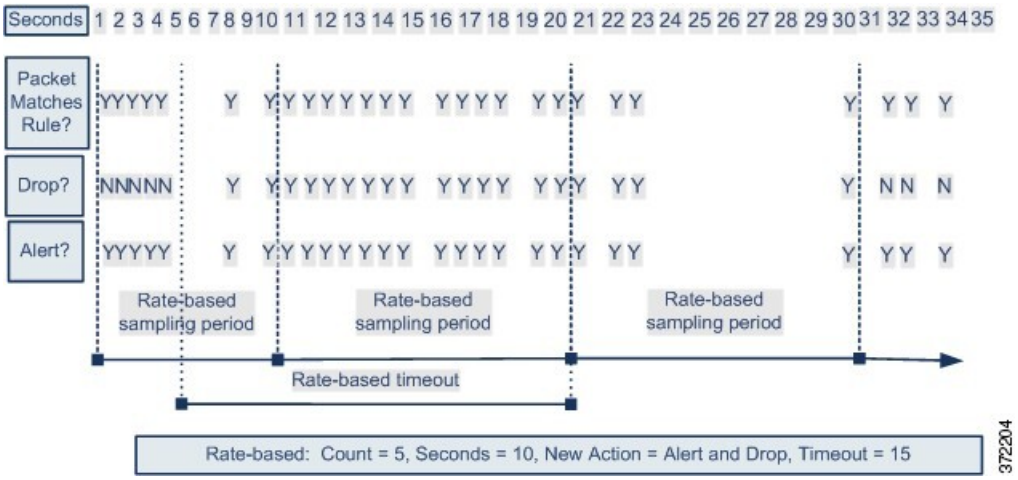
任意の侵入ルールに応じるように `rate_filter` は設定できますが、攻撃を検出して反応するようにするには、指定するルールを `rate_filter` で有効にする必要があります。たとえば、DDOS/SYN フラッド攻撃に対する防御を確立するには、ルール 135:1（TCP SYN を受信）を有効にして、ルール 135:1 の過剰なトリガーについて警告するように `rate_filter` を設定します。

レートベースの攻撃防止は、異常なトラフィック パターンを識別し、正規の要求に対するそのトラフィックの影響を最小限に抑えようとします。特定の宛先 IP アドレスに送信されるトラ

フィックまたは特定の送信元 IP アドレスから送信されるトラフィックの過剰なルール一致を識別できます。また、検出されたすべてのトラフィックを通して特定のルールの過剰な一致に対処することもできます。

次の図は、攻撃者がホストにアクセスしようとしている例を示しています。繰り返しパスワードを特定しようとする試みが、レート ベースの攻撃防御が設定されたルールをトリガーします。レート ベースの設定は、ルール一致が 10 秒間に 5 回発生した時点で、ルール属性を [ドロップしてイベントを生成する（Drop and Generate Events）] に変更します。新しいルール属性は 15 秒後にタイムアウトします。

タイムアウト後も、そのパケットは後続のレートベースのサンプリング期間にドロップされることに注意してください。サンプリング レートが現在または前回のサンプリング期間中にしきい値を超えている場合は、新しいアクションが続行されます。新しいアクションは、サンプリング レートがしきい値レートを下回るサンプリング期間の終了後にのみ、[イベントを生成する（Generate Events）] に戻ります。



同じルールだけでなく異なるルールにも複数のレートベースのフィルタを定義できます。複数のレートベースのフィルタが定義されている侵入ポリシーでは、ポリシーにリストされている最初のフィルタの優先度が最も高くなります。2 つのレートベースのフィルタアクションが競合している場合は、最初のレートベースのフィルタのアクションが実行されます。

`rate_filter`に設定した設定パラメータは、展開全体のすべてのトラフィックに適用されます。ただし、システムはそのシステムがモニタする一意の接続ごとにサンプリング期間内の一致の数に対して個別のカウンタを維持します。また、システムは、接続ごとにアクションに変更を適用します。

（注） レート ベース アクションでは、無効にされたルールを有効にすることも、無効にされたルールに一致するトラフィックをドロップすることもできません。

レートフィルタのパラメータ

rate_filter[]

rate_filter 情報の配列を指定します。各 rate_filter には、トラフィックにレートベースの攻撃が含まれている場合にルールアクションを変更できる一連のフィールドが含まれています。

型：配列（オブジェクト）

例：

```
{
    "rate_filter": {
        "data": [
            {
                "apply_to": "[10.1.2.100, 10.1.2.101]",
                "count": 5,
                "gid": 135,
                "new_action": "alert",
                "seconds": 1,
                "sid": 1,
                "timeout": 5,
                "track": "by_src"
            }
        ],
        "enabled": true,
        "type": "singleton"
    }
}
```

rate_filter[].gid

照合するルールを識別するジェネレータ ID（GID）を指定します。

型：整数

有効な範囲：0 〜 4,294,967,295 (max32)

デフォルト値：1

rate_filter[].sid

照合するルールを識別する署名 ID（SID）を指定します。

型：整数

有効な範囲：0 〜 4,294,967,295 (max32)

デフォルト値：1

rate_filter[].track

送信元アドレスまたは接続先アドレスを照合するフィルタを指定します。

型：列挙体

有効な値は、次のとおりです。

- by_src：rate_filter[].gid と rate_filter[].sid によって指定されたルールに一致し、送信元アドレスが rate_filter[].apply_to と一致するトラフィックのみをフィルタ処理します。
- by_dst：gid と sid によって指定されたルールに一致し、宛先アドレスが rate_filter[].apply_to に一致するトラフィックのみをフィルタ処理します。
- by_rule：rate_filter[].gid と rate_filter[].sid によって指定されたルールに一致するすべてのトラフィックをフィルタ処理します。

デフォルト値：by_src

rate_filter[].count

代替アクション（rate_filter[].new_action）を適用する前に、サンプリング期間（rate_filter[].seconds）で許可するルール一致の数を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：1

rate_filter[].seconds

トラフィックを照合するサンプリング期間の秒数を指定します。rate_filter[].seconds は、一致の内部カウンタをゼロにリセットするまでに経過する時間を表します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：1

rate_filter[].new_action

rate_filter[].seconds と rate_filter[].count で指定された制限を超えるトラフィック内で一致する応答で実行するアクションを指定します。

型：文字列

有効な値：alert、disable、block、drop いずれかの文字列。

デフォルト値：alert

rate_filter[].timeout

一致するトラフィックへの応答の rate_filter[].new_action で指定したアクションを実行する秒数を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

rate_filter[].apply_to

rate_filter[].track の値に応じて、トラフィックの送信元または接続先アドレスと照合するネットワークアドレスのリストを指定します。

型：文字列

有効な値：有効な IPv4 アドレス または CIDR 形式の IPv4 アドレスブロック。

デフォルト値：なし

レートフィルタのルール

rate_filter には、関連付けられたルールがありません。

侵入ルールに応答するように rate_filter を設定できます。ルールの rate_filter を有効にして、攻撃を検出して応答します。

レートフィルタの侵入ルールのオプション

rate_filter には、侵入ルールのオプションがありません。

レートフィルタ操作の確認

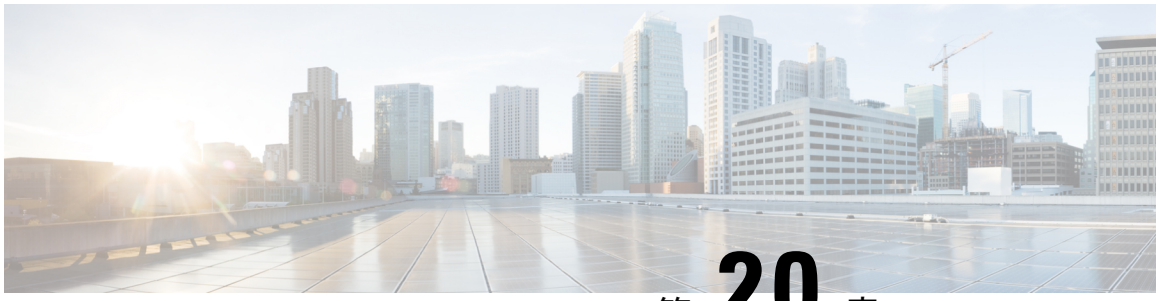
設定されたしきい値を超過した後に実行されるアクションを調べることにより、rate_filter が期待どおりに機能していることを確認できます。

トラフィックが関連付けられた rate_filter を持つ IPS 署名と一致すると、システムは最初にその署名に設定されているアクションを適用します。一致するイベントの数がレートフィルタしきい値に達すると、システムは rate_filter で設定されたアクションを適用します。このしきい値後アクションの発生により、レートフィルタロジックがトリガーされたことが確認されます。

たとえば、IPS 署名アクションが [イベントの生成（Generate Events）] に設定されており、rate_filter アクションが n 回の後に [ブロック（Block）] に設定されている場合、イベントログで [ブロック（Block）] アクションが表示されると、それは n 回のイベントのしきい値に達してレートフィルタ処理が有効になっていることを示します。

送信元または宛先 IP アドレス別にイベントを追跡するように rate_filter が設定されている場合は、指定された IP アドレスとしきい値後アクションに一致するエントリのイベントログを確認します。たとえば、IP アドレス x.x.x.x からのトラフィックをブロックするようにレートフィルタが設定されている場合、しきい値を超過すると、イベントログには x.x.x.x に関連付けられているトラフィックに対する [ブロック（Block）] アクションが表示されます。

IPS 署名によって生成されたイベントとレートフィルタしきい値を超過した後に生成されたイベントを区別できない場合、`rate_filter` は意図した目的を達成していません。正しく設定されたレートフィルタでは、しきい値に達したときに個別のアクションが適用されるはずです。しきい値前イベントとしきい値後イベントがまったく同じに見える場合は、レートフィルタ設定でしきい値とアクションの設定が正しく定義されていることを確認します。



第 20 章

S7CommPlus インスペクタ

- S7CommPlus インスペクタの概要 （185 ページ）
- S7CommPlus インスペクタを設定するためのベストプラクティス （186 ページ）
- S7CommPlus インスペクタのパラメータ （186 ページ）
- S7CommPlus インスペクタのルール （186 ページ）
- S7CommPlus インスペクタの侵入ルールのオプション （187 ページ）

S7CommPlus インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`false`

S7CommPlus は、Siemens が開発した独自のプロトコルです。S7CommPlus は、Siemens S7 ファミリ製品のプログラマブル ロジック コントローラ間の通信を可能にします。

`s7commplus` インスペクタは、S7CommPlus トラフィックを検出して分析します。指定した S7CommPlus 関数と操作コードのヘッダーフィールドで警告を発し、S7CommPlus トラフィックでの攻撃を検出するように侵入ルールのオプションを設定できます。

S7CommPlusインスペクタを設定するためのベストプラクティス

ネットワークに有効になっているS7CommPlusデバイスが含まれていない場合は、トラフィックに適用するネットワーク分析ポリシーの `s7commplus` インスペクタを有効にする必要があります。

S7CommPlus インスペクタのパラメータ

S7CommPlus TCP ポートの設定

`binder` インスペクタは、S7CommPlus TCP ポートの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "role": "server",
            "proto": "tcp",
            "ports": "102"
        },
        "use": {
            "type": "s7commplus"
        }
    },
    {
        "when": {
            "role": "any",
            "service": "s7commplus"
        },
        "use": {
            "type": "s7commplus"
        }
    }
]
```

（注） `s7commplus` インスペクタはパラメータを提供しません。

S7CommPlus インスペクタのルール

`s7commplus` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *23 : S7CommPlus* インスペクタのルール

GID:SID	ルール メッセージ
149:1	S7commplus MBAP ヘッダーの長さが、指定した S7commplus 関数に必要な長さと一致しない（length in S7commplus MBAP header does not match the length needed for the given S7commplus function）
149:2	S7commplus プロトコル ID がゼロ以外になっている（S7commplus protocol ID is non-zero）
149:3	予約済み S7commplus 機能コードは使用中になっている（reserved S7commplus function code in use）

S7CommPlus インスペクタの侵入ルールのオプション

`s7commplus` インスペクタが検出したトラフィックに対する攻撃を識別するカスタム侵入ルールを作成するには、`s7commplus` キーワードを単体で使用するか、または組み合わせて使用します。設定可能なキーワードについては、許容範囲内の既知の単一の値か、または単一の整数を指定します。

次の点に注意してください。

- 同じルール内の複数の `s7commplus` キーワードは、AND 演算されます。
- 同じルールで複数の `s7commplus_func` キーワードまたは `s7commplus_opcode` キーワードを使用すると、ルールが無効になります。無効にされたルールはトラフィックを照合できません。これらのキーワードで複数の値を検索するには、複数のルールを作成します。

s7commplus_content

`s7commplus_content` キーワードを使用して、検出カーソルを S7CommPlus パケットペイロードの先頭に配置します。S7CommPlus 侵入ルールで `content` または `protected_content` キーワードを使用する前に、このキーワードを設定することをお勧めします。

シンタックス：`s7commplus_content;`

例：`s7commplus_content;`

s7commplus_func

`s7commplus_func` キーワードを使用して、指定した S7CommPlus ヘッダーパラメータの 1 つと照合します。S7CommPlus パラメータ名または対応する 16 進コードを指定できます。

型：文字列

Syntax: `s7commplus_func: <header_parameter>;`

有効な値は、次のとおりです。

名前	コード
explore	0x04BB
createobject	0x04CA
deleteobject	0x04D4
setvariable	0x04F2
getlink	0x0524
setmultivar	0x0542
getmultivar	0x054C
beginsequence	0x0556
endsequence	0x0560
invoke	0x056B
getvarsubstr	0x0586
0x0 〜 0xFF	数式では追加の値を使用できることに注意してください。

例：`s7commplus_func: createobject;`

s7commplus_opcode

`s7commplus_opcode` キーワードを使用して、指定された S7CommPlus ヘッダーパラメータの 1 つと照合します。S7CommPlus パラメータ名または対応する 16 進コードを指定できます。

型：文字列

シンタックス：`s7commplus_opcode: <header_parameter>`

有効な値は、次のとおりです。

名前	コード
request	0x31
response	0x32
Notification	0x33
response2	0x02

名前	コード
0x0 〜 0xFF	数式では追加の値を使用できることに注意してください。

例：`s7commplus_opcode: 0x31;`

第 21 章

SIP インスペクタ

- SIP インスペクタの概要 （191 ページ）
- SIP インスペクタのパラメータ （192 ページ）
- SIP インスペクタのルール （195 ページ）
- SIP インスペクタの侵入ルールのオプション （197 ページ）

SIP インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_udp`
有効	`true`

Session Initiation Protocol（SIP）は、1 人以上の参加者を含むリアルタイム呼び出しセッションの作成、変更、および破棄を管理します。SIP が制御できるアプリケーションには、インターネット電話、マルチメディア会議、インスタントメッセージ、オンラインゲーム、ファイル転送などがあります。SIP プロトコルは、テキストベースの要求および応答プロトコルです。

各 SIP 要求には、要求の目的を識別する `method` フィールドと、要求の送信先を指定する `Request-URI` が含まれています。各 SIP 応答のステータス コードは、要求されたアクションの結果を示します。SIP プロトコルは、TCP（ポート 5060）または UDP（ポート 5061）を使用します。

SIP がコールセッションを作成すると、SIP は Real-time Transport Protocol（RTP）を介してオーディオストリームとビデオストリームを送信できます。SIP メッセージの本文には、データチャネルパラメータのネゴシエーション、セッション通知、およびセッションの招待がSession Description Protocol（SDP）の形式で埋め込まれます。

sip インスペクタは、ネットワークトラフィック内の SIP メッセージを検出して分析します。sip インスペクタは、SIP ヘッダーとメッセージ本文を抽出し、SIP メッセージ本文のデータを検出エンジンに渡します。

sip インスペクタは、SIP トラフィックの異常や障害や無効などのコールシーケンスなど既知の脆弱性を検出します。

（注）
- sip インスペクタは RTP メッセージを復号化しません。sip インスペクタは、SDP データで定義されているポートに基づいて RTP チャネルを識別します。
- UDP は通常、SIP でサポートされるメディア セッションを伝送します。sip インスペクタは、復号化された UDP ストリームからセッションの追跡情報を取得します。
- SIP ルールのオプションを使用すると、検索カーソルを SIP パケットヘッダーやメッセージ本文に配置し、パケットの検出を特定の SIP メソッドまたはステータスコードに限定することができます。

SIP インスペクタのパラメータ

SIP サービスの設定

binder インスペクタは、SIP サービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "role": "any",
            "service": "sip"
        },
        "use": {
            "type": "sip"
        }
    }
]
```

ignore_call_channel

オーディオ/ビデオ データ チャネル トラフィックを検査するかどうかを指定します。有効にすると、sip インスペクタはデータ以外のすべての SIP チャネルトラフィックを復号化し、オーディオ/ビデオ SIP データチャネルのトラフィックを無視します。

型：ブール値

有効な値：true、false

デフォルト値：false

max_call_id_len

Call-ID ヘッダーフィールドで許可されるバイトの最大数を指定します。Call-ID フィールドでは、要求と応答の SIP セッションが一位に識別されます。max_call_id_len が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:5 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。sip インスペクタは、Call-ID ヘッダーの長さが max_call_id_len の値より大きい場合にイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：256

max_contact_len

Contact ヘッダーフィールドで許可されるバイトの最大数を指定します。Contact フィールドには、後続のメッセージの連絡先を指定する URI が示されます。値が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:15 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。sip インスペクタは、Contact ヘッダーフィールドの長さが max_contact_len の値より大きい場合にイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：256

max_content_len

メッセージ本文のコンテンツで許可されるバイトの最大数を指定します。値が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:16 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。コンテンツの長さが max_content_len の値より大きい場合に、sip インスペクタはイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：1024

max_dialogs

ストリームセッション内で許容されるダイアログの最大数を指定します。ダイアログの数が設定した制限以上の場合、ダイアログの数が指定した最大数を超えない数まで、sip インスペクタは最も古いダイアログをドロップします。

ルール 140:27 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。

型：整数

有効な範囲：1 ～ 4,294,967,295 (max32)

デフォルト値：4

max_from_len

Fromヘッダーフィールドで許可されるバイトの最大数を指定します。Fromフィールドは、メッセージの発信者を識別します。値が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:9 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。sipインスペクタは、Fromフィールド長さがmax_from_lenの値よりも大きい場合にイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：256

max_request_name_len

要求名で許可されるバイトの最大数を指定します。SIP 要求名は、SIP CSeq トランザクション識別子に指定したメソッドの名前を参照します。値が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:7 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。sip インスペクタは、要求名の長さが max_request_name_len の値よりも大きい場合にイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：20

max_requestName_len

max_requestName_len パラメータは推奨されていません。代わりに max_request_name_len パラメータを使用します。

max_to_len

Toヘッダーフィールドで許可されるバイトの最大数を指定します。Toフィールドは、メッセージの受信側を識別します。値が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:11 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。sipインスペクタは、Toフィールドの長さがmax_to_lenの値より大きい場合にイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：256

max_uri_len

Request-URI で許可されるバイトの最大数を指定します。Request-URI は、要求したリソースへの接続先パスを示します。値が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:3 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。sip インスペクタは、Request-URI フィールドの長さが max_uri_len の値より大きい場合にイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：256

max_via_len

Via ヘッダーフィールドで許可されるバイトの最大数を指定します。Via フィールドは、要求で使用するトランスポートと受信者の場所を識別します。値が 0 の場合、sip インスペクタはアラートを生成しません。

ルール 140:13 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。sip インスペクタは、Via フィールドの長さが max_via_len の値より大きい場合にイベントを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：1024

方法

検出する SIP メソッドのリストを指定します。メソッド名では大文字と小文字が区別されません。リスト内のメソッド名を区切るには、コンマまたはスペースを使用します。メソッド名には英字、数字、下線文字のみが使用できます。

型：文字列

有効な値：ack、benotify、bye、cancel、do、info、invite、join、message、notify、options、prack、publish、quath、refer、register、service、sprack、subscribe、unsubscribe、update

デフォルト値： invite cancel ack bye register options

SIP インスペクタのルール

sip インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *24 : SIP* インスペクタのルール

GID:SID	ルール メッセージ
140:2	要求 URI が空になっている（empty request URI）
140:3	URI が長すぎる（URI is too long）
140:4	コール ID が空になっている（empty call-Id）
140:5	コール ID が長すぎる（Call-Id is too long）
140:6	CSeq 番号が大きすぎるかまたは負になっている（CSeq number is too large or negative）
140:7	CSeq 内の要求名が長すぎる（request name in CSeq is too long）
140:8	From ヘッダーが空になっている（empty From header）
140:9	From ヘッダーが長すぎる（From header is too long）
140:10	To ヘッダーが空になっている（empty To header）
140:11	To ヘッダーが長すぎる（To header is too long）
140:12	Via ヘッダーが空になっている（empty Via header）
140:13	Via ヘッダーが長すぎる（Via header is too long）
140:14	連絡先が空になっている（empty Contact）
140:15	連絡が長すぎる（contact is too long）
140:16	コンテンツの長さが大きすぎるか負になっている（content length is too large or negative）
140:17	パケット内に複数の SIP メッセージがある（multiple SIP messages in a packet）
140:18	コンテンツ長が一致しない（content length mismatch）
140:19	要求名が無効（request name is invalid）
140:20	反射攻撃の招待（Invite replay attack）
140:21	セッション情報の不正な変更（illegal session information modification）
140:22	応答ステータスコードが 3 桁の数字ではない（response status code is not a 3 digit number）
140:23	Content-type ヘッダーが空になっている（empty Content-type header）

GID:SID	ルール メッセージ
140:24	SIP バージョンが無効（SIP version is invalid）
140:25	要求の METHOD と CSEQ ヘッダーが一致しない（mismatch in METHOD of request and the CSEQ header）
140:26	メソッドが不明（method is unknown）
140:27	セッション内のダイアログの最大数に到達した（maximum dialogs within a session reached）

SIP インスペクタの侵入ルールのオプション

sip_method

SIP 要求メソッドは要求の目的を識別します。`sip_method` キーワードを使用して、SIP 要求のメソッドを照合します。メソッド名では大文字と小文字が区別されません。複数のメソッド名はコンマで区切ります。

型：文字列

シンタックス：`sip_method: <methods>;`

有効な値：`ack`、`benotify`、`bye`、`cancel`、`do`、`info`、`invite`、`join`、`message`、`notify`、`options`、`prack`、`publish`、`quath`、`refer`、`register`、`service`、`sprack`、`subscribe`、`unsubscribe`、`update`

例： `sip_method: "ack,service,info,bye";`

sip_stat_code

SIP応答には、3桁のステータスコードが含まれます。SIPステータスコードは、要求したアクションの結果を示します。`sip_stat_code` キーワードを使用して、SIP応答を指定したステータスコードと照合します。

3 桁のステータスコードの最初の桁を表す 1 桁の数字、3 桁の数字、またはいずれかの数字の組み合わせを使用した数字のコンマ区切りのリストを指定できます。リスト内のいずれか1つの番号が SIP 応答内のコードに一致する場合、そのリストが一致します。

型：整数

シンタックス：`sip_stat_code: <codes>;`

有効な範囲：

- `1 〜 9`
- `100 〜 999`

例： `sip_stat_code: "1";`

表 *25 : SIP* パラメータ値とステータスコード

パラメータ値	検出されたステータスコード	説明
189	189	特定のステータスコードを設定します。
1	100 ～ 199	1 桁の数字を設定します。
222, 3	222; 300 ～ 399	3 桁または 1 桁の数字のコンマ区切りのリストを設定します。

sip_header

`sip_header` キーワードを使用して、抽出された SIP ヘッダーバッファの先頭に検出カーソルを配置します。検査をヘッダーフィールドに制限します。

シンタックス：`sip_header;`

例：`sip_header;`

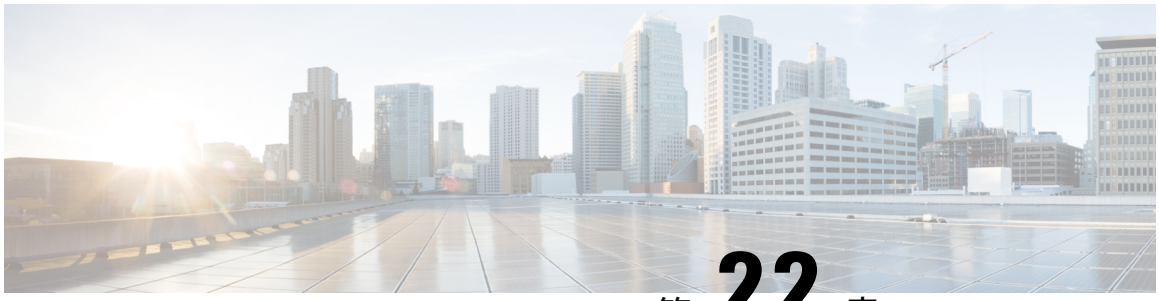
sip_body

`sip_body` キーワードを使用して、抽出された SIP メッセージの本文の先頭に検出カーソルを配置します。検査をメッセージの本文に制限します。

シンタックス：`sip_body;`

例：`sip_body;`

（注） `sip` インスペクタはメッセージの本文全体を抽出して、ルールエンジンで使用できるようにします。ルールエンジンは、Session Description Protocol（SDP）コンテンツの検索に限定されません。



第 22 章

SMTP インスペクタ

- SMTP インスペクタの概要 （199 ページ）
- SMTP インスペクタを設定するためのベストプラクティス （200 ページ）
- SMTP インスペクタのパラメータ （200 ページ）
- SMTP インスペクタのルール （209 ページ）
- SMTP インスペクタの侵入ルールのオプション （210 ページ）

SMTP インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`true`

Simple Mail Transfer Protcocol（SMTP）を使用すると、メールクライアントはメールサーバーにメッセージを送信できます。SMTP は、メッセージを受信者に配信するコマンドを発行します。SMTP サーバーは、安全でないセッションには TCP ポート 25 を使用し、SSL/TLS を介した SMTP には TCP ポート 587 を使用します。

`smtp` インスペクタは、SMTP トラフィックを検出し、SMTP コマンドと応答を分析します。

`smtp`インスペクタは、SMTP メッセージのコマンド、ヘッダー、および本文のセクションを識別し、Multi-purpose Internet Mail Extension（MIME）添付ファイルを抽出して復号化します。MIME 添付ファイルには、複数の添付ファイルや複数のパケットにまたがる大きな添付ファイルが含まれる場合があります。

`smtp` インスペクタは、SMTP メッセージを識別し、Snort の許可リストに追加します。有効にすると、侵入ルールは異常な SMTP トラフィックに関するイベントを生成します。

`smtp` インスペクタを次のように設定できます。

- セッションで生成されたすべてのイベントとともに、送信者の電子メール ID、受信者の電子メール ID、電子メールのヘッダー、および添付ファイルのファイル名をログに記録する。
- 余分な余白文字を削除して、SMTP コマンド ラインを正規化する。smtp インスペクタでスペース（ASCII 0x20）またはタブ（ASCII 0x09）文字を正規化する。
- TLS で暗号化されたトラフィックを無視してパフォーマンスを向上させる。
- プレーンテキストの電子メールデータを無視してパフォーマンスを向上させる。

SMTP インスペクタを設定するためのベストプラクティス

RFC 2821 のガイドラインに従って、smtp インスペクタのコア設定パラメータを設定することをお勧めします。

- max_command_line_len：512 文字
- max_header_line_len：1024 文字
- max_response_line_len：512 文字

SMTP インスペクタのパラメータ

SMTP サービスの設定

binder インスペクタは、SMTP サービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "service": "smtp",
            "role": any
        },
        "use": {
            "type": "smtp"
        }
    }
]
```

alt_max_command_line_len[]

SMTP コマンドの配列と、コマンドの代替の最大行長を指定します。代替最大行長は、SMTP コマンドの max_command_line_len の値をオーバーライドします。このパラメータのイベントを生成するには、ルール 124:4 を有効にします。

型：配列

例：

```
{
    "alt_max_command_line_len": [
        {
            "command": "AUTH",
            "length": 240
        }
    ]
}
```

alt_max_command_line_len[].command

コマンド ストリングを指定します。

型：文字列

有効な値：SMTP コマンド

デフォルト値：「SMTP コマンドとデフォルトの代替コマンド長」を参照してください。

alt_max_command_line_len[].length

代替の最大コマンドライン長を指定します。コマンドのコマンドライン長の検出を無効にするには、0 を指定します。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：「SMTP コマンドとデフォルトの代替コマンド長」を参照してください。

表 *26 : SMTP* コマンドとデフォルトの代替コマンド長

コマンド	長さ
ATRN	255
AUTH	246
BDAT	255
DATA	246
DEBUG	255
EHLO	500
EMAL	255
ESAM	255
ESND	255
ESOM	255

コマンド	長さ
ETRN	500
EVFY	255
EXPN	255
HELO	500
HELP	500
IDENT	255
MAIL	260
NOOP	255
ONEX	246
QUEU	246
QUIT	246
RCPT	300
RSET	255
SAML	246
SEND	246
SIZE	255
SOML	246
STARTTLS	246
TICK	246
TIME	246
TURN	246
TURNME	246
VERB	246
VRFY	255
XADR	246
XAUTH	246
XCIR	246
XEXCH50	246

コマンド	長さ
X-EXPS	246
XGEN	246
XLICENSE	246
X-LINK2STATE	246
XQUE	246
XSTA	246
XTRN	246
XUSR	246

auth_cmds

認証交換を開始するSMTPコマンドのリストを指定します。複数のSMTPコマンドはスペースで区切ります。

型：文字列

有効な値：SMTP 認証交換開始コマンド

デフォルト値：`AUTH XAUTH X-EXPS`

b64_decode_depth

各 Base64 エンコード MIME 電子メール添付ファイルから抽出してデコードできる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 124:10 を有効にして、このパラメーターのイベントを生成し、インライン展開で、デコードが失敗したときに問題のあるパケットをドロップすることができます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

binary_data_cmds

データの送信を開始し、コマンドの後に長さの値（オクテット単位）を使用して送信するデータの量を示すSMTPコマンドのリストを指定します。複数のSMTPコマンドはスペースで区切ります。

型：文字列

有効な値：データ長引数を使用する有効な SMTP データ送信開始コマンド

デフォルト値：`BDATA XEXCH50`

bitenc_decode_depth

バイトの最大数を指定し、エンコードされていない各MIME電子メールの添付ファイルから抽出します。65535 未満の整数を指定するか、または 0 を指定して、エンコードされていない MIME添付ファイルの抽出を無効にすることができます。抽出するバイト数に制限を設定しない場合は、-1 を指定します。これらの添付ファイルのタイプには、7 ビット、8 ビット、バイナリ、ならびにプレーンテキスト、JPEG イメージと PNG イメージ、および MP4 ファイルなど、マルチパートのコンテンツタイプが含まれます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

data_cmds

データの送信を開始し、データの末尾のデリミタ（<CRLF>.<CRLF>）を使用する SMTP コマンドのリストを指定します。

型：文字列

有効な値：データの末尾のデリミタを使用する SMTP データ送信開始コマンド。

デフォルト値：`DATA`

decompress_pdf

MIME 添付ファイルの `application/pdf` （PDF） ファイルを圧縮解除するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_swf

MIME 添付ファイルの `application/vnd.adobe.flash-movie` （SWF）の圧縮を解除するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_vba

MIME 添付ファイルの Microsoft Office Visual Basic for Applications のマクロファイルの圧縮を解除するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

decompress_zip

MIME 添付ファイルのアプリケーション/zip（ZIP）ファイルを解凍するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

email_hdrs_log_depth

SMTP データから抽出する電子メールヘッダーのバイト数を指定します。電子メールヘッダーの抽出を無効にするには、0 を指定します。

型：整数

有効な範囲：`0` 〜 `20480`

デフォルト値：`1464`

ignore_data

電子メールデータセクションを復号化するかどうかを指定します（MIME 電子メールヘッダーを除く）。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

ignore_tls_data

TLS で暗号化されたデータを復号化するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

log_email_hdrs

SMTP 電子メールヘッダーとセッションで生成されたすべてのイベントを復号化してログに記録するかどうかを指定します。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

log_filename

MIME 本文内の `Content-Disposition` ヘッダーから抽出された MIME 添付ファイル名とセッションで生成されたすべてのイベントを復号化してログに記録するかどうかを指定します。メッセージに複数の MIME 添付ファイルが含まれている場合、SMTP インスペクタはファイル名をコンマで区切ってログに記録します。SMTP インスペクタでログに記録できるのは、1024 バイトまでです。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

log_mailfrom

SMTP `MAIL FROM` コマンドから抽出された送信者の電子メールアドレスとセッションで生成されたすべてのイベントを復号化してログに記録するかどうかを指定します。メッセージに複数の送信者が含まれている場合、SMTP インスペクタは送信者をコンマで区切ってログに記録します。SMTP インスペクタでログに記録できるのは、1024 バイトまでです。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

log_rcptto

SMTP `RCPT TO` コマンドからの受信者の電子メールアドレスとセッションで生成されたすべてのイベントを復号化してログに記録するかどうかを指定します。メッセージに複数の受信者が含まれている場合、SMTP インスペクタは受信者をコンマで区切ってログに記録します。SMTP インスペクタでログに記録できるのは、1024 バイトまでです。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

max_auth_command_line_len

SMTP 認証コマンドラインで受け入れられる最大バイト数を指定します。

ルール 124:15 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。SMTP `AUTH` コマンドのアラートを無効にするには 0 を指定するか、または Snort 設定の `max_auth_command_line_len` パラメータを省略します。

型：整数

有効な範囲：`0` 〜 `65535`

デフォルト値：`1000`

max_command_line_len

SMTP コマンドラインで受け入れられる最大バイト数を指定します。

RFC 2821 の SMTP に関するネットワーク作業部会は、最大コマンドライン長は 512 バイトをお勧めしています。SMTP コマンドライン長でアラートを無効にするには 0 を指定するか、または Snort 設定の `max_command_line_len` パラメータを省略します。

ルール 124:1 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。

型：整数

有効な範囲：`0` 〜 `65535`

デフォルト値：`512`

max_header_line_len

SMTP データヘッダー行に許可される最大バイト数を指定します。

RFC 2821 の SMTP に関するネットワーク作業部会は、最大データヘッダー行の長さは 1024 バイトをお勧めしています。SMTP データヘッダー長でアラートを無効にするには 0 を指定するか、または Snort 設定の `max_header_line_len` パラメータを省略します。

ルール 124:2 と 124:7 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。

型：整数

有効な範囲：`0` 〜 `65535`

デフォルト値：`1000`

max_response_line_len

SMTP 応答行で受け入れられる最大バイト数を指定します。

RFC 2821 の SMTP に関するネットワーク作業部会は、最大応答行長は 512 バイトをお勧めしています。SMTP 応答行の長さでアラートを無効にするには 0 を指定するか、または Snort 設定の `max_response_line_len` パラメータを省略します。

ルール 124:3 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。

型：整数

有効な範囲：`0` 〜 `65535`

デフォルト値：`512`

normalize

すべてのコマンド、コマンドなし、またはコマンドのリストを正規化するかどうかを指定します。コマンドのリストは、`normalize_cmds` パラメータで指定できます。インスペクタは、コマンドの後に複数のスペース（ASCII 0x20）またはタブ（ASCII 0x09）文字を確認します。

型：列挙体

有効な値は、次のとおりです。

- `none`
- `cmds`
- `all`

デフォルト値：`none`

normalize_cmds

正規化する SMTP コマンドのリストを指定します。複数の SMTP コマンドはスペースで区切ります。

型：文字列

有効な値：SMTP コマンド

デフォルト値：なし

qp_decode_depth

各 quoted-printable（QP）エンコード MIME 電子メール添付ファイルから抽出してデコードできる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 124:11 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

uu_decode_depth

各 Unix-to-Unix エンコード（UU エンコード）MIME 電子メール添付ファイルから抽出して復号化できる最大バイト数を指定します。65535 未満の整数を指定するか、または 0 を指定して復号化を無効にすることができます。復号化するバイト数に制限を設定しない場合は、-1 を指定します。

ルール 124:13を有効にしてこのパラメータのイベントを生成し、インライン展開で、（エンコードが正しくないか、またはデータが壊れていることが原因など）復号化が失敗した場合に問題のあるパケットをドロップすることができます。

型：整数

有効な範囲：`-1` 〜 `65535`

デフォルト値：`-1`

valid_cmds

SMTP インスペクタが有効と判断する SMTP コマンドの追加リストを指定します。

SMTP インスペクタは、デフォルトの有効な SMTP コマンド（`ATRN AUTH BDAT DATA DEBUG EHLO EMAL ESAM ESND ESOM ETRN EVFY EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET SAML SEND SIZE STARTTLS SOML TICK TIME TURN TURNME VERB VRFY X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR`）のリストを定義します。

ルール 124:5 を有効にしてイベントを生成し、インライン展開で問題のあるパケットをドロップします。

型：文字列

有効な値：SMTP コマンド

デフォルト値：なし

xlink2state

SMTP インスペクタが X-Link2State Microsoft Exchange バッファ データ オーバーフロー攻撃の一部であるパケットを処理する方法を指定します（脆弱性の説明については、CVE-2005-0560 を参照してください）。検出を無効（`disable`）にし、検出を有効にしてアラートを生成する（`alert`）、または検出を有効にして問題のあるパケットをドロップする（`drop`）ことができます。

このパラメータのイベントを生成し、インライン展開で問題のあるパケットをドロップするには、ルール 124:8 を有効にします。

型：列挙体

有効な値は、次のとおりです。

- `disable`
- `alert`
- `drop`

デフォルト値：`alert`

SMTP インスペクタのルール

`smtp`インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *27 : SMTP* インスペクタのルール

GID:SID	ルール メッセージ
124:1	コマンドバッファのオーバーフローを試行した（attempted command buffer overflow）

GID:SID	ルール メッセージ
124:2	データヘッダーバッファのオーバーフローを試行した（attempted data header buffer overflow）
124:3	応答バッファのオーバーフローを試行した（attempted response buffer overflow）
124:4	特定のコマンドバッファのオーバーフローを試行した（attempted specific command buffer overflow）
124:5	不明なコマンド（unknown command）
124:6	不正なコマンド（illegal command）
124:7	ヘッダー名バッファのオーバーフローを試行した（attempted header name buffer overflow）
124:8	X-Link2State コマンドバッファのオーバーフローを試行した（attempted X-Link2State command buffer overflow）
124:10	Base64 の復号化に失敗した（base64 decoding failed）
124:11	Quoted-Printable の復号化に失敗した（quoted-printable decoding failed）
124:13	Unix-to-Unix の復号化に失敗した（Unix-to-Unix decoding failed）
124:14	Cyrus SASL 認証攻撃（Cyrus SASL authentication attack）
124:15	認証コマンドバッファのオーバーフローを試行した（attempted authentication command buffer overflow）
124:16	ファイルの圧縮解除に失敗した（file decompression failed）

SMTP インスペクタの侵入ルールのオプション

vba_data

検出カーソルを Microsoft Office Visual Basic for Applications マクロバッファに設定します。

シンタックス：`vba_data;`

例：`vba_data;`

第 23 章

SnortML

[名前（Name）]	Snort_ml
タイプ	インスペクタ（パッシブ）
使用方法	検査
インスタンス タイプ	単一
その他のインスペクタが必要	`snort_ml_engine`、`http_inspect`
有効	[最大検出（Maximum Detection）] ポリシー

最新の環境の機能に不可欠なソフトウェアでは、毎日新たな脆弱性が検出されています。セキュリティアナリストは、これらの新たな脆弱性を分析してそれらをトリガーするのに必要なものを分離し、それらをターゲットとするエクスプロイトを検出するための署名を作成します。ほとんどの署名は、特定の脆弱性に対してのみ実際に作成できます。

SnortML は、Snort 侵入防御システム用のニューラル ネットワークベースのエクスプロイト検出です。トレーニングデータから既知の攻撃を検出するだけでなく、これまでに確認されたことのない攻撃を検出するように設計されています。

`snort_ml` インスペクタは、主に HTTP を介した SQL インジェクション攻撃を検索します。このインスペクタはパフォーマンスに影響を与える可能性があるため、[最大検出（Maximum Detection）] ポリシーでのみデフォルトでのみ有効になっています。

Cisco Secure Firewall バージョン 10.0.0 以降、HTTP トラフィックを介したコマンド ライン インジェクション攻撃の検出がサポートされています。コマンド ライン インジェクション攻撃は、攻撃者がホストOS上でオペレーティングシステム（OS）コマンドを入力できる場合に発生します。

また、Cisco Secure Firewall バージョン 10.0.0 以降、サイズが 256、512、または 1,024 バイトのニューラル ネットワーク モデルが URI 入力に基づいて自動的に選択されます。トレーニングされたニューラル ネットワーク モデルを小さくすると、有効性に影響を与えることなく、大規模なモデルに比べて短時間で分析を行えます。

- SnortML ルール （212 ページ）
- SnortML パラメータ （212 ページ）

• SnortML ルールの有効化 （212 ページ）
• SnortML インスペクタの無効化 （213 ページ）
• 統合イベントでの SnortML ルールイベントの表示 （214 ページ）
• 侵入イベントでの SnortML ルールイベントの表示 （214 ページ）

SnortML ルール

snort_ml インスペクタルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。snort_ml インスペクタルールは、[最大検出（Maximum Detection）] 侵入およびネットワーク アクセス ポリシーでのみデフォルトで有効になります。

表 *28 : Snort ML* インスペクタルール

GID:SID	ルール メッセージ
411:1	（snort_ml）ニューラル ネットワーク ベースのエクスプロイト検出を介して HTTP パラメータで潜在的な脅威を検出。

SnortML パラメータ

uri_depth

HTTP URI からスキャンするバイト数を指定します。値 -1 は無制限を意味します。

型：整数

有効な範囲：-1 ～ 2147483648

デフォルト値：-1

client_body_depth

HTTP クライアントボディからスキャンするバイト数を指定します。値 -1 は無制限を意味します。

型：整数

有効な範囲：-1 ～ 2147483648

デフォルト値：0

SnortML ルールの有効化

設定した侵入ポリシーのベースポリシーとして [最大検出（Maximum Detection）] を使用している場合、SnortML ルールはデフォルトで有効になります。ベースポリシーとして [最大検出

（Maximum Detection）] を使用していない場合、SnortML ルールはデフォルトで無効になります。SnortML ルールを有効にするには、次の手順を実行します。

手順

ステップ 1 [ポリシー（Policies）] > [侵入（Intrusion）] の順に選択します。.

ステップ 2 [侵入ポリシー（Intrusion Policies）] タブで、SnortML ルールを有効にする侵入ポリシーの [Snort 3バージョン（Snort 3 Version）] をクリックします。 [侵入ポリシーについて（About Intrusion Policies）] ウィンドウが表示されます。

ステップ 3 [侵入ポリシーについて（About Intrusion Policies）] ウィンドウに表示される情報を確認し、[取り消す（Dismiss）] をクリックします。

ステップ 4 [概要（Summary）] タブで、[有効なポリシーを表示（View Effective Policy）] をクリックします。

ステップ 5 [有効なポリシー（Effective Policy）] ウィンドウのフィルタバーに **GID=411** と入力し、**Enter** キーを押します。**411:1** ルールが表示されます。

ステップ 6 [ルールアクション（Rule Action）] ドロップダウンリストから、[ブロック（Block）] を選択します。

ステップ 7 表示されるウィンドウで、[オーバーライド（Override）] をクリックします。

ステップ 8 [概要（Summary）] をクリックして [概要（Summary）] ウィンドウに戻り、[オーバーライドされたルール（Overridden Rules）] の数が 1 つ増えているかどうかを確認します。これで SnortML ルールが有効になり、SnortML インスペクタも自動的に有効になります。

SnortML インスペクタの無効化

snort_ml インスペクタを無効にするには、次の手順を実行します。

手順

ステップ 1 [ポリシー（Policies）] > [侵入（Intrusion）] の順に選択します。

ステップ 2 [ネットワーク分析ポリシー（Network Analysis Policies）] タブで、snort_ml インスペクタを無効にするネットワーク分析ポリシーの [Snort 3バージョン（Snort 3 Version）] をクリックします。表示される警告を確認し、[はい（Yes）] をクリックします。

ステップ 3 [インスペクタ（Inspectors）] のリストにある [snort_ml] をクリックします。

ステップ 4 [オーバーライドされた設定（Overridden Configuration）] の横にある [編集（Edit）]（ ） アイコンをクリックします。

ステップ 5 [設定のオーバーライド（Override Configuration）] ウィンドウで、[「有効」：true（"enabled": true）] を [「有効」：false（"enabled": false）] に変更します。

ステップ 6 テキストボックス以外をクリックすると、JSON 形式を確認できます。[JSONシンタックスOK（JSON syntax ok）] と表示されたら、[OK] をクリックして設定を保存します。加えた変更は、[オーバーライドされた設定（Overridden Configuration）] で確認できます。

ステップ 7 [保存（Save）] をクリックし、ネットワーク分析ポリシーに加えた変更を保存します。

次のタスク

アクセス コントロール ポリシーの必要なアクセス制御ルールで、変更された侵入ポリシーを適用していることを確認します。また、アクセス コントロール ポリシーの [詳細設定（Advanced Settings）] にある [ネットワーク分析および侵入ポリシー（Network Analysis and Intrusion Policies）] セクションから、変更したネットワーク アクセス ポリシーを適用します。

統合イベントでの SnortML ルールイベントの表示

手順

ステップ 1 [イベントとログ（Events & Logs）] > [分析（Analysis）] > [統合イベント（Unified Events）] の順に選択します。.

ステップ 2 検索バーに侵入（intrusion）と入力し、[侵入イベント（Intrusion Events）] を選択します。

ステップ 3 検索バーの [追加（Add）] フィールドに *snort_ml* と入力し、[適用（Apply）] をクリックします。SnortML ルールが原因で生成された侵入イベントが表示されます。

ステップ 4 （任意） 生成されたイベントの詳細を表示するには、[右矢印（right arrow）]（ > ） をクリックします。

侵入イベントでの SnortML ルールイベントの表示

手順

ステップ 1 [イベントとログ（Events & Logs）] > [侵入（Intrusions）] > [イベント（Events）] の順に選択します。.

ステップ 2 侵入イベントの詳細ビューについては、[イベントのテーブルビュー（Table View of Events）] をクリックします。

ステップ 3 [検索の編集（Edit Search）] をクリックします。

ステップ 4 [メッセージ（Message）] フィールドに *snort_ml* と入力し、[検索（Search）] をクリックします。SnortML ルールによって生成されたすべての侵入イベントが表示されます。

第 24 章

SSH インスペクタ

- SSH インスペクタの概要 （215 ページ）
- SSH インスペクタを設定するためのベストプラクティス （216 ページ）
- SSH インスペクタのパラメータ （216 ページ）
- SSH インスペクタのルール （218 ページ）
- SSH インスペクタの侵入ルールのオプション （218 ページ）

SSH インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	なし
有効	`true`

Secure Shell Protocol （SSH）は、セキュリティで保護されていないネットワークを介したクライアントとサーバー間の安全な通信を可能にするネットワークプロトコルです。SSH はトンネリングをサポートし、公開鍵暗号化を使用してリモートホストを認証します。

SSH を使用してファイルを安全に転送したり、リモートホストにログインしてコマンドラインと連携動作したりできます。SSH プロトコルは、TCP、UDP、または SCTP を介してポート 22 を使用します。

`ssh` インスペクタは、ストリームパケットを復号化し、次の SSH エクスプロイトを検出します。

- チャレンジ/応答バッファ オーバーフロー エクスプロイト
- CRC-32 エクスプロイト
- SecureCRT SSH クライアント バッファ オーバーフロー エクスプロイト

- 正しくない SSH メッセージの方向

ホスト間のネットワーク接続が暗号化されている場合、認証後にチャレンジ/レスポンス バッファ オーバーフローと CRC-32 攻撃が発生します。どちらのタイプの攻撃も、認証チャレンジ直後に 20 KB を超える大きなペイロードをサーバーに送信します。

ssh インスペクタは、サーバーに送信されたバイト数をカウントすることで、チャレンジ/応答バッファ オーバーフローと CRC-32 攻撃を検出します。バイト数が、事前に定義したパケット数内で定義した制限を超えた場合、ssh インスペクタはアラートを生成します。CRC-32 攻撃の対象となるのは SSH バージョン 1 のみであり、チャレンジ/応答バッファ オーバーフロー エクスプロイトの対象となるのは SSH バージョン 2 のみです。ssh インスペクタは、セッションの開始時に SSH バージョン文字列を読み取り、攻撃のタイプを識別します。

SecureCRT SSH クライアント バッファ オーバーフロー攻撃とプロトコル不一致攻撃は、キー交換前にホストが接続をセキュリティで保護しようとするときに発生します。SecureCRT SSH クライアント バッファ オーバーフロー攻撃では、非常に長いプロトコル識別子の文字列がクライアントに送信され、それが原因でバッファオーバーフローが発生します。プロトコル不一致攻撃は、SSH 以外のクライアント アプリケーションがセキュア SSH サーバーに接続しようとするか、またはサーバーとクライアントのバージョン番号が一致しない場合に発生します。

（注） ssh インスペクタはブロートフォース攻撃を処理しません。

SSH インスペクタを設定するためのベストプラクティス

デフォルトの ssh インスペクタ設定を使用することをお勧めします。max_encrypted_packets パラメータで定義されているセッションの暗号化パケットの最大数を超えると、ssh インスペクタはそのセッションのトラフィックの処理を停止してパフォーマンスを向上させます。ssh インスペクタは、SSH セッションの開始時に表示される SSH の脆弱性のみを検出します。

（注） ssh インスペクタがチャレンジ/応答 オーバーフローまたは CRC 32 で誤検知を生成した場合、max_client_bytes パラメータを使用して、必要なクライアントバイト数を増やすことができます。

SSH インスペクタのパラメータ

SSH サービスの設定

binder インスペクタは、SSH サービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "service": "ssh",
            "role": any
        },
        "use": {
            "type": "ssh"
        }
    }
]
```

max_encrypted_packets

ssh インスペクタが SSH セッションを無視するまでに調べる暗号化パケットの最大数を指定します。セッションの暗号化パケットの最大数を超えた場合、パフォーマンスを向上させるために、ssh インスペクタはそのセッションのトラフィックの処理を停止します。

型：整数

有効な範囲：-1 ～ 65535

デフォルト値：25

max_client_bytes

チャレンジ/応答オーバーフローまたは CRC 32 で ssh インスペクタがアラートを生成するまでにサーバーに送信する未応答バイトの最大数を指定します。max_encrypted_packets が送信される前に max_client_bytes の制限を超えた場合、インスペクタは攻撃が発生したと見なし、トラフィックを無視します。

ルール 128:1 を有効にして、インスペクタがチャレンジ/応答オーバーフローを検出したときにアラートを生成するか、ルール 128:2 を有効にして、が CRC 32 エクスプロイトを検出したときにアラートを生成することができます。

クライアントがサーバーから受信する有効な応答ごとに、ssh インスペクタは max_client bytes のパケット数をリセットします。

（注） max_client_bytes を 0 または 1 に設定することはお勧めしません。max_client_bytes を 0 または 1 に設定すると、ssh インスペクタは常にアラートを生成します。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：19600

max_server_version_len

SSH サーバーのバージョン文字列の最大長を指定します。SSH サーバーのバージョン文字列の長さが max_server_version_len を超える場合、ssh インスペクタはアラートを生成します。ルール 128:3 を有効にして、セキュア CRT サーバーのバージョン文字列のオーバーフローのアラートを生成できます。

型：整数

有効な範囲：0 ～ 255

デフォルト値：80

（注） ssh インスペクタのデフォルト設定では、アラートは有効になりません。

SSH インスペクタのルール

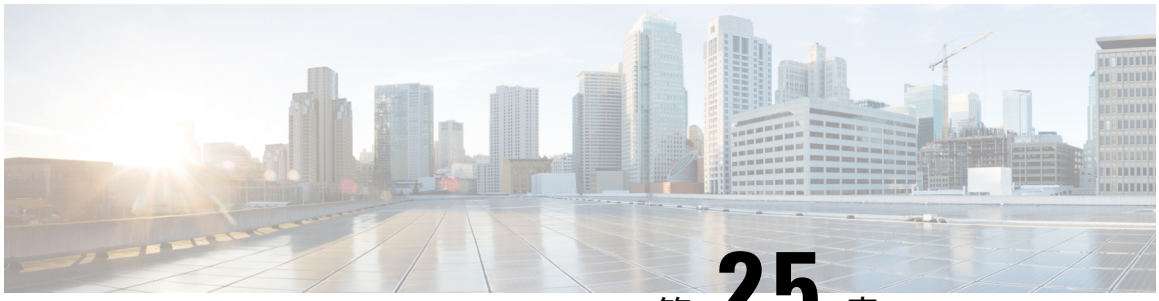
ssh インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。 。

表 *29 : SSH* インスペクタのルール

GID:SID	ルール メッセージ
128:1	challenge-response オーバーフローのエクスプロイト（challenge-response overflow exploit）
128:2	SSH1 CRC32 エクスプロイト（SSH1 CRC32 exploit）
128:3	サーバーバージョン文字列のオーバーフロー（server version string overflow）
128:5	不正なメッセージの方向（bad message direction）
128:6	指定のペイロードに対してペイロードサイズが正しくない（payload size incorrect for the given payload）
128:7	SSH バージョンの文字列の検出に失敗した（failed to detect SSH version string）

SSH インスペクタの侵入ルールのオプション

ssh インスペクタには、侵入ルールのオプションはありません。



第 25 章

ストリーム ICMP インスペクタ

- ストリーム ICMP インスペクタの概要 （219 ページ）
- ストリーム ICMP インスペクタを設定するためのベストプラクティス （220 ページ）
- ストリーム ICMP インスペクタのパラメータ （220 ページ）
- ストリーム ICMP インスペクタのルール （220 ページ）
- ストリーム ICMP インスペクタの侵入ルールのオプション （220 ページ）

ストリーム ICMP インスペクタの概要

タイプ	インスペクタ（ストリーム）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	なし
有効	`true`

Internet Control Message Protocol（ICMP）は、ネットワーク ユーティリティ アプリケーションやネットワークデバイスで使用されるネットワーク層プロトコルです。ICMP は、診断情報とエラー情報を送信して、IP ホスト間の通信が成功したか、または失敗したかを識別します。ICMP メッセージには、ヘッダーセクションとデータセクションが含まれます。

ICMP は、他のフローに関する情報を伝達します。リアセンブルが必要なデータは伝送せず、ターゲットベースのバインドも必要ありません。

`stream_icmp` インスペクタは、ICMP フロートラッキングを定義します。ping の場合、インスペクタは、ICMP ヘッダーの送信元および接続先の IP アドレスフィールドとポートフィールドを介して基本的なフロートラッキングを実行します。到達不能な接続先の場合、インスペクタは元の IP アドレスと転送ポートを分析し、セッションの状態を更新します。`port_scan` インスペクタは、到達不能なホストとポートが使用可能な場合は、それらを使用できます。

ストリーム ICMP インスペクタを設定するためのベストプラクティス

stream_icmp インスペクタを設定する場合は、次のベストプラクティスを考慮してください。

- ホストまたはネットワークに適用するセッションタイムアウトごとに stream_icmp インスペクタを作成します。stream_icmp インスペクタは、session_timeout を binder インスペクタで定義されている ICMP ホストまたはネットワークに関連付けます。

  同じネットワーク分析ポリシー（NAP）に複数のバージョンの stream_icmp インスペクタを含めることができます。

ストリーム ICMP インスペクタのパラメータ

session_timeout

stream_icmp インスペクタが状態テーブルの非アクティブな ICMP ストリームを保持する秒数を指定します。Snort が同じフローキーを持つ ICMP データグラムを次に 検出すると、以前のフローのセッションタイムアウトが期限切れになっているかどうかが確認されます。タイムアウトの期限が切れると、Snort はフローを閉じて新しいフローを開始します。Snort は、基本のストリーム設定に関連付けられた古いフローを確認します。

型：整数

有効な範囲：0 ～ 2,147,483,647 (max31)

デフォルト値：60

ストリーム ICMP インスペクタのルール

stream_icmp インスペクタには、関連付けられたルールはありません。

ストリーム ICMP インスペクタの侵入ルールのオプション

stream_icmp インスペクタには侵入ルールのオプションはありません。

第 26 章

ストリーム IP インスペクタ

- ストリーム IP インスペクタの概要 （221 ページ）
- ストリーム IP インスペクタを設定するためのベストプラクティス （222 ページ）
- ストリーム IP インスペクタのパラメータ （222 ページ）
- ストリーム IP インスペクタのルール （224 ページ）
- ストリーム IP インスペクタの侵入ルールのオプション （225 ページ）

ストリーム IP インスペクタの概要

タイプ	インスペクタ（ストリーム）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	なし
有効	`true`

Internet Protocol（IP）は、インターネットの基礎を形成するコネクションレス型のネットワーク層プロトコルです。IP はホストアドレスを使用して、IP ネットワークを介し、送信元ホストから接続先ホストにメッセージをルーティングします。IPは、他のトランスポートプロトコルに加えて、TCP データパケットと UDP データパケットの両方をルーティングできます。

IP メッセージには、ヘッダーセクションとデータセクションが含まれています。IP ヘッダーには、メッセージを接続先にルーティングするために使用される IP アドレスが含まれています。IPデータセクションでは、メッセージペイロードがカプセル化されます。IPは、メッセージのリアセンブルとフラグメント化を処理します。

`stream_ip` インスペクタは、IP ネットワークフローを検出し、フロー内のパケットを確認します。`stream_ip` インスペクタは、IP セッションとフロートラッキング、オペレーティングシステム ポリシー、およびデータグラムのオーバーラップ設定パラメータを定義します。モードに応じて、`stream_ip` インスペクタまたは Snort データプレーンが最適化を処理します。

ストリームIPインスペクタを設定するためのベストプラクティス

stream_ip インスペクタを設定する場合は、次のベストプラクティスを考慮してください。

- ホスト、エンドポイント、またはネットワークに適用する IP 設定ごとに stream_ip インスペクタを作成します。ストリーム IP インスペクタは、IP 設定を binder インスペクタで定義された IP ホスト、エンドポイント、またはネットワークに関連付けます。

  同じネットワーク分析ポリシーに複数のバージョンの stream_ip インスペクタを含めることができます。

ストリーム IP インスペクタのパラメータ

max_overlaps

データグラムごとの最大許容オーバーラップを指定します。オーバーラップを無制限に許可するには、0 を指定します。

ルール 123:12 を有効にして、フラグメントのオーバーラップが過剰な場合にアラートをトリガーできます。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

min_frag_length

IP フラグメントで予想される最小バイト数を指定します。IP フラグメントのバイト数を無制限に許可するには、0 を指定します。

ルール 123:13 を有効にして、min_frag_length よりも短いフラグメントのアラートをトリガーできます。

型：整数

有効な範囲：0 ～ 65535

デフォルト値：0

min_ttl

ホップの最小数または存続時間（TTL）を指定します。指定した最小TTLを下回るフラグメントを破棄します。

ルール 123:11 を有効にして、TTL でこの値を下回るフラグメントのアラートをトリガーできます。

型：整数

有効な範囲：1 ～ 255

デフォルト値：1

policy

ターゲットホスト（複数可）のオペレーティングシステムを指定します。オペレーティングシステムは、適切な IP フラグメントのリアセンブルポリシーとオペレーティングシステムの特性を決定します。ストリーム IP インスペクタごとに policy パラメータを 1 つだけ定義できます。

（注） policy パラメータを first に設定すると、Snort はある程度の保護を提供できますが、攻撃は見逃します。IP ストリームインスペクタの policy パラメータを編集して、正しいオペレーティングシステムを指定する必要があります。

型：列挙体

有効な値： policy パラメータのオペレーティングシステムのタイプを設定します。

表 *30*：ポリシーの有効な値

ポリシー	オペレーティング システム
first	不明な OS（Unknown OS）
linux	Linux
bsd	AIX FreeBSD OpenBSD
bsd_right	HP JetDirect（プリンタ）
last	Cisco IOS
windows	Windows 98 Windows NT Windows 2000 Windows XP
solaris	Solaris OS SunOS

デフォルト値：`linux`

session_timeout

`stream_ip` インスペクタが状態テーブルの非アクティブな IP ストリームを保持する秒数を指定します。Snort が同じフローキーを持つ IP データグラムを次に検出すると、以前のフローのセッションタイムアウトが期限切れになっているかどうかが確認されます。タイムアウトの期限が切れると、Snort はフローを閉じて新しいフローを開始します。Snort は、基本のストリーム設定に関連付けられた古いフローを確認します。

型：整数

有効な範囲：`0 ～ 2,147,483,647 (max31)`

デフォルト値：`60`

ストリーム IP インスペクタのルール

`stream_ip` インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

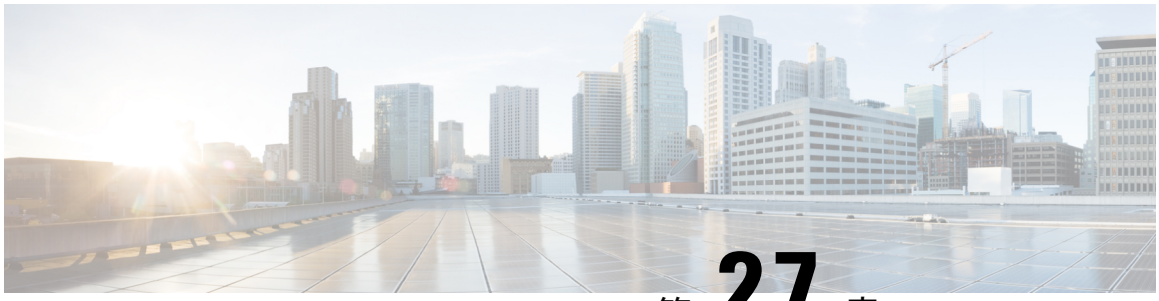
表 *31 :* ストリーム *IP* インスペクタのルール

GID:SID	ルール メッセージ
123:1	フラグメント化されたパケットの IP オプションに一貫性がない（inconsistent IP options on fragmented packets）
123:2	ティアドロップ攻撃（teardrop attack）
123:3	短いフラグメント、DOS 試行の可能性がある（short fragment, possible DOS attempt）
123:4	フラグメントパケットが最適化済みのパケットの後で終了している（fragment packet ends after defragmented packet）
123:5	ゼロバイトのフラグメントパケット（zero-byte fragment packet）
123:6	フラグメントサイズが誤っているかパケットサイズが負になっている（bad fragment size, packet size is negative）
123:7	フラグメントサイズが誤っているか、パケットサイズが 65536 を超えている（bad fragment size, packet size is greater than 65536）
123:8	フラグメント化が重複している（fragmentation overlap）
123:11	TTL 値が設定された最小値よりも小さい、リアセンブルには使用しない（TTL value less than configured minimum, not using for reassembly）
123:12	フラグメントの重複が多すぎる（excessive fragment overlap）

GID:SID	ルール メッセージ
123:13	フラグメントが小さい（tiny fragment）

ストリーム IP インスペクタの侵入ルールのオプション

stream_ip インスペクタには侵入ルールのオプションはありません。



第 27 章

ストリーム TCP インスペクタ

- ストリーム TCP インスペクタの概要 （227 ページ）
- ストリーム TCP インスペクタを設定するためのベストプラクティス （228 ページ）
- TCP ストリームのリアセンブルのためのベストプラクティス （229 ページ）
- ストリーム TCP インスペクタのパラメータ （230 ページ）
- ストリーム TCP インスペクタのルール （235 ページ）
- ストリーム TCP インスペクタの侵入ルールのオプション （237 ページ）

ストリーム TCP インスペクタの概要

タイプ	インスペクタ（ストリーム）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	なし
有効	`true`

Transmission Control Protocol（TCP）は、コネクション型のステートフルなトランスポート層プロトコルです。TCP は、クライアントとサーバー間で IP ネットワークを介して順序付けされたバイトストリームを確実に送信できます。TCP では、接続パラメータ値が同じ接続は、一度に 1 つしか存在できません。ホストのオペレーティングシステムは、TCP 接続の状態を管理します。

`stream_tcp` インスペクタは、TCP フロートラッキング、ストリームの正規化、およびストリームのリアセンブルを実行します。各ストリーム TCP インスペクタが、ネットワーク内の 1 つ以上のホストの TCP トラフィックを処理できます。さらに、TCP トラフィックをネットワークに送信しているホストに関する十分な情報がある場合は、それらのホストに対して `stream_tcp` インスペクタを設定できます。

ネットワーク分析ポリシー（NAP）では、Snort は設定した `stream_tcp` インスペクタそれぞれを、`binder` インスペクタの設定で定義されている TCP サービスに適用します。

複数のストリームTCPインスペクタを設定して、さまざまなオペレーティングシステムとTCPトラフィックを処理できます。

stream_tcp インスペクター構成には、次のものが含まれます。

- TCP ホストのオペレーティング システム
- オペレーティングシステムのオプション：リアセンブル時の重複の処理方法
- トラフィック処理のオプション：セッションまたは方向のバイトまたはセグメントの最大数
- TCP ストリームリアセンブルのオプション：リアセンブルされた PDU の最大サイズ

（注） インライン IPS モードでは、`stream_tcp` インスペクタはペイロードストリームを正規化し、重複が常に最初に見つかったコピーに解決されるようにします。各ストリーム TCP インスペクタは、繰り返される SYN、RST 検証、およびタイムスタンプチェックを処理します。

ストリーム TCP インスペクタを設定するためのベストプラクティス

`stream_tcp` インスペクタをリアセンブルするときは、次のベストプラクティスを考慮してください。

- Snort がフローの片側だけを検査できるように、デバイスにセンシングインターフェイスを展開しないでください。`stream_tcp` インスペクタで `reassemble_async` パラメータを有効にして、非対称トラフィックを処理できます。ただし、ストリームTCPインスペクタは、すべての場合で非対称トラフィックを処理できるわけではありません。たとえば、HTTP HEAD要求への応答により、HTTPインスペクタが同期しなくなる可能性があります。IDS モードでは、TCP 確認応答がないため、回避がはるかに簡単になります。

  IPSモードの場合、Snortがフローの両側を検査できる場合にのみデバイスを展開することをお勧めします。

- TCP トラフィックを送受信する予定の TCP ホストのオペレーティングシステムごとに `stream_tcp` インスペクタを作成します。同じネットワーク分析ポリシーに複数のバージョンの `stream_tcp` インスペクタを含めることができます。各 `stream_tcp` インスペクタで定義された TCP ポリシーは、`binder` インスペクタで指定された TCP ホストに適用されます。
- IPS モードを有効にするには、`normalizer` インスペクタの `normalizer.tcp.ips` パラメータを `true` に設定します。
- ネットワーク分析ポリシー（NAP）の詳細設定では、カスタムターゲットベースの `stream_tcp` インスペクタで識別するネットワークが親 NAP で処理されるネットワーク、

ゾーン、および VLAN のサブネットと一致するか、またはサブネットであることを確認します。

- システムは、各リーフドメインに個別のネットワークマップを作成します。マルチドメイン展開では、実際の IP アドレスを使用してこの設定を抑制すると、予期しない結果になる可能性があります。 上書き対応オブジェクトを使用すると、子孫ドメインの管理者は、グローバル コンフィギュレーションを自分のローカル環境に調整できます。
- イベントを生成し、インライン展開では、違反パケットをドロップします。 には、`stream_tcp` インスペクタのルール（GID 129）を有効にします。

TCP ストリームのリアセンブルのためのベストプラクティス

`stream_tcp` インスペクタは、TCP セッションでのサーバーからクライアントへの通信ストリーム、クライアントからサーバーへの通信ストリーム、またはその両方の通信ストリームに含まれるすべてのパケットを収集してリアセンブルします。TCP ストリームアセンブリでは、Snort は特定のストリームに含まれる個々のパケットだけを検査するのではなく、リアセンブルされた単一のエンティティ（プロトコルデータユニット（PDU））としてストリームを検査できます。PDU が大きい場合、ルールエンジンはその PDU をいくつかの部分に分割します。

ストリームのリアセンブルにより、Snort は、個々のパケットを検査する場合には検出できない可能性のあるストリームベースの攻撃を識別できます。リアセンブルする通信ストリームはネットワークの必要に基づいて指定できます。たとえば、Web サーバー上のトラフィックをモニタする際に、独自の Web サーバーから不正なトラフィックを受信する可能性が低いため、クライアントトラフィックだけを検査するという場合もあります。

`stream_tcp` インスペクタごとに、`binder` の設定で TCP ポートのリストを指定できます。TCP ストリームインスペクタは、トラフィックを識別してリアセンブルするように設定されたポートを自動的かつ透過的に組み込みます。適応型プロファイルの更新が有効になっている場合、リアセンブルするトラフィックを識別するサービスを、ポートの代わりとして、あるいはポートと組み合わせてリストすることもできます。

次の Snort インスペクタの `binder` の設定で TCP ポートを指定します。

- `dnp3`
- `ftp_server`
- `gtp_inspect`（デフォルトで提供されるポート）
- `http_inspect`
- `imap`
- `iec104`（デフォルトで提供されるポート）
- `mms`（デフォルトで提供されるポート）

- modbus（デフォルトで提供されるポート）
- pop
- sip
- smtp
- ssh
- ssl
- telnet

（注） 複数のトラフィックタイプ（クライアント、サーバー、両方）をリアセンブルすると、Snort のリソースの需要が増加する可能性があります。

ストリーム TCP インスペクタのパラメータ

ストリーム TCP リアセンブルの設定

binder インスペクタは、ネットワーク分析ポリシー（NAP）の TCP ストリームリアセンブル設定を定義します。TCP ストリーム リアセンブル ポリシーを適用するホスト IP アドレスを指定します。ストリーム TCP インスペクタは、NAP の binder に設定されているポートに自動的に関連付けられます。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

（注） システムは、各リーフドメインに個別のネットワークマップを作成します。マルチドメイン展開では、実際の IP アドレスを使用してこの設定を抑制すると、予期しない結果になる可能性があります。 上書き対応オブジェクトを使用すると、子孫ドメインの管理者は、グローバルコンフィギュレーションを自分のローカル環境に調整できます。

デフォルトポリシーの default 設定では、別のターゲットベースのポリシーでカバーされていないモニタ対象ネットワークセグメントのすべての IP アドレスが指定されます。したがって、デフォルトポリシーの IP アドレスまたは CIDR ブロック/プレフィックス長は指定できず、また指定する必要もありません。また、別のポリシーでこの設定を空白にしたり、any を表すアドレス表記（0.0.0.0/0 または ::/0）を使用したりすることはできません。

policy

ターゲットホスト（複数可）のオペレーティングシステムを指定します。オペレーティングシステムは、適切な TCP リアセンブルポリシーとオペレーティングシステムの特性を決定します。ストリーム TCP インスペクタごとに policy パラメータを 1 つだけ定義できます。

（注） policy パラメータを first に設定すると、Snort はある程度の保護を提供できますが、攻撃は見逃します。TCP ストリームインスペクタの policy パラメータを編集して、適切なオペレーティングシステムを指定する必要があります。

型：列挙体

有効な値： policy パラメータのオペレーティングシステムのタイプを設定します。

表 *32 : TCP* オペレーティング システム ポリシー

ポリシー	オペレーティング システム
first	不明な OS
last	Cisco IOS
bsd	AIX FreeBSD OpenBSD
hpux_10	HP-UX 10.2 以前
hpux_11	HP-UX 11.0 以降
irix	SGI Irix
linux	Linux 2.4 カーネル Linux 2.6 カーネル
macos	Mac OS 10（Mac OS X）
old_linux	Linux 2.2 以前のカーネル
solaris	Solaris OS SunOS
vista	Windows Vista
windows	Windows 98 Windows NT Windows 2000 Windows XP
win_2003	Windows 2003

デフォルト値：bsd

max_window

受信側ホストが許可する TCP ウィンドウの最大サイズを指定します。65535 未満の整数を指定するか、または 0 を指定して TCP ウィンドウのサイズの検査を無効にすることができます。

注意 max_window の上限は、RFC 1323 で許可されている最大ウィンドウサイズです。上限を設定して、攻撃者が検出を回避するのを防ぐことができますが、TCP ウィンドウの最大サイズが非常に大きいと、自発的にサービス拒否が発生する可能性があります。

型：整数

有効な範囲：0 ～ 1,073,725,440

デフォルト値：0

overlap_limit

各 TCP セッションで許可される重複セグメントの最大数を指定します。重複セグメントを無制限に許可するには、0 を指定します。0 ～ 255 の数値を設定すると、セッションのセグメントのリアセンブルが停止します。

ルール 129:7 を有効にしてイベントを生成し、インライン展開では、違反パケットをドロップします。。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：0

max_pdu

リアセンブルされたプロトコルデータユニット（PDU）の最大サイズを指定します。

型：整数

有効な範囲：1460 ～ 32768

デフォルト値：16384

reassemble_async

トラフィックが両方向で確認される前に、データがリアセンブルのキューに入るようにします。モニタ対象ネットワークが非同期ネットワークの場合、reassemble_async パラメータを有効にする必要があります。非同期ネットワークでは、一度に一方向のトラフィックと 1 つのフローのみが許可されます。reassemble_async パラメータが有効になっている場合、Snort はパフォーマンスを向上させるために TCP ストリームをリアセンブルしません。

（注） ストリーム TCP インスペクタは、すべての場合において非対称トラフィックを正しく処理できるわけではありません。たとえば、HTTP HEAD 要求への応答により、HTTP インスペクタが同期しなくなる可能性があります。IDS モードでは、TCP 確認応答がないため、回避がはるかに簡単になります。IPS モードの場合、ルールエンジンでフローの両側を検査できる場合にのみデバイスをデプロイすることをお勧めします。

`reassemble_async` パラメータは Cisco Secure Firewall Threat Defenseルーテッドインターフェイスと透過的インターフェイスの場合は無視されます。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`true`

require_3whs

ストリーム TCP インスペクタが中間ストリームセッションの追跡を停止するまでの、起動からの秒数を指定します。いつ発生したかに関係なく、すべての中間 TCP セッションを追跡するには、`-1` を指定します。

Snort はほとんどのプロトコルストリームを同期しません。Snort は、ハンドシェイクオプション（タイムスタンプ、ウィンドウスケール、またはMSS）のいずれかが必要な場合、常にSYNを感知します。通常、中間ピックアップを許可しても IPS の有効性は向上しません。

型：整数

有効な範囲：`-1` 〜 `2,147,483,647 (max31)`

デフォルト値：`-1`

queue_limit.max_bytes

TCP接続の一方の側でセッションのキューに入れるバイトの最大数を指定します。バイト数を無制限に許可するには、`0` を指定します。

注意 `queue_limit.max_bytes` パラメータのデフォルト設定を変更する前に、Cisco TAC に連絡することをお勧めします。

型：整数

有効な範囲：`0` 〜 `4,294,967,295 (max32)`

デフォルト値：`4,194,304`

queue_limit.max_segments

TCP接続の一方の側でセッションのキューに入れるデータセグメントの最大数を指定します。データセグメントの数を無制限に許可するには、`0` を指定します。

注意 queue_limit.max_segments パラメータのデフォルト設定を変更する前に、Cisco TAC に連絡することをお勧めします。

型：整数

有効な範囲：0 ～ 4,294,967,295 (max32)

デフォルト値：3072

small_segments.count

連続する小さな TCP セグメントの予想数を超える数を指定します。連続した小さな TCP セグメントのカウントを無視するには、0 を指定します。

small_segments.count パラメータと small_segments.maximum_size パラメータには、同じ型の値を設定する必要があります。両方のパラメータに 0 を指定するか、または各パラメータをゼロ以外の値に設定します。

（注） Snort は、各セグメントの長さが 1 バイトであっても、通常の連続 TCP セグメント数を超える 2,000 の連続セグメントを考慮します。

Snort は、Firewall Threat Defense のルーテッドインターフェイスと透過的なインターフェイスの small_segments.count パラメータを無視します。

ルール 129:12 を有効にして、イベントを生成し、インライン展開では、違反パケットをドロップします。。

型：整数

有効な範囲：0 ～ 2048

デフォルト値：0

small_segments.maximum_size

TCP セグメントを小さい TCP セグメントよりも大きいものとして識別するバイト数を指定します。小さい TCP セグメントのサイズは、1 ～ 2048 バイトの範囲です。小さいセグメントの最大サイズを無視するには、0 を指定します。

Snort は、Firewall Threat Defense のルーテッドインターフェイスと透過的なインターフェイスの small_segments.maximum_size パラメータを無視します。

small_segments.maximum_size パラメータと small_segments.count パラメータには、同じ型の値を設定する必要があります。両方のパラメータに 0 を指定するか、または各パラメータをゼロ以外の値に設定します。

（注） 2048 バイトの TCP セグメントは、標準的な 1500 バイトのイーサネットフレームよりも大きいセグメントです。

ルール 129:12 を有効にして、イベントを生成し、インライン展開では、違反パケットをドロップします。。

型：整数

有効な範囲：0 ～ 2048

デフォルト値：0

session_timeout

Snort が非アクティブな TCP ストリームを状態テーブルに保持する秒数を指定します。指定した時間内にストリームがリアセンブルされない場合、Snort はそのストリームを状態テーブルから削除します。セッションがまだ動作しており、さらにパケットが表示される場合は、Snort はストリームを中間フローとして処理します。

session_timeout パラメータをホストの TCP セッションタイムアウト以上に設定することをお勧めします。

型：整数

有効な範囲：0 ～ 2,147,483,647 (max31)

デフォルト値：180

ストリーム TCP インスペクタのルール

stream_tcp インスペクタのルールを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *33*：ストリーム *TCP* インスペクタのルール

GID:SID	ルール メッセージ
129:1	確立済みセッションの SYN（SYN on established session）
129:2	SYN パケットのデータ（data on SYN packet）
129:3	ストリームで送信されたデータがデータを受け入れない（data sent on stream not accepting data）
129:4	TCP タイムスタンプが PAWS ウィンドウ内にない（TCP timestamp is outside of PAWS window）

GID:SID	ルール メッセージ
129:5	不良セグメント、調整済みサイズ <= 0（非推奨）（bad segment, adjusted size <= 0 (deprecated)）
129:6	ポリシーが許可するよりも大きいウィンドウ サイズ（スケーリング後）（window size (after scaling) larger than policy allows）
129:7	重複 TCP パケット数の制限に到達した（limit on number of overlapping TCP packets reached）
129:8	TCP リセット送信後にストリームで送信されたデータ（data sent on stream after TCP reset sent）
129:9	TCP クライアントがハイジャックされた可能性がある、別のイーサネットアドレス（TCP client possibly hijacked, different ethernet address）
129:10	TCP サーバーがハイジャックされた可能性がある、別のイーサネットアドレス（TCP server possibly hijacked, different ethernet address）
129:11	TCP フラグが設定されていない TCP データ（TCP data with no TCP flags set）
129:12	連続した TCP の小さなセグメントがしきい値を超えている（consecutive TCP small segments exceeding threshold）
129:13	4 ウェイハンドシェイクが検出された（4-way handshake detected）
129:14	TCP タイムスタンプがない（TCP timestamp is missing）
129:15	外部ウィンドウをリセット（reset outside window）
129:16	FIN 番号が前の FIN よりも大きい（FIN number is greater than prior FIN）
129:17	ACK 番号が前の FIN より大きい（ACK number is greater than prior FIN）
129:18	TCP リセットの受信後にストリームで送信されたデータ（data sent on stream after TCP reset received）
129:19	データを受信する前に TCP ウィンドウが閉じた（TCP window closed before receiving data）
129:20	3 ウェイハンドシェイクなしの TCP セッション（TCP session without 3-way handshake）

ストリーム TCP インスペクタの侵入ルールのオプション

stream_reassemble

一致するトラフィックでTCPストリームのリアセンブルを有効にするかどうかを指定します。`stream_reassemble` ルールオプションには、`stream_reassemble.action`、`stream_reassemble.direction`、`stream_reassemble.noalert`、および `stream_reassemble.fastpath` の 4 つのパラメータが含まれます。

シンタックス：`stream_reassemble: <enable|disable>, <server|client|both>, noalert, fastpath;`

例：`stream_reassemble: disable,client,noalert;`

stream_reassemble.action

ストリームのリアセンブルを停止または開始します。

型：列挙体

シンタックス：`stream_reassemble: <action>;`

有効な値：`disable` または `enable`

例： `stream_reassemble: enable;`

stream_reassemble.direction

指定された方向にアクションが適用されます。

型：列挙体

シンタックス：`stream_reassemble: <direction>`

有効な値：`client`、`server`、`both`

例： `stream_reassemble: both;`

stream_reassemble.noalert

ルールが一致したときにアラートを出しません。`stream_reassemble.noalert` パラメータはオプションです。

シンタックス：`stream_reassemble: noalert;`

例：`stream_reassemble: noalert;`

stream_reassemble.fastpath

必要に応じて、セッションの残りを信頼します。`stream_reassemble.fastpath` パラメータはオプションです。

シンタックス：`stream_reassemble: fastpath;`

例：`stream_reassemble: fastpath;`

stream_size

ストリームサイズチェックの検出オプション。TCP シーケンス番号によって決定される監視されたバイト数に従って、ルールがトラフィックを照合できるようにします。`stream_size` ルールオプションには、`stream_size.direction` と `stream_size.range` の 2 つのパラメータが含まれます。

シンタックス：`stream_size: <server|client|both|either>, <operator><number>;`

例： `stream_size: client, <6;`

stream_size.direction

比較はフローの方向に適用されます。

型：列挙体

シンタックス：`stream_size: <direction>;`

有効な値は、次のとおりです。

- `either`
- `to_server`
- `to_client`
- `both`

例：`stream_size: to_client;`

stream_size.range

ストリームサイズが指定した範囲内にあるかどうかを確認します。`range` 演算子と 1 つ以上の正の整数を指定します。

型：間隔

構文: stream_size:<range_operator><positive integer> ;または stream_size: ;

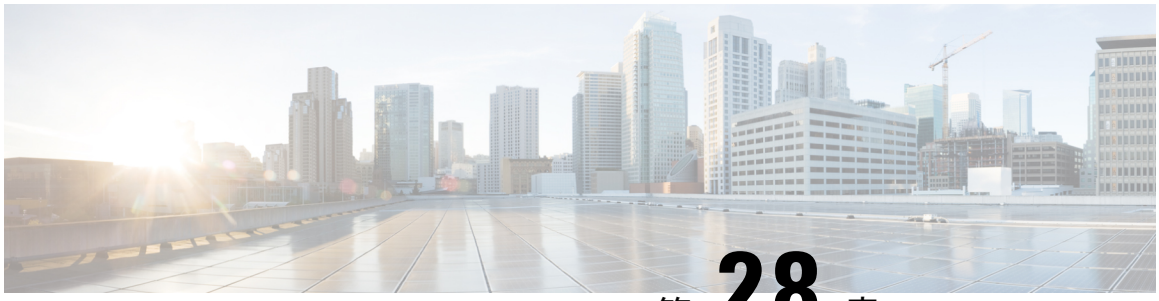
有効な値：1 つ以上の一連の正の整数と有効な範囲に指定されている `range_operator` の 1 つ。

例：`stream_size: >6;`

表 *34 :* 範囲の形式

範囲の形式	演算子	説明
operator i		
	<	より少ない
	>	右辺と比較して大きい

範囲の形式	演算子	説明
	=	等しい
	≠	等しくない
	≤	以下
	≥	以上
j operator k		
	<>	j よりも大きく、k よりも小さい
	<=>	j 以上で k 以下



第 28 章

ストリーム UDP インスペクタ

- ストリーム UDP インスペクタの概要 （241 ページ）
- ストリーム UDP インスペクタを設定するためのベストプラクティス （242 ページ）
- ストリーム UDP インスペクタのパラメータ （242 ページ）
- ストリーム UDP インスペクタのルール （242 ページ）
- ストリーム UDP インスペクタの侵入ルールのオプション （242 ページ）

ストリーム UDP インスペクタの概要

タイプ	インスペクタ（ストリーム）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	なし
有効	`true`

User Datagram Protocol（UDP）はコネクションレス型の低遅延トランスポート層プロトコルです。UDP を使用すると、受信側から同意書が提供される前に、2 つのネットワークエンドポイント間のステートレス通信が可能になります。メッセージヘッダーとデータの整合性を評価するには、UDP はチェックサムを使用します。

`stream_udp` は IP データグラムヘッダー内の送信元および接続先の IP アドレスフィールドと、UDP ヘッダー内のポートフィールドを確認して、フローの方向を決定し、セッションを識別します。セッションは、設定可能なタイマーの期限が切れるか、または他のエンドポイントが到達不能という ICMP メッセージをいずれかのエンドポイントが受信したときに終了します。

UDP ストリームインスペクタはイベントを生成しません。パケットデコーダのルール（GID 116）を有効にして、UDP ヘッダーの異常を検出できます。

ストリームUDPインスペクタを設定するためのベストプラクティス

stream_udp インスペクタを設定する場合は、次のベストプラクティスを考慮してください。

- ホストまたはエンドポイントに適用するセッションタイムアウトごとに stream_udp インスペクタを作成します。ストリーム UDP インスペクタは、session_timeout を binder インスペクタで定義されている UDP ホストに関連付けます。

  同じネットワーク分析ポリシーに複数のバージョンの stream_udp インスペクタを含めることができます。

- パケットデコーダのルール（GID 116）を有効にして、UDP ヘッダーの異常を検出します。

ストリーム UDP インスペクタのパラメータ

session_timeout

UDP インスペクタが非アクティブな UDP ストリームを状態テーブルに保持する秒数を指定します。Snort が同じフローキーを持つ UDP データグラムを次に検出すると、以前のフローのセッションタイムアウトが期限切れになっているかどうかが確認されます。タイムアウトの期限が切れると、Snort はフローを閉じて新しいフローを開始します。Snort は、基本のストリーム設定に関連付けられた古いフローを確認します。

型：整数

有効な範囲：0 ～ 2,147,483,647 (max31)

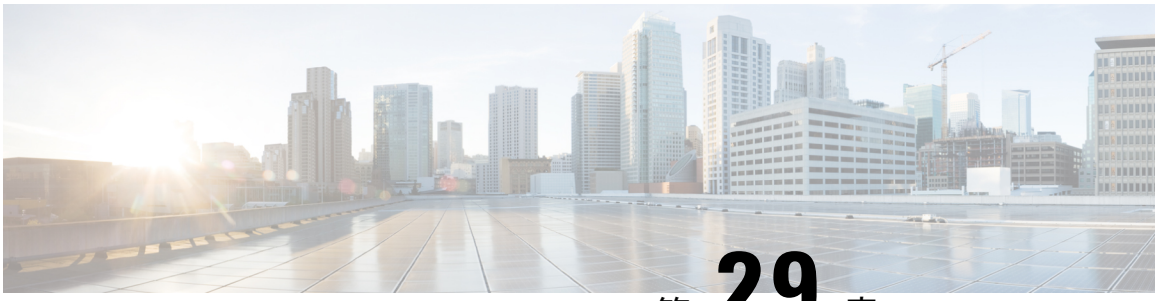
デフォルト値：30

ストリーム UDP インスペクタのルール

stream_udp インスペクタには、関連付けられたルールがありません。

ストリームUDPインスペクタの侵入ルールのオプション

stream_udp インスペクタには、侵入ルールのオプションはありません。



第 29 章

Telnet インスペクタ

- Telnet インスペクタの概要 （243 ページ）
- Telnet インスペクタのパラメータ （244 ページ）
- Telnet インスペクタのルール （245 ページ）
- Telnet インスペクタの侵入ルールのオプション （245 ページ）

Telnet インスペクタの概要

タイプ	インスペクタ（サービス）
使用方法	検査
インスタンス タイプ	マルチトン
その他のインスペクタが必要	`stream_tcp`
有効	`false`

Telnet は、TCP 上で 8 ビットバイトの通信チャネルを作成するアプリケーション層プロトコルです。Telnet は、ネットワーク仮想端末を使用して、クライアントとリモートホスト間の通信を行います。Telnet サーバーは TCP ポート 23 を使用します。

`telnet` インスペクタは、Telnet コマンドシーケンスとオプションのネゴシエーションを検出することで Telnet データバッファを正規化します。`telnet` インスペクタは、パケットから Telnet コマンドシーケンス（RFC 854）を除去します。`telnet` インスペクタは、Telnet 暗号化オプション（RFC 2946）を確認することで、暗号化された Telnet 接続を検出できます。

Telnet インスペクタのパラメータ

Telnet サービスの設定

`binder` インスペクタは、Telnet サービスの設定を定義します。詳細については、『バインダインスペクタの概要 （15 ページ）』を参照してください。

例：

```
[
    {
        "when": {
            "service": "telnet",
            "role": any
        },
        "use": {
            "type": "telnet"
        }
    }
]
```

ayt_attack_thresh

連続する Are You There（AYT） Telnet コマンドの最大数を指定します。`telnet` インスペクタは、`ayt_attack_thresh` 値を超える連続した AYT コマンドの数を検出し、アラートを生成します。`ayt_attack_thresh` パラメータは、Telnet の BSD 実装に関連する特定の脆弱性に対処します。`ayt_attack_thresh` パラメータを無効にするには、`-1` を指定します。ルール 126:1 を有効にし、このパラメータのイベントを生成し、インライン展開では、違反パケットをドロップします。。

型：整数

有効な範囲：`-1` 〜 `2,147,483,647 (max31)`

デフォルト値：`-1`

encrypted_traffic

暗号化された Telnet トラフィックを検出するかどうかを指定します。ルール 126:2 を有効にし、このパラメータのイベントを生成し、インライン展開では、違反パケットをドロップします。。

型：ブール値

有効な値：`true`、`false`

デフォルト値：`false`

normalize

Telnet トラフィックを正規化するかどうかを指定します。`telnet` インスペクタは、telnet エスケープシーケンスを除外することで、Telnet トラフィックを正規化します。有効な侵入ルール

で raw コンテンツパラメータが指定されている場合、このルールでは telnet インスペクタが作成した正規化された Telnet バッファを無視します。

型：ブール値

有効な値：true、false

デフォルト値：false

Telnet インスペクタのルール

telnet インスペクタを有効にし、イベントを生成し、インライン展開では、違反パケットをドロップします。。

表 *35 : Telnet* インスペクタのルール

GID:SID	ルール メッセージ
126:1	連続した Telnet AYT コマンドがしきい値を超えている（consecutive Telnet AYT commands beyond threshold）
126:2	Telnet トラフィックが暗号化されている（Telnet traffic encrypted）
126:3	Telnet サブネゴシエーション開始コマンドにサブネゴシエーション終了がない（Telnet subnegotiation begin command without subnegotiation end）

Telnet インスペクタの侵入ルールのオプション

telnet インスペクタには、侵入ルールのオプションがありません。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。