



スタンドアロン Content Engine の Rules Template の設定

この章では、スタンドアロン Content Engine の Rules Template (ルール テンプレート) の設定方法について説明します。Rules Template を使用して、Content Engine が HTTP、HTTPS、MMS、RTSP のトラフィックをフィルタリングするルールを指定します。これらの設定済みのルールにより、特定のヘッダーが書き換えられたり、要求がリダイレクトされたり、他の方法で要求が処理されることがあります。

この章の構成は、次のとおりです。

- [Rules Template の概要 \(P. 13-2\)](#)
- [アクションとパターンの概要 \(P. 13-5\)](#)
- [Rules Template の設定 \(P. 13-21\)](#)
- [設定済みルールの統計情報の表示 \(P. 13-31\)](#)
- [設定済みルールの統計情報のクリア \(P. 13-32\)](#)



(注) 「HTTP トラフィック」という用語は、HTTP、FTP-over-HTTP、HTTPS-over-HTTP を含む、HTTP からの要求を表すために使用されます。Rules Template は、FTP ネイティブ要求に対してはサポートされません。

この章に記述されている CLI コマンドの構文と使用方法については、『Cisco ACNS Software Command Reference, Release 5.3』の資料を参照してください。

Rules Template の概要

Rules Template 機能を使用し、アクションやパターンで識別される 1 組のルールを指定します。この機能によって、HTTP、HTTPS、MMS、RTSP のトラフィックをフィルタリングするときに個々の規則を使用するようにスタンドアロン Content Engine を設定できます。この機能は一般に、要求された Web ページが既知のインターネット ワームのパターンに一致するかどうかをチェックし、一致した場合には要求を自動的にブロックし、組織内でインターネット ワームやウイルスが蔓延するのを防ぐように Content Engine を設定する目的で使用します。

Content Engine 上でルール処理を有効にしていた (Content Engine で Rules Template 機能を有効にし、Content Engine 用のルールを設定していた) 場合には、Content Engine は、着信の各クライアント要求をチェックし、要求されたコンテンツに一致するルールパターンがあるかどうかを調べます。要求に一致するルールパターンがあると、Content Engine は、指定されたアクション (ポリシー) を使用して、この着信トラフィックを処理します。

Content Engine により、着信要求を次のパターンと照合できます。

- IP アドレス、ネットワーク マスク、ポート リストを含む、コンテンツを要求しているクライアントの IP アドレス (送信元 IP アドレス) 内のパターン
- IP アドレス、ネットワーク マスク、ポート リストを含む、オリジン Web または メディア サーバの IP アドレス (送信先 IP アドレス) 内のパターン
- URL の正規表現
- URL のドメイン部の正規表現
- クライアントが要求している Web オブジェクトの MIME タイプ
- ドメイン名を記号化する正規表現
- 次のものを含む、要求の中で送信されるヘッダー
 - どのクライアント ソフトウェアが要求を発行しているかを示す「要求のユーザーエージェント」
 - ブラウザがどの Web ページからこのリンクにジャンプしたかを示す「参照元 (referer)」
 - 要求ライン自体を示す「要求ライン (Request Line)」

Rules Template は、HTTP、FTP、HTTPS の各プロトコルに最適な機能です。表 13-1 では、HTTP、FTP、HTTPS のほか、MMS や RTSP などのストリーミング メディア オブジェクトプロトコルをサポートするポリシーを示しています。

表 13-1 Rules Template ポリシー

ポリシー (アクション)	HTTP 要求	MMS 要求	RTSP 要求
要求を許可します。	表 13-2 を参照してください。	*	*
ユーザ名を要求ヘッダーに追加します。			
要求をブロックします。		*	*
HTTP 応答ヘッダーをオーバーライドして、オブジェクトをキャッシングします。			
HTTP 応答ヘッダに応じてオブジェクトをキャッシングします。			
要求に対する認証をバイパスします。		*	*
特定のオブジェクト鮮度計算係数を使用します。			
要求をリセットします。			
オブジェクトをキャッシングしません。			
要求のアップストリーム プロキシをバイパスします。			
別の URL に要求をリダイレクトします。			*
オリジン サーバでオブジェクトを再検証します。			
URL を書き換えます。		*	*
指定された HTTP および HTTPS 要求について URL フィルタリングは行いません。	*		
特定の ICAP サーバを使用します。			
特定のアップストリーム プロキシを使用します。			
要求に特定のサーバを使用します。			
クライアントに送信する応答に ToS または DSCP を設定します。			
クライアントに送信する応答に ToS または DSCP を設定します。			

プロトコルごとにサポートされるルール アクション

表 13-2 では、ACNS 5.3.1 またはそれ以降のソフトウェアを実行しているスタンドアロン Content Engine でプロトコルごとにサポートするルール アクションを示しています。星印 (*) は、該当のプロトコルで 1 つのルール アクションがサポートすることを示しています。WCCP は透過的サポートを意味します。「*1」は、ACNS 5.1.5 またはそれ以降のソフトウェアで、特定プロトコルで 1 つのルール アクションがサポートされることを示します。

RTSP ストリーミングプロトコルについて、**redirect** および **redirect_url_for_cdn rule** の各ルール アクションは、RealMedia Player からの RTSP 要求に対してはサポートされますが WMT 要求に対してはサポートされません。したがって、Windows Media Player からの RTSP 要求に対し、この 2 つのルール アクションはサポートされません。たとえば、Windows Media Services 9 (WMS 9) は RTSP 要求に対する **block**、**reset**、**rewrite**、および **allow** の各ルール アクションをサポートします。しかし、RTSP 要求に対する **redirect** および **redirect_url_for_cdn** の各ルール アクションはサポートされません。

表 13-2 プロトコルごとにサポートするルールアクションのマトリックス (ACNS 5.3.1 またはそれ以降のソフトウェア)

ルールアクション	プロトコル							ストリーミングプロトコル			
	HTTP-over-HTTP	FTP-over-HTTP	HTTP Sover-HTTP	HTTP WCCP	FTP-WCCP (Native FTP)	HTTPS-WCCP	HTTPS-WCCP (トンネル化、リリース 5.1.5 ACNS 5.1.5 ソフトウェア使用可)	RTSP	MMSU	MMST	MMS
allow	*	*	*	*		*	ここにはいずれのルールも適用されません。	*	*	*	*
append-username-header	*			*		*					
block	*	*	*	*		*		*	*	*	*
cache-cookie	*			*		*					
cache-non-cacheable	*			*		*					
cache-only	*			*		*			*	*	*
dscp	*	*		*		*					
freshness-factor	*			*		*					
insert-no-cache	*	*		*		*					
no-auth	*	*		*		*					
no-cache	*	*		*		*			*	*	*
no-persistent-connection	*					*					
no-proxy	*	*		*							
no-url-filtering	*		*	*		*					
redirect	*	*		*		*1		*			
redirect-url-for-cdn	*							*			
refresh	*	*		*		*					
reset	*	*		*		*		*	*	*	*
rewrite	*			*		*1		*	*	*	*
use-dns-server	*			*							
use-icap-service	*										
use-proxy	*	*		*							
use-server	*			*		*1					
use-xforward-clt-ip	*					*					

アクションとパターンの概要

ルールは、パターンとアクションで指定します。

- パターンは、着信要求の制限を指定します。たとえば、パターンによって、送信元 IP アドレスがサブネット範囲 172.16.*.* に含まれることを指定できます。

パターンを定義するときには、次の情報を指定します。

- パターンタイプ (たとえば、送信元 IP アドレスを表す「src-ip」)
- パターン値 (たとえば、送信元 IP アドレスがこのサブネット範囲に含まれることを示す「172.16.*.*」)
- このパターンの追加先となるパターンリストの番号 (たとえば、パターンリスト 10)

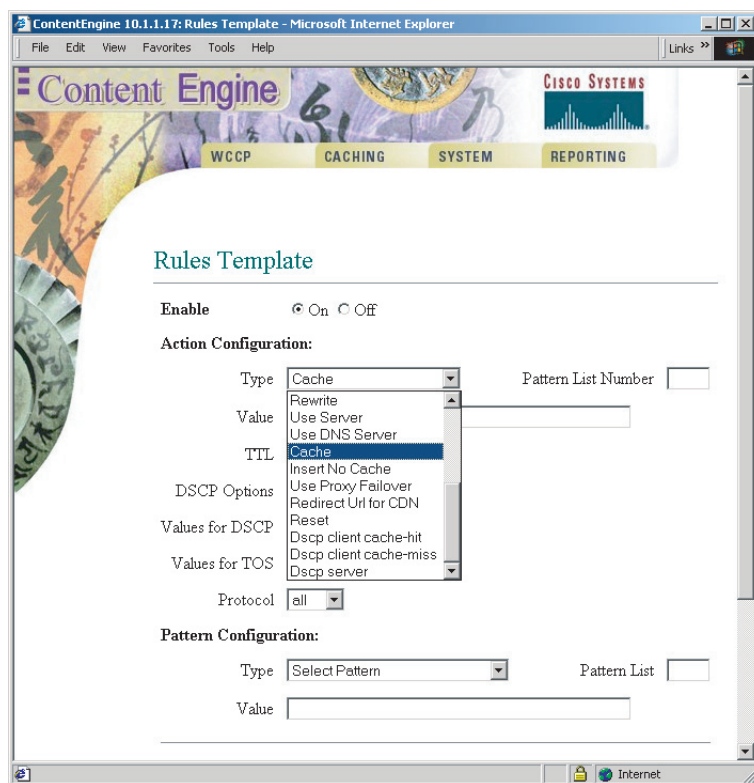
サポートされているパターンの詳細リストについては、表 13-3 を参照してください。

- 着信要求が指定のパターンに一致すると、要求に基づいて 1 つのアクションが実行されます。アクションは、Content Engine が要求の処理時に実行します。たとえば、要求をブロックするアクションや、代替プロキシを使用するアクションなどがあります。サポートされているアクションの詳細リストについては、表 13-4 を参照してください。

Content Engine のルールは、動的に追加、表示、削除できます。ルールは、該当の Content Engine CLI コマンドや GUI を使用して NVRAM などの永続的ストレージに書き込まれるため、再起動後も保持されます。Content Engine がサポートするルール数を制限するのは、システムリソースです。ルールはリソースを消費するため、定義するルールの数が多いほど、Content Engine のパフォーマンスに大きな影響を及ぼします。

Content Engine CLI または GUI (図 13-1 を参照) を使用して、パターンとそれに対応するアクションを指定できます。

図 13-1 Rules Template ウィンドウ





(注) Rules Template ウィンドウを表示するには、Content Engine GUI から **System > Rules Templates** の順に選択します。Rules Template を設定するためにウィンドウを使用する方法に関する詳細情報を表示するには、**HELP** ボタンをクリックします。

rule pattern-list グローバル設定コマンドを使用して、1つのパターンリストを作成し、そのリストに個々のパターンを追加できます。パターンリスト（複数）を定義したら、**rule action** グローバル設定コマンドを使用して、定義済みパターンリストに個々のアクションを関連付けることができます。次の例は、同じパターンリスト番号でさまざまなパターンを設定し、そのパターンリストを1つのアクションに適用することにより、どのようにパターンが AND 演算されるかを示したものです。

```
ContentEngine(config)# rule pattern-list 1 url-regex yahoo
ContentEngine(config)# rule pattern-list 1 dst-port 80
ContentEngine(config)# rule action block pattern-list 1
```

これ以降の説明では、常に CLI メソッドを使用して、スタンドアロン Content Engine での Rules Template の設定方法を示します。Content Engine CLI による Rules Template の詳しい設定方法については、「Rules Template の設定」(P. 13-21) を参照してください。

サポートされているルールパターン

ACNS 5.1 ソフトウェア リリースでは、新しい3つのルールパターン (**groupname**、**username**、および **groupname-regex**) が追加されました。これら3つのルールパターンは、認証済みの NTLM ユーザと LDAP ユーザのグループ名とユーザ名に基づいたアクセス制御ポリシーをサポートします。グループ名に基づいたルールは、NTLM と LDAP を通して認証されたユーザに適用されます。ユーザ名に基づいたルールは、LDAP、NTLM、RADIUS、TACACS+（認証用にユーザ名を含んだ要求認証メソッド）で認証されるユーザに適用されます。

たとえば、次の例は、(**rule enable** グローバル設定コマンドを使用して) Content Engine でのルール処理を有効にし、Engineering グループ内のすべてのエンドユーザが「java」という表現を含んだ FTP URL（クライアント ブラウザからの FTP 要求）のダウンロードをブロックする方法を示しています。

```
ContentEngine(config)# rule enable
ContentEngine(config)# rule pattern-list 1 group-type and
ContentEngine(config)# rule pattern-list 1 groupname Engineering
ContentEngine(config)# rule pattern-list 1 url-regex java
ContentEngine(config)# rule action block pattern-list 1 protocol ftp
```



(注) グループ名とユーザ名に基づいたルールによる許可は、HTTP 要求認証とグループに基づいたアクセスリスト許可が既に与えられている場合にのみ発生します。Rules Template 内の設定とアクセスリストが一致した場合は、アクセスリストが優先されます。

表 13-3 では、パターンリストに追加できるルールパターンのタイプについて説明しています。

表 13-3 サポートされているパターンのタイプ



パターン	説明
domain	URL または Host ヘッダー内のドメイン名を正規表現と照合します。たとえば、「*ibm.*」は、「ibm」サブストリングを含むすべてのドメイン名と一致します。「\foo.com\$」は、「foo.com」サブストリングで終わるすべてのドメイン名と一致します。正規表現の構文では、メタ文字のドル記号「\$」は、このパターンが行の末尾にある場合だけ一致するように指定します。
dst-ip	<p>要求の送信先 IP アドレスおよびネットマスクと、ルールに指定されている IP アドレスおよびネットマスクを照合します。</p> <p> (注) プロキシモードでは、Content Engine が DNS ルックアップを実行して HTTP 要求の送信先 IP アドレスを解決するため、応答時間が長くなり、dst-ip ルールを設定する利点が生かされないことがあります。発信プロキシが設定される場合、キャッシュミス要求は送信先サーバの IP アドレスを調べることなく、Content Engine によって発信プロキシに転送され、最初の Content Engine 上で dst-ip ルールを強制不可にします。</p>
dst-port	要求の送信先ポート番号とルールに指定されているポート番号を照合します。
groupname	<p>LDAP または NTLM で認証されたエンドユーザ（コンテンツを要求している Web クライアント）のグループ名とルールに指定されているグループ名を照合します。たとえば、パターンリスト 1 にグループ名 Engineering を次のように指定します。</p> <pre>ContentEngine(config)# rule pattern-list 1 groupname Engineering</pre> <p>このパターンは、LDAP または NTLM を通して認証されたユーザに対する要求認証にのみ適用できます。このパターンは、正確な文字列比較をサポートし、大文字小文字が区別されます。グループ名の最大長は 255 文字です。有効な文字はアンダースコアと英数字です。Rules Template 内のグループ名設定と、グループ名に基づいたアクセスリストが一致した場合は、アクセスリストのほうが優先されます。</p> <p> ヒント グループ名パターンを使用する場合には、認証グループキャッシュ内の最大グループエントリの正しい数値を必ず設定してください (http authentication cache max-group-entries number グローバル設定コマンド)。この数値は、認証クエリー中に返される最大グループ数（たとえば、AAA サーバで定義されている総グループ数）に対応します。数値は、500 ~ 12000 が可能で、認証グループキャッシュ内のエントリ数は、Content Engine で使用可能なリソースにより異なります。</p>

表 13-3 サポートされているパターンのタイプ (続き)

パターン	説明
groupname-regex	<p>エンドユーザ (コンテンツを要求している Web クライアント) のグループ名と、ルールに指定されている正規表現を照合します。たとえば、正規表現に基づいたポリシーをグループ名に設定したり、1つのパターン リストの同一ライン内にある複数のグループ名を OR 演算する場合に使用します。</p> <p>たとえば、3つのグループ名を OR 演算するには、このコマンドを入力します。</p> <pre>ContentEngine(config)# rule pattern-list 1 groupname-regex Engineering Marketing Finance\</pre> <p>前記のいずれかのグループ名が要求の中のいずれかのグループ名と一致すると、指定のアクションが実行されます。</p>
group-type	パターン リストを AND と OR のどちらのタイプにするかを指定します。デフォルトは OR です。
header-field	<p>要求ヘッダー フィールドパターン。 referer、 request-line、 および user-agent の各要求ヘッダー フィールドパターンは、 block、 reset、 redirect、 および rewrite の各アクションに対してサポートされます。 referer パターンは要求内の Referer ヘッダーと照合され、 request-line パターンは要求の先頭行と照合され、 user-agent パターンは要求内の User-Agent ヘッダーと照合されます。</p>
header-field-sub	要求ヘッダー フィールドパターンと代替の置換パターン。
icap-service-name	使用する ICAP サービスの名前を指定します。
mime-type	<p>応答の MIME タイプと、ルールに指定されている MIME タイプ文字列 (たとえば、 http://www.faqs.org/rfcs/rfc2046.html にある RFC 2046 に規定されている「image/gif」) を照合します。サブストリング (たとえば、「java」) を指定し、「java」のサブストリングをもつすべての MIME タイプ (たとえば、「application/x-javascript」) に、そのサブストリングを適用することができます。</p>
src-ip	要求の送信元 IP アドレスおよびネットマスクと、ルールに指定されている IP アドレスおよびネットマスクを照合します。

表 13-3 サポートされているパターンのタイプ (続き)

パターン	説明
username	<p>LDAP、NTLM、RADIUS、TACACS+ を通して認証されたエンド ユーザ (コンテンツを要求している Web クライアント) のユーザ名とルールに指定されているユーザ名 (複数) を照合します。LDAP、RADIUS、TACACS+ の認証の場合のユーザ名の最大長は 255 文字です。NTLM 認証の場合のユーザ名の最大長の詳細については、下記を参照してください。</p> <p>有効な文字はアンダースコアと英数字です。このパターンは、正確な文字列比較をサポートし、大文字小文字が区別されます。</p> <p>同じパターン リストの同一ラインに複数のユーザ名を指定する場合は、区切り文字を使用します。</p> <p>次に例を示します。</p> <pre>ContentEngine(config)# rule pattern-list 1 username jdoe8,dsmith7,jsmith50</pre> <p>デフォルトでは、照合時にはドメイン名は考慮されず、ユーザ名のみが照合されます。照合にドメイン名とユーザ名を含める場合は、<code>domainname\username</code> のように指定します。</p> <p>次に例を示します。</p> <pre>ContentEngine(config)# rule pattern-list 1 username domain cisco\jdoe8</pre> <p>NTLM 認証の場合は、<code>domain\username:password:NTLM</code> 文字列を 50 文字以下にする必要があります。この文字列が 50 文字を超えると、ドメイン名が切り詰められるため、ルールのユーザ名パターンが一致なくなります。その場合は、システム ログにエラー・メッセージが生成されます。</p> <p>特定ドメイン内のすべてのユーザを照合するには、次のように入力します。</p> <pre>ContentEngine(config)# rule pattern-list 1 username domain domainname*</pre> <p><i>domainname</i> とは、ドメインの名前 (たとえば、<code>cisco</code>) です。</p>
URL-regex	<p>URL と、ルールに指定されている正規表現を照合します。この照合では大文字 / 小文字は区別されません。次の Web サイトに示されている構文の正規表現を指定してください。</p> <p>http://yenta.www.media.mit.edu/projects/Yenta/Releases/Documentation/regex-0.12/</p>
URL-regsub	<p>rewrite と redirect のアクションに対して、URL と正規表現を照合し、パターンの代替仕様に基づいて新しい URL を作成します。この照合では大文字 / 小文字は区別されません。有効な代替インデックスの範囲は 1 ~ 9 です。</p>

rule グローバル設定コマンドによるスタンドアロン Content Engine へのパターンの指定方法については、「[スタンドアロン Content Engine のためのルールの設定](#)」(P. 13-22) を参照してください。

サポートされているルール アクション

サポートされているルール アクションのリストを表示するには、**rule action ?** グローバル設定コマンドを入力します。

```
ContentEngine(config)# rule action ?
allow                Allow the request
append-username-header Append the username in the header to the request
                    sent to the server
block                Block
cache-cookie         Cache request containing Cookies
cache-non-cacheable Cache this object (Overriding HTTP response headers)
cache-only           Cache this object only
dscp                 Set IP TOS/DSCP (Differentiated Services) Field
freshness-factor    Caching heuristic modifiers
insert-no-cache      Insert no-cache headers to the response
no-auth              Do not authenticate
no-cache             Do not cache this object
no-persistent-connection Dont use persistent connection to all/client/server
                    connections
no-proxy             Do not use any upstream proxy
no-url-filtering     Do not use URL filtering
redirect             Redirect request to rewritten URL
redirect-url-for-cdn Redirect alternate url for cdn content
refresh             Revalidate the object with the web server
reset               Issue a TCP RST
rewrite              Rewrite URL and fetch
use-dns-server       Use a specific DNS server
use-icap-service     Use a specific ICAP service
use-proxy            Use a specific upstream proxy
use-server           Use a specific server
use-xforward-clt-ip Client IP in x-forwarded header will be used for
                    filtering.
```

表 13-4 では、定義済みのパターン リストに関連付けることができるアクション タイプを説明しています。



(注)

ACNS 5.1 またはそれ以降のソフトウェアでは、最大 520 のアクションがサポートされています。

表 13-4 サポートされているアクションのタイプ

アクション	説明
allow	着信要求を許可します。
append-username-header	キャッシュ ミス時にユーザ名ヘッダーを追加します。
block	着信要求をブロックします。
cache-cookie	クッキーを含んだ要求をキャッシングします。
cache-non-cacheable	HTTP 応答ヘッダーをオーバーライドして、オブジェクトをキャッシングします。

表 13-4 サポートされているアクションのタイプ (続き)

アクション	説明
cache-only	HTTP 応答ヘッダーに応じてオブジェクトをキャッシングします。オブジェクトが一致し、HTTP によるキャッシングが可能な場合にのみ、このオブジェクトをキャッシングします。ルール (複数) にこのアクションを指定した場合、オブジェクトがキャッシュに保存されるのは、オブジェクトが少なくとも 1 つの cache-only ルールに一致し、オブジェクトサイズチェックや no-cache-on-authenticated-object チェックなど、他のキャッシング制限をすべて通過した場合のみです。オブジェクトがどの cache-only ルールにも一致しない場合、そのオブジェクトはキャッシングされません。
dscp client	クライアントの IP の ToS/DSCP (Type of Service/Differentiated Services Code Point) フィールド応答を設定します。ToS または DSCP を設定することをパケットマーキングと言い、このマーキングによって、ネットワークデータを複数の優先度レベルまたはサービスのタイプに区分できます。
dscp client cache-hit	キャッシュヒットによる DSCP クライアントへの応答
dscp client cache-miss	キャッシュミスによる DSCP クライアントへの応答
dscp server	オリジンサーバへの要求用に IP の ToS または DSCP コードポイントフィールドを設定します。
dscp server match-client	クライアントの ToS または DSCP 値を使用します。
dscp server set-dscp	DSCP 値を指定します。DSCP 値のリストについては、表 11-1 を参照してください。
dscp server set-tos	ToS (Type of Service) 値を指定します。ToS 値のリストについては、表 11-2 を参照してください。
freshness-factor	要求 URL が指定の正規表現と一致した場合に、TTL (存続可能時間) を決定します。refresh 設定は、freshness-factor 設定より優先されます。
insert-no-cache	no-cache ヘッダーを応答に挿入します。
no-auth	認証を行いません。
no-cache	このオブジェクトをキャッシングしません。no-cache アクションと selective-cache アクションの両方が一致した場合には、no-cache が優先します。
no-persistent-connection all	どの接続にも永続接続を使用しません。
no-persistent-connection client	クライアント接続にのみ永続接続を使用しません。
no-persistent-connection server	サーバ接続にのみ永続接続を使用しません。
no-proxy	キャッシュミスの場合に、設定済みのアップストリームプロキシを使用せず、ダイレクトにサーバに接続します。
no-url-filtering	特定の HTTP および HTTPS 要求について URL フィルタリングをバイパスします。この機能は、ローカル リスト URL フィルタリングだけでなく、Websense、SmartFilter、または ACNS 5.2.3 またはそれ以降のソフトウェアの N2H2 URL フィルタリングをサポートしています。

表 13-4 サポートされているアクションのタイプ (続き)

アクション	説明
redirect-url-for-cdn	ACNS ネットワーク コンテンツの場合に、元の要求を代替の URL にリダイレクトします。このルールアクションは、Content Distribution Manager に登録された Content Engine でのみサポートされます。つまり、スタンドアロン Content Engine ではサポートされません。
redirect	元の要求を指定の URL にリダイレクトします。リダイレクトは、RADIUS サーバに redirect が設定されている場合にのみ、その RADIUS サーバに関連します。
refresh	キャッシュ ヒットの場合に、オブジェクト新鮮度チェックをサーバに要求します。
reset	TCP RST を発行します。このリセット要求は、Code Red または Nimda のウィルス要求をリセットする場合に便利です。
rewrite	オリジナルの要求を指定の URL に書き換えます。Content Engine はキャッシュ内で書き換えられた URL を検索し、キャッシュ ミスの場合は、書き換えられた URL を取り出して、そのオブジェクトを透過的にクライアントに送信します。パフォーマンスに影響を与える可能性があるため、 rewrite ではなく、 redirect ルールを使用することをお勧めします。 URL の書き換えによって、URL のドメイン名が変わる可能性があります。その場合、要求の送信先の新しいサーバの送信先 IP アドレスを見付けるために DNS ルックアップが必要になります。WCCP リダイレクト パケットから得られた元の IP アドレスは使用できません。
selective-cache	HTTP が許可したオブジェクトをキャッシングします。
use-dns-server	指定の DNS サーバを使用します。
use-icap-service	指定のパターン リストで指定の ICAP サービスを使用します。
use-proxy	キャッシュ ミスの場合に、指定のアップストリーム プロキシを使用します。アップストリーム プロキシの IP アドレス (またはドメイン名) とポート番号を指定します。
use-proxy-failover	フェールオーバー機能をサポートします。
use-server	キャッシュ ミス時に、サーバ形式の HTTP 要求を Content Engine から指定の IP アドレスとポートに送信します。
use-xforward-clt-ip	フィルタリング用として、X-forwarded ヘッダー内の クライアント IP アドレスを使用します。

no-url-filtering アクションの例

ここでは、URL filtering 機能を Websense URL フィルタリングでバイパスする **no-url-filtering** アクションの使い方の例を示します。

rule action no-url-filtering コマンドを指定し、それを 特定のパターン リスト (pattern list 100) と関連付けます。**domain** パターン タイプをパターン リスト 100 に追加することにより、Content Engine を設定し、foo.com をドメインとしてもつ要求を照合します。この例では Websense URL filtering は既に設定済みであり、Content Engine で使用可能です。

```
ContentEngine (config)# rule action no-url-filtering pattern-list 100
ContentEngine (config)# rule pattern-list 100 domain .*foo.com
ContentEngine (config)# rule enable
```



(注) **no-url-filtering** アクションは、src-ip、dst-ip、dst-port、domain、group-name、groupname-regex、header-field、url-regex、および username をサポートしています。パターンは、グループタイプパターン（たとえば、**rule pattern-list 1 group-type and**）を使用して AND または OR することが可能です。デフォルトは OR です。

Content Engine が foo.com をドメインとしてもつ HTTP または HTTPS 要求を受信すると、**rule action no-url-filtering** ルールが照合されます。したがって、**debug http proxy** コマンドの下記の出力の一部に示されるように、Content Engine はこの特定の要求に対する URL filtering のバイパスを行います。

```
Oct 28 12:25:12 Content Engine 3: Rule action no-url-filtering match
- Bypassing urlfiltering
```

rule action no-url-filtering ルールが照合され Websense URL filtering の代わりに SmartFilter URL filtering が使用されている場合 **debug http proxy** コマンドの出力は次のようになります。

```
Oct 28 12:25:12 Content Engine 3: Rule action no-url-filtering match
- Bypassing SmartFilter processing
```

Content Engine が foo.com 以外をドメインとしてもつ HTTP または HTTPS 要求（www.abc.com をドメインとしてもつ要求）を受信すると、**rule action no-url-filtering** ルールは照合されません。したがって、**debug http proxy** コマンドの下記の出力の一部に示されるように、Content Engine はこの特定の要求に対する Websense URL filtering を続行します。

```
Oct 28 12:28:06 Content Engine 3: Rule action no-url-filtering not hit - Proceed with
urlfiltering
```

rule action no-url-filtering ルールが照合され Websense URL filtering の代わりに SmartFilter URL filtering が使用されている場合 **debug http proxy** コマンドの出力は次のようになります。

```
Oct 28 12:25:12 Content Engine 3: Rule action no-url-filtering not hit - Proceed with
SmartFilter processing
```

no-url-filtering アクションについての統計情報を表示するには **show statistics rule http action no-url-filtering EXEC** コマンドを入力します。

no-url-filtering アクションについての情報を表示するには **show run** および **show statistics rule all** の各コマンドを入力します。

use-server アクションの例

次に示すのは、**use-server** アクションの標準的な使用例です。

指定の基準に一致する HTTP 要求に対して、Content Engine が、オリジン サーバにコンタクトする必要がある場合（たとえば、キャッシュミスが起こった場合）に、その Content Engine は要求されたオブジェクトを検索するために要求に示されたサーバにアクセスするのではなく、別の宛先を使用します。これは主に、オンデマンド要求で使用され、通常はリバースプロキシの配置で使用されます。

この例では、Content Engine は www.abcbigcorp.com のリバースプロキシであり、そしてインターネットの残りに対してはプロキシです。この企業の Web サイト（www.abcbigcorp.com）の IP アドレスは、実際には Content Engine の IP アドレスであり、その企業の Web サイトのサーバの IP アドレスではありません。Content Engine が要求 http://www.abcbigcorp.com/main.html を受け取ると、通

常の処理として、www.abcbigcorp.com の IP アドレスが取得され、その IP アドレスに要求が送信されます。ただし、このケースでは、www.abcbigcorp.com の IP アドレスが Content Engine の IP アドレスであるため、管理者は、Content Engine が要求を自分自身に送信するのを防ぐ必要があります。

そのため、CE1 の管理者は、これらの要求（たとえば、キャッシュミス）を www.abcbigcorp.com の Web サーバに送信することを CE1 に指示するルールを次のように設定できます。

```
CE1(config)# rule use-server 1.2.3.4 80 domain www.abcbigcorp.com
```

この場合の 1.2.3.4 は、www.abcbigcorp.com の Web サーバの IP アドレスです。



(注) このルールは HTTP 処理にのみ適用されます。

no-proxy アクションの例

no-proxy アクションは、Content Engine の管理者が Content Engine 用の発信プロキシサーバをすでに設定しているときに適用できます。**no-proxy** アクションは、基準に一致した要求に対して、オリジンサーバとの接続が必要になった場合（たとえば、キャッシュミスが原因で）に、Content Engine 側で指定のプロキシサーバを使用してオリジンサーバとの接続を設定してはならないことを意味します。

このルールは、企業がすべてのインターネットコンテンツをキャッシングする Content Engine (CE1) をインターネットゲートウェイに設置していて、各支店に Content Engine (CE2、CE3、CE4) を設置している場合に役立ちます。このケースでは、管理者は、発信プロキシサーバとして CE4 を使用するように支店の CE2、CE3、CE4 を設定できますが、社内のコンテンツに対する要求向けに **no-proxy** ルールをセットアップできます。CE2、CE3、CE4 がクライアント要求を受信し、要求されたコンテンツがローカルキャッシュにまだ保存されていないときには、これらの Content Engine は要求を次のように処理します。

- インターネットコンテンツに対するクライアント要求の場合、CE2、CE3、CE4 は、オリジンサーバに直接にアクセスするのではなく、インターネットゲートウェイにある CE1 を使用する必要があります。
- 社内のコンテンツに対するクライアント要求の場合は、CE2、CE3、CE4 は、CE1 にアクセスするのではなく、オリジンサーバに直接に接続する必要があります。

rule action グローバル設定コマンドの使用方法についての詳細は、「[既存パターンリストへのアクションの関連付け](#)」(P. 13-27) を参照してください。サポートされているアクションとパターンの組み合わせのリストについては、[表 13-5](#) と [表 13-6](#) を参照してください。ACNS 5.3 またはそれ以降のソフトウェアを実行しているスタンドアロン Content Engine でプロトコルごとにサポートされるルールアクションのリストについては、[表 13-2](#) を参照してください。

サポートされているアクションとパラメータの組み合わせ

アクションによってはパターンが意味をもたない場合があるため、要求の照合に関して、すべてのアクションがすべてのパターンをサポートするとは限りません。[表 13-5](#) と [表 13-6](#) は、スタンドアロン Content Engine でサポートされているアクションとパターンの組み合わせのマトリックスです。

星印「*」は、ACNS 5.2.1 またはそれ以降のソフトウェアでアクションとパラメータの特定の組み合わせがサポートされることを示しています。

表 13-5 サポートされているアクションとパターンの組み合わせ：パート 1

アクション	パターン													
	domain	dst-ip	dst-port	header-f ield- referrer	header-f ield- req- line	header-f ield- user- agent	header-f ield- sub- referrer	header-f ield- sub- req- line	header-f ield- sub- user- agent	mime- type	src-ip	url- regex	url- regsub	groupname, username, groupname regex
allow	*	*	*	*	*	*					*	*		*
append- username header	*	*	*							*	*			*
block	*	*	*	*	*	*				*	*			*
cache- cookie	*	*							*		*			*
cache- non- cacheable	*	*							*		*			
cache- only	*	*							*		*			
dscp client	*	*	*						*	*	*			
dscp server	*	*	*							*	*			
freshness -factor	*	*	*						*	*	*			
insert- no-cache	*	*									*			
no-auth	*	*	*							*	*			
no-cache	*	*	*						*	*	*			

表 13-6 サポートされているアクションとパターンの組み合わせ：パート 2

アクション	パターン													
	domain	dst-ip	dst-port	header-f ield- referrer	header-f ield- req- line	header-f ield- user- agent	header-f ield- sub- referrer	header-f ield- sub- req- line	header-f ield- sub- user- agent	mime- type	src-ip	url- regex	url- regex	url- regsub
no-persistent-connection	*		*	*	*	*					*	*		
no-proxy	*	*	*								*	*		
no-url-filtering	*	*	*	*	*	*					*	*		*
redirect				*	*	*						*		
refresh	*	*	*						*	*	*	*		
reset	*	*	*	*	*	*					*	*		
rewrite							*	*	*				*	
selective-cache	*	*	*						*	*	*	*		
use-dns-server	*	*												
use-icap-service	*	*	*	*	*	*					*	*		
use-proxy	*	*	*								*	*		
use-proxy-failover	*	*	*								*	*		
use-server	*	*	*								*	*		
use-xforward-clt-ip	*		*	*	*	*					*	*		

Rules Template 処理上の考慮事項

スタンドアロン Content Engine に複数のルールを設定しているときには、ルールがある特定の順序で実行されます。

ACNS 5.x でのルールの処理順序を理解するには、ルール処理の次の側面を理解する必要があります。

- 定義済みのアクションの実行順序は事前に定められています。すなわち、同じアクションに関係するルールのグループは常に、別のアクションに関連する別のルールグループの前または後に実行されます。この順序は事前に定義されているため、ルールの入力順序の影響を受けません。
- 同じアクションのルールの中で、ルールパターンに事前に定義されている順序があります。つまり、同じアクション内では、同じパターンに関連するあるルールグループは常に、別のパターンの別のルールグループの前または後に実行されます。この場合も、この順序は事前に定義されているため、ルールの入力順序の影響を受けません。

同じアクションと同じルールパターンのすべてのルールでは、最後に入力されたものを最初に調べる（ルールの入力順序と逆の順序で調べる）という方法でルールが評価されます。たとえば、次のように指定したとします。

```
ContentEngine(config)# rule action use-proxy 1.2.3.4 abc.abc.com
ContentEngine(config)#rule action use-proxy 2.3.4.5 *.abc.com
```

Content Engine は、IP アドレス 2.3.4.5 をもつプロキシサーバに、abc.abc.com に対する要求を送ります。これは、**use-proxy 2.3.4.5 *.abc.com** ルールが最後に入力されたため、最初に評価され、要求がこのルールに一致するためです。

しかし、次のように入力すると、

```
ContentEngine(config)# rule action use-proxy 2.3.4.5 *.abc.com
ContentEngine(config)# rule action use-proxy 1.2.3.4 abc.abc.com
```

Content Engine は、IP アドレス 1.2.3.4 をもつプロキシサーバに、abc.abc.com に対する要求を送信します。

- ACNS ソフトウェアの最新リリースでルールが追加されると（たとえば、ACNS 5.2.1 ソフトウェアでは、**groupname**、**username**、**groupname-regex** のパターンと **cache-cookie** アクションが追加されています）、既存のアクションとパターンの処理順序が変わります。

ルールパターンの実行順序については、「[ルールパターンの実行順序](#)」(P. 13-19) を参照してください。この順序は、Content Engine GUI または CLI コマンドでルールを入力する順序の影響を受けることはありません。



ヒント

ACNS 5.x ソフトウェアでは、スタンドアロン Content Engine で **show rule EXEC** コマンドを入力すると、ルールがランダムに表示されます。しかし、**show statistics rule EXEC** コマンドを入力した場合には、ルールアクションの実行順序でルールが表示されます。そのため、このコマンドを使用して、Content Engine がユーザ定義ルールを処理する方法を調べることができます。

ルールアクションの実行順序

ACNS 5.2.3 またはそれ以降のソフトウェアでのルールアクションの実行順序は、次のとおりです。

1. Redirect-url-for-cdn（このルールアクションは、Content Distribution Manager に登録された Content Engine でのみサポートされ、スタンドアロン Content Engine ではサポートされません）
2. No-auth（RADIUS、LDAP、または NTLM を使用した認証の前）
3. Reset
4. Block
5. Redirect（キャッシュルックアップの前）
6. Rewrite（キャッシュルックアップの前）
7. No-url-filtering
8. Refresh（キャッシュヒットの場合はキャッシュルックアップの後）
9. Freshness-factor（キャッシュヒットの場合はキャッシュルックアップの後）
10. Use-server
11. No-proxy
12. Use-proxy-failover
13. Use-proxy
14. Use-dns-server
15. ToS/DSCP server（サーバとの接続時の ToS ビット）

16. ToS/DSCP client (サーバがクライアントに応答を送信するときに使用する接続時の ToS ビット)
17. DSCP client cache-miss
18. DSCP client cache-hit
19. Insert-no-cache
20. No-cache
21. Cache (サーバから応答が受信されたとき)
22. Selective-cache (サーバから応答が受信されたとき)
23. Append-username-header
24. Use-icap-service
25. Use-xforward-clt-ip
26. No-persistent-connection
27. Cache-cookie
28. No-selective-cache
29. Allow

reset、**block**、**rewrite**、**redirect** のルールアクションは、次に示す別のパターンをサポートします。すなわち、**request-line**、**referer**、**user-agent** の正規表現です。The **request-line** 正規表現は、要求の最初のラインを照合します。**user-agent** 正規表現は、要求の User-Agent ヘッダー値を照合します。**referer** 正規表現は、要求 Referer ヘッダー値を照合します。

次の例では、要求内の「internal.domain.com」という文字列をサーバ名「dummy」に置き換えるように Content Engine を設定します。

```
ContentEngine(config)# rule rewrite header-field referer internal.domain.com dummy
```

次の例では、置換パターンとして空の文字列を指定した場合、Referer ヘッダーが除去されます。ルールは、ABCBigCorp の社内サーバを示す referer ヘッダーを含んだ要求に対して、外部の Web サーバーから社内サーバの名前を確認できないように referer フィールドを除去することを意味します。これにより、ネットワーク セキュリティを強化できます。

```
ContentEngine(config)# rule rewrite header-field referer internal.abcbigcorp.com ""
```

Referer ヘッダーの除去は、**user-agent** パターンでも発生します。



(注) **rule action no-proxy**、**rule action use-proxy hostname port-number failover**、**rule action use-proxy** のコマンドは、**https proxy outgoing**、**http proxy outgoing**、**ftp proxy outgoing** のグローバル設定コマンドより優先されます。

use-server ルール、**no-proxy** ルール、**use-proxy** ルールの中では、**use-server** ルールが最初にチェックされます。いずれのルールも一致しなかった場合には、**no-proxy** ルールと **use-proxy** ルールが続けて実行されます (**no-proxy** ルールが一致した場合には、**use-proxy** ルールはチェックされません)。ルールが完全修飾ドメイン名 (FQDN) を使用して設定され、要求が透過モードでドメイン名の一部として受信された場合には、FQDN が要求 URL 内にないため、このルールは実行されません。透過モードでは、要求が特定のドメイン (ドメイン ルールが設定されているドメイン) に宛先を指定され、Host ヘッダーを含んでいない場合には、ルール パターンの照合に失敗します。**no-proxy** と **use-proxy** の両ルールが一致した場合には、**no-proxy** ルールが優先します。

use-proxy-failover ルールは **use-proxy** ルールに似ていますが、設定済みの発信プロキシでの接続に失敗した場合には、HTTP プロキシ発信設定で指定された発信プロキシに要求がフェールオーバーされます。ルール要求では、**http proxy outgoing origin-server** オプションが使用されます（設定されている場合）。**use-proxy-failover** ルールは、**use-proxy** ルールより優先します。**no-proxy** と **use-proxy-failover** の両ルールが一致した場合には、**no-proxy** ルールが優先します。送信先が除外リストに存在する場合は、HTTP フェールオーバーは適用されません。透過モードでは、オリジナルのプロキシの設定が優先します。



(注) Rules Template の設定は、**ip dscp** コマンドより優先されます。**url-filter** コマンドは、**rule** コマンドより優先されます。したがって、**rule no-block** コマンドが実行されるのは、**url-filter** コマンドが要求をブロックしなかった場合のみです。

ルール パターンの実行順序

実行順序は次のとおりです。

1. **Header-field**
2. **Header-field-sub**
3. その他のパターン：**url-regex**、**dst-port**、**src-ip**、**url-regex**、**domain**、**dst-ip**、**mime-type**。

これらのパターンで実行されるアクションに対しては実行順序は定められていません。



(注) MIME タイプは応答内にもみ存在するため、**freshness-factor**、**refresh**、**no-cache**、**selective-cache** の各アクションのみが MIME タイプのルールに適用されます。

たとえば、次の一連のアクションでは、**pattern-list 2 header-field** パターンが最初に実行され、次に **pattern-list 1 domain** パターンが実行されます。この順序が適用されるのは、ヘッダー情報が使用可能になった後にのみルールアクションが実行されるからです。

```
ContentEngine(config)# rule action block pattern-list 1
ContentEngine(config)# rule action block pattern-list 2
ContentEngine(config)# rule pattern-list 1 domain roti
ContentEngine(config)# rule pattern-list 2 header-field user-agent browser
```

次の例に示すパターンの AND 演算では、パターン エントリに基づいた実行順序は適用されません。

```
ContentEngine(config)# rule action block pattern 3
ContentEngine(config)# rule pattern-list 3 dst-port 80
ContentEngine(config)# rule pattern-list 3 header-field user-agent browser
```

上記の例では、送信先ポート (**dst-port**) が最初にチェックされ、次に **header-field** がチェックされます。

実行順序は次のようになります（ルールは AND 演算されません）。

1. **Header-field**
2. **Header-field-sub**
3. その他のパターン：**url-regex**、**dst-port**、**src-ip**、**url-regex**、**domain**、**dst-ip**、**mime-type**。

これらが実行されるアクションに基づいた実行順序はありません。

ある特定のルールに通常のパターン タイプのヘッダー フィールドが含まれている場合は、特別な実行順序はありません。

残りのパターン リストに一致するルールの検索は、一致していることがすでにわかっている場合は実行されません。たとえば、**rule action block** アクション コマンドの一致が **URL-regex** 要求に見つかった場合、残りのパターンの **domain**、**dst-ip**、**MIME-type** は検索されません。すべてのパターンが **rewrite** と **redirect** のアクションに適用できるわけではありません。

ルールは相互に OR 演算されます。複数のルールすべてが 1 つの要求に一致する場合があります。その場合、すべてのアクションが実行されますが、「[ルールアクションの実行順序](#)」(P. 13-17) に定義されているルールアクションの実行順序に基づいて、競合するアクション間で優先順位が設定されます。**rule action** グローバル設定コマンドには、**rule pattern-list** グローバル設定コマンドで設定する複数のパターンを含めることができます。

特定のルールを適用対象外にできます。たとえば、**domain** パターンを含むルールを適用対象外にするには、ブラウザの中にドメイン名ではなく Web サーバの IP アドレスを入力します。ルールが予期せぬ影響を受ける場合があります。たとえば、「**ibm**」として指定された **domain** パターンをもつルールは「**www.ibm.com**」と一致することを意図したのですが、**www.ribman.com** のようなドメイン名とも一致する場合があります。



(注)

src-ip ルールは、Content Engine が別のプロキシまたは別の Content Engine から受信する要求に対して、意図した通りに適用されない場合があります。これは、元のクライアント IP アドレスが X-Forwarded-For ヘッダー内に含まれているためです。すなわち、元の要求の送信元 IP アドレスが、オリジン サーバに向かう途中で、送信 Content Engine の IP アドレスに透過的に置き換えられています。

1 つのルールパターンが一致すると、残りのパターンは検索されません。サーバがすでにオブジェクトにキャッシュ不能 (**noncacheable**) のマークを付けている場合、サーバ側では、このオブジェクトがキャッシングされないことがわかっているため、**no-cache** ルールはチェックされません。**no-cache** ルールのチェックはすべて、キャッシング可能な要求に対してのみ実行されます。

Rules Template の設定

スタンドアロン Content Engine でルールを設定するときには、次の点を念頭に置くことが重要です。

- アクションの数は無制限です。
- パターンリストの最大数は 512 です。
- アクション当たりのパターンの最大数は 128 です。
- パターンタイプ当たりにパターンリストに設定できるパターンの最大数は 128 です。
- Content Engine CLI からルールの正規表現に疑問符 (?) を入力するには、エスケープ文字を使用し、その後に疑問符 (?) を入力します。これにより、状況依存ヘルプが CLI に表示されなくなります。
- Rules Template の設定は、**ip dscp** コマンドより優先されます。**url-filter** コマンドは、**rule** コマンドより優先されます。そのため、**rule no-block** コマンドが実行されるのは、**url-filter** コマンドが要求をブロックしなかった場合のみです。
- グループ名とユーザ名に基づいたルールによる許可は、HTTP 要求認証とグループに基づいたアクセス制御リスト (ACL) 許可が行われた後でのみ起こります。Rules Template 内の設定と ACL が一致した場合は、ACL のほうが優先されます。
- オブジェクトが認証されている場合には、**rule action cache-non-cacheable** コマンドはそのオブジェクトをキャッシングできません。すなわち、認証済みのオブジェクトに関しては、オリジンサーバの中には Last-Modified、または ETag エンティティ ヘッダーを送信しないものがあります。そのため、これらの許可されたオブジェクトは Content Engine で再び有効にされません。認証されたオブジェクトは、オリジンサーバからのみ使用可能となります。認証されたオブジェクトに対して、サーバが Last-Modified および ETag ヘッダーを送信すると、それらのオブジェクトが正しく再び有効にされ、キャッシュから取り出されます。
- 次のような状況では、**no-auth** ルールを適用すると、複数の認証ウィンドウが表示されます。
 - **no-auth** ルールを使用して、メイン ウィンドウ (たとえば、index.htm) がプロキシ認証から除外される場合
 - ユーザ入力情報が Content Engine 認証キャッシュにまだ入っている場合
 - index.htm ウィンドウに別のドメインに属するオブジェクトが含まれている場合

複数の認証ウィンドウが表示されないようにするには、グローバル設定モードで **http avoid-multiple-auth-prompts** 非表示コマンドを入力します。次の例に示すように、**show http avoid-multiple-auth-prompts EXEC** 非表示コマンドを使用して設定をチェックしてください。

```
ContentEngine# show http avoid-multiple-auth-prompts
Avoiding multiple authentication prompts due to no-auth rules is enabled
```

この例に示すコマンドは、この特定の状況にのみ適用されるため、非表示になります。



(注) Rules Template は、CLI または Content Engine GUI から設定できます (図 13-1 を参照)。

Content Engine のルール処理の設定方法については、以下の各項を参照してください。

- [ルール処理の有効化 \(P. 13-21\)](#)
- [スタンドアロン Content Engine のためのルールの設定 \(P. 13-22\)](#)

ルール処理の有効化

デフォルトでは、Content Engine 上のルール処理は無効になっています。スタンドアロン Content Engine でのルール処理を有効にするには、**rule** グローバル設定コマンドを次のように使用します。

```
ContentEngine(config)# rule enable
```

スタンドアロン Content Engine のためのルールの設定

スタンドアロン Content Engine で HTTP、HTTPS、MMS、RTSP のトラフィックをフィルタリングするルールを設定するには、**rule** グローバル設定コマンドを使用します。

```
rule {action action-type pattern-list list_num [protocol {all | protocol-type}]} enable | pattern-list list_num pattern-type}
```

表 13-7 では、**rule** コマンドのパラメータを説明しています。ルール コマンド パラメータがアクションタイプやパターンタイプの場合は、表 13-7 に明記しています。



(注) ほとんどのアクションがパラメータをとりません。例外は、**use-server**、**freshness-factor**、**use-proxy** のアクションです。

表 13-7 rule CLI コマンドのパラメータ

パラメータ	アクションまたはパターンタイプ	説明
action		着信要求が指定のパターンに一致した場合に、このルールで実行されるアクションを設定します。
<i>action-type</i>		着信要求が指定のパターンに一致した場合に実行するアクションのタイプ（たとえば、 allow ）を指定します。
allow	アクションタイプ	要求を許可します。
pattern-list		パターンリストを設定します。
<i>list_num</i>		パターンリスト番号（1～512）。
protocol		この規則が照合されるプロトコル。
all		このルールとこのアクションに適用可能なすべてのプロトコルを照合します。
<i>protocol-type</i>		このルールの照合対象のプロトコルタイプ（着信トラフィックのタイプ）を指定します。
enable		ルール処理を有効にします。
http		着信 HTTP トラフィックとこのルールを照合します。
https		このルールと着信の HTTPS トラフィックを照合します。
mms		着信 MMS トラフィックとこのルールを照合します。
rtsp		着信 RTSP トラフィックとこのルールを照合します。
append-username-header	アクションタイプ	要求ヘッダーにユーザ名を追加します。
block	アクションタイプ	要求をブロックします。
cache-cookie	アクションタイプ	クッキーを含んだ要求をキャッシングします。
ttl		このオブジェクトの Time To Live (TTL; 存続可能時間) 値。
days		Time To Live 単位 (日数)。
<i>days</i>		Time To Live 日数の値 (1～1825)。
hours		Time To Live 単位 (時間数)。
<i>hours</i>		Time To Live 時間数の値 (1～43800)。
minutes		Time To Live 単位 (分数)。
<i>minutes</i>		Time To Live 分数の値 (1～2628000)。
seconds		Time To Live 単位 (秒数)。

表 13-7 rule CLI コマンドのパラメータ (続き)

パラメータ	アクションまたはパターンタイプ	説明
<i>seconds</i>		Time To Live 秒数の値 (1 ~ 157680000)。
groupname		文字列とグループ名 (たとえば、「Engineering」) を照合します。このグループ名に基づいたルール ポリシーは、LDAP や NTLM を通して認証されたユーザに対する要求認証にのみ適用できます。
<i>groupname</i>		グループ名の文字列。
username		文字列と指定のユーザ名を照合します。このユーザ名に基づいたルール ポリシーは、認証用のユーザ名を含んだすべての要求認証メソッド (たとえば、LDAP、NTLM、RADIUS、TACACS+) に適用できます。
<i>username</i>		ユーザ名の文字列 (たとえば、「jdoe8」)。
groupname-regex		正規表現とグループ名を照合します。
<i>groupname-regex</i>		グループ名と照合する正規表現。
cache-non-cacheable	アクションタイプ	HTTP 応答ヘッダーをオーバーライドして、このオブジェクトをキャッシングします。
ttl		このオブジェクトの Time To Live (TTL ; 存続可能時間) 値。
days		Time To Live 単位 (日数)。
<i>days</i>		Time To Live 日数の値 (1 ~ 1825)。
hours		Time To Live 単位 (時間数)。
<i>hours</i>		Time To Live 時間数の値 (1 ~ 43800)。
minutes		Time To Live 単位 (分数)。
<i>minutes</i>		Time To Live 分数の値 (1 ~ 2628000)。
seconds		Time To Live 単位 (秒数)。
<i>seconds</i>		Time To Live 秒数の値 (1 ~ 157680000)。
cache-only	アクションタイプ	このオブジェクトのみをキャッシングします。
dscp client	アクションタイプ	クライアントの IP の ToS/DSCP (Type of Service/Differentiated Services Code Point) フィールド応答を設定します。
cache-hit		キャッシュ ヒット時に応答をクライアントに送信します。
match-server		サーバの元の ToS または DSCP 値を使用します。
set-dscp		DSCP (Differentiated Services Code Point) 値を設定します。DSCP 値のリストについては、表 11-1 を参照してください。
set-tos		ToS (Type of Service) 値を設定します。ToS 値のリストについては、表 11-2 を参照してください。
cache-miss		キャッシュ ミス発生時に応答をクライアントに送信します。
dscp server	アクションタイプ	発信応答用の ToS または DSCP サービス コード ポイントを設定します。
match-client		クライアントの元の ToS または DSCP 値を使用します。
enable		ルール処理を有効にします。
freshness-factor	アクションタイプ	ヒューリスティック モディファイアをキャッシングします。
<i>exp_time</i>		オブジェクトの存在期限に対する満了時間のパーセント値 (0 ~ 100)。
insert-no-cache	アクションタイプ	no-cache ヘッダーを応答に挿入します。
no-auth		認証を行いません。
no-cache	アクションタイプ	オブジェクトをキャッシングしません。

表 13-7 rule CLI コマンドのパラメータ (続き)

パラメータ	アクションまたはパターンタイプ	説明
no-persistent-connection		永続接続を使用できないようにします。
all		クライアントまたはサーバとの永続接続を使用できないようにします。
client-only		クライアントとの永続接続を使用できないようにします。
server-only		サーバとの永続接続を使用できないようにします。
no-proxy	アクションタイプ	アップストリーム プロキシを使用しません。このルールアクションの使い方の例については、「 no-proxy アクションの例 」(P. 13-14)を参照してください。
no-url-filtering	アクションタイプ	特定の HTTP および HTTPS 要求について URL フィルタリングをバイパスします。この機能は、ローカルリスト URL フィルタリングだけでなく、Websense、SmartFilter、または ACNS 5.2.3 またはそれ以降のソフトウェアの N2H2 URL フィルタリングをサポートしています。このルールアクションの使い方の例については、「 no-url-filtering アクションの例 」(P. 13-12)を参照してください。
redirect	アクションタイプ	書き換えられた URL に要求をリダイレクトします。
<i>url</i>		リダイレクト URL。
redirect-url-for-cdn	アクションタイプ	ACNS ネットワーク コンテンツに対する要求を代替の URL にリダイレクトします。このルールアクションは、Content Distribution Manager に登録された Content Engine でのみサポートされます。つまり、スタンドアロン Content Engine ではサポートされません。
refresh	アクションタイプ	オブジェクトを Web サーバで再検証します。
reset	アクションタイプ	TCP RST を発行します。
rewrite	アクションタイプ	オリジナルの要求を指定の URL に書き換えて、その URL をキャッシュミス時にフェッチします。
selective-cache	アクションタイプ	HTTP が許可したオブジェクトをキャッシングします。
use-dns-server	アクションタイプ	特定の DNS サーバを使用します。
<i>hostname</i>		DNS サーバのホスト名。
<i>ip-address</i>		DNS サーバの IP アドレス。
use-icap-service	アクションタイプ	特定の ICAP サーバを使用します。
icap-service-name	パターンタイプ	ICAP サービス名を使用します。
<i>service name</i>		ICAP サービスの名前。
use-proxy	アクションタイプ	特定のアップストリーム プロキシを使用します。
<i>hostname</i>		特定のプロキシのホスト名。
<i>ip-address</i>		特定のプロキシの IP アドレス。
<i>port</i>		特定のプロキシのポート番号 (1 ~ 65535)。
use-proxy-failover	アクションタイプ	発信プロキシを指定の発信 HTTP プロキシサーバにフェールオーバーさせます。
<i>hostname</i>		特定のプロキシのホスト名。
<i>ip-address</i>		特定のプロキシの IP アドレス。
<i>port</i>		特定のプロキシのポート番号 (1 ~ 65535)。
use-server	アクションタイプ	特定のサーバを使用します。このルールアクションの使い方の例については、「 use-server アクションの例 」(P. 13-13)を参照してください。
use-xforward-clt-ip		フィルタリング用として、x-forwarded ヘッダー内のクライアント IP アドレスを使用します。
<i>hostname</i>		特定のサーバのホスト名。

表 13-7 rule CLI コマンドのパラメータ (続き)

パラメータ	アクションまたはパターンタイプ	説明
<i>ip-address</i>		特定のサーバの IP アドレス。
<i>port</i>		特定のサーバのポート番号 (1 ~ 65535)。
domain	パターンタイプ	ドメイン名に一致する正規表現。
<i>dn_regexp</i>		ドメイン名と照合する正規表現。
dst-ip	パターンタイプ	要求の送信先 IP アドレス。
<i>d_ipaddress</i>		要求の送信先 IP アドレス。
<i>d_subnet</i>		送信先 IP サブネットマスク。
dst-port	パターンタイプ	送信先ポート番号。
<i>port</i>		宛先ポート番号 (1 ~ 65535)。
group-type	パターンタイプ	パターンリストを AND と OR のどちらのタイプにするかを指定します。デフォルトは OR です。
and		パターンリストに AND パターンを指定します。
or		パターンリストに OR パターンを指定します。
header-field	パターンタイプ	要求ヘッダーフィールドパターン。
referer		参照元 (referer) 要求ヘッダー。
<i>ref_regexp</i>		参照元要求ヘッダーと照合する正規表現。
request-line		要求メソッドライン。
<i>req_regexp</i>		要求メソッドラインと照合する正規表現。
user-agent		ユーザ エージェント要求ヘッダー。
<i>ua_regexp</i>		ユーザ エージェント要求ヘッダーと照合する正規表現。
header-field-sub	パターンタイプ	要求ヘッダーフィールドパターンを要求し、置換パターンと代替する。
referer		参照元 (referer) 要求ヘッダー。
<i>ref_regexp</i>		参照元要求ヘッダーと照合する正規表現。
<i>ref_sub</i>		要求ヘッダーの正規表現の置換文字列。
request-line		要求メソッドライン。
<i>req_regexp</i>		要求メソッドラインと照合する正規表現。
<i>req_sub</i>		要求メソッドラインの正規表現の置換文字列。
user-agent		ユーザ エージェント (User Agent) 要求ヘッダー。
<i>ua_regexp</i>		ユーザ エージェント要求ヘッダーと照合する正規表現。
<i>ua_sub</i>		ユーザ エージェント要求ヘッダーの正規表現の置換文字列。
mime-type	パターンタイプ	Content-Type (コンテンツタイプ) HTTP ヘッダーと照合する MIME タイプ。
<i>mt_regexp</i>		コンテンツタイプと照合する正規表現。
src-ip	パターンタイプ	要求の送信元 IP アドレスを使用するように設定します。
<i>s_ipaddress</i>		要求の送信元 IP アドレス。
<i>s_subnet</i>		送信元 IP サブネットマスク。
url-regex	パターンタイプ	URL のサブストリングと照合する正規表現を使用するように設定します。
<i>url_regexp</i>		URL 文字列と照合する正規表現。
url-regsub	パターンタイプ	URL および置換パターンと照合する正規表現を設定します。
<i>url_regexp</i>		URL 文字列と照合する正規表現。
<i>url_sub</i>		URL 文字列置換パターン。

ACNS 5.x ソフトウェアでは、一致した URL、ファイルタイプ、ドメイン、送信先 IP アドレス、送信元 IP アドレス、または送信先ポートに基づいて、IP パケット内に ToS 値または DSCP 値を設定できます。個々の ToS 値または DSCP 値を次の項目に設定できます。

- Content Engine からサーバへの要求
- キャッシュ ヒット時のクライアントへの応答
- キャッシュ ミス時のクライアントへの応答

ToS または DSCP の設定は、**src-ip s_ipaddress s_subnet**、**dst-ip d_ipaddress d_subnet**、**dst-port port**、**domain LINE**、**url-regex LINE**、または **mime-type LINE** のオプションと一致するポリシーのいずれかに基づいて行います。**ip dscp** コマンドを使用して、ToS または DSCP のグローバル設定値を指定することもできます。

パターン リストの設定

スタンドアロン Content Engine でパターン リストを作成するには、**rule pattern-list** グローバル設定コマンドを次のように使用します。

```
ContentEngine(config)# rule pattern-list list_num
```

ここで各パラメータの意味は、次のとおりです。

list_num はパターン リスト番号 (1 ~ 152) です。

たとえば、次のようにしてパターン リスト 10 を作成します。

```
ContentEngine(config)# rule pattern-list 10
```

既存のパターン リストへのパターンの追加

スタンドアロン Content Engine の既存のパターン リストにパターンを新規に追加する手順は、次のとおりです。

ステップ 1 既存のパターン リストへパターンを新規に追加します。

```
ContentEngine(config)# rule pattern-list list_num pattern type pattern value
```

次の例では、パターン リスト 10 にパターンを追加する方法を示しています。このパターンでは、**dst-ip** (送信先 IP アドレス) のパターンタイプを使用して、送信先 IP アドレス 172.16.25.25 に定義するアクションが実行されます。

```
ContentEngine(config)# rule pattern-list 10 dst-ip 172.16.25.25 255.255.255.0
```



(注) サポートされているパターンタイプの詳細リストについては、[表 13-3](#) を参照してください。

ステップ 2 指定のパターン リストに新しいパターンが追加されているか確認してください。

次の例では、[ステップ 1](#) で作成したパターンがパターン リスト 10 に追加されているか検証する方法を示しています。

```
ContentEngine# show rule pattern-list 10 all
Rules Template Configuration
-----
Rule Processing Enabled

Pattern-Lists :

rule pattern-list 10 dst-ip 172.16.25.25 255.255.255.0
```

アクションをパターン リストに関連付ける方法については、次の「[既存パターン リストへのアクションの関連付け](#)」を参照してください。

既存パターン リストへのアクションの関連付け

スタンドアロン Content Engine の既存のパターン リストにアクションを関連付ける手順は、次のとおりです。

ステップ1 既存パターン リストにアクションを関連付けます。

```
ContentEngine(config)# rule action action-type pattern-list list_num
protocol {protocol-type | all}
```

アクションを特定のプロトコルまたは複数のプロトコルに適用できます。プロトコルが設定されていない場合には、Content Engine を通過するすべてのトラフィックに対して指定のアクションが実行されます。次の例では、すべてのプロトコルに対して、パターン リスト 10 に **block** アクションが関連付けられます。

```
ContentEngine(config)# rule action block pattern-list 10 protocol all
```



(注) **block** アクションと同様に、ほとんどのアクションはパラメータをとりません。rule action グローバル設定コマンドのパラメータの詳細については、[表 13-7](#)を参照してください。

ステップ2 指定のパターン リストに新しいアクションが関連付けられているか確認します。

```
ContentEngine# show rule action action-type protocol {protocol-type | all}
```

次の例では、**block** アクションがパターン リスト 10 に関連付けられているか確認する方法を示しています。

```
ContentEngine# show rule action block
Rules Template Configuration
-----
Rule Processing Enabled

Actions :

rule action block pattern-list 10 protocol all
ContentEngine#
```

パターン リストに基づいて実行されるアクションの検証

指定のパターン リストに基づいて特定のアクションが実行されるかを確認するには、アクションの実行後に、ローカルの Rules Template 設定統計情報を表示します。

```
ContentEngine# show statistics rule action action-type
```

次の例では、**rule action block** コマンドを設定して、既存のパターン リストに関連付けます。ここでは、パターンとして yahoo.com ドメインがリスト表示されます。

```
ContentEngine(config)# rule action block pattern-list 10 protocol all
ContentEngine# show statistics rule action block
Rules Template Statistics
-----
Rule hit count = 3   Rule: rule action block pattern-list 10 protocol all
ContentEngine#
```

この例では、統計情報 (Rule hit count) として、yahoo.com への要求が 3 回拒否されたことが示されています。

スタンドアロン Content Engine でのルール設定例

ここでは、スタンドアロン Content Engine でルール設定例を示します。



(注)

以降の例では、特に注記がない限り、すべてのアクションとパターンがすべてのプロトコルに適用されることを前提としています。

- **domain** パターン タイプを使用して .foo.com を含んだドメインを指定して、このパターンをパターン リスト 12 に追加します。

```
ContentEngine(config)# rule pattern-list 12 domain \.foo.com
```

block アクションをパターン リスト 12 に関連付けて、.foo.com を含んだドメインに対する URL 要求をすべて拒否するように Content Engine を設定します。

```
ContentEngine(config)# rule action block pattern-list 12
```

同じパターン リスト (パターン リスト 12) に複数のパターンを設定します。いずれかのパターンが着信要求と一致した場合、該当のアクションがとられます。次の例では、**rule action block** グローバル設定コマンド (アクション) によって、パターン リスト 12 に指定されているすべてのパターンを拒否します。

```
ContentEngine(config)# rule pattern-list 12 domain \.foo.com
ContentEngine(config)# rule pattern-list 12 dst-ip 172.16.25.25 255.255.255.0
ContentEngine(config)# rule action block pattern-list 12
```

- ***cgi-bin*** という文字列を含んだ URL 要求をキャッシュしないように Content Engine を設定します。

```
ContentEngine(config)# rule pattern-list 13 url-regex \.*cgi-bin.*
ContentEngine(config)# rule action no-cache pattern-list 13
```

- **rule action** グローバル設定コマンドの前に **no** を使用して、ルールを削除します。

```
ContentEngine(config)# no rule use-proxy foo.com 8080 pattern-list 13
ContentEngine(config)# no rule action block pattern-list 2
```

- MIME-type イメージの Content Engine 新鮮度係数を設定します。


```
ContentEngine(config)# rule pattern-list 13 mime-type image/.*
```

```
ContentEngine(config)# rule action freshness-factor 75 pattern-list 13
```
- 送信先 IP アドレス 10.1.1.1 への発信要求用として、Content Engine の ToS 値を最小遅延値に設定します。


```
ContentEngine(config)# rule action dscp server set-tos min-delay protocol all
```

```
ContentEngine(config)# rule pattern-list 2 dst-ip 10.1.1.1 255.255.255.255
```
- すべての発信要求用として、Content Engine の ToS 値を最小遅延値に設定します。


```
ContentEngine(config)# ip dscp server set-tos min-delay
```
- 同じオブジェクトに対する今後のすべてのキャッシュ ヒット応答に対して、(オブジェクトが最初にフェッチされたときに) サーバが当初送信した ToS、または DSCP 値を使用するように、Content Engine を設定します。


```
ContentEngine(config)# ip dscp client cache-hit match-server
```

```
ContentEngine(config)# rule action no-cache pattern-list 3 protocol all
```

```
ContentEngine(config)# rule pattern-list 3 url-regex \.*cgi-bin.*
```

```
ContentEngine(config)# rule pattern-list 4 dst-ip 172.31.120.0 255.255.192.0
```
- new-domain-name に変更された old-domain-name に対する要求を新しいドメイン名にリダイレクトするように Content Engine を設定します。


```
ContentEngine(config)# rule action redirect http://old-domain-name/
```

```
ContentEngine(config)# rule pattern-list 1 protocol http
```

```
ContentEngine(config)# rule pattern-list 1 url-regex
```

```
http://old-domain-name/http://new-domain-name/
```
- IETF サイトからの要求をローカルにミラーリングされるサイトにリダイレクトするように Content Engine を設定します。


```
ContentEngine(config)# rule action redirect http://www.ietf.org/rfc/(.*)
```

```
ContentEngine(config)# rule pattern-list 2 protocol http
```
- 次の例では、要求 URL が http://www.ietf.org/rfc/rfc1111.txt である場合、Content Engine は、その URL を http://wwwin-eng.cisco.com/RFC/RFC/rfc1111.txt として書き換え、Location ヘッダー内に書き換えられた URL をもつ 302 Temporary Redirect 応答をクライアントに送ります。ブラウザは、書き換えられた URL に対する要求を自動的に開始します。


```
ContentEngine(config)# rule pattern-list 2 url-regex
```

```
http://www.ietf.org/rfc/(.*) http://wwwin-eng.cisco.com/RFC/RFC/\1
```
- linux.org に対するすべての要求を、インドにある Content Engine の場所に近いローカルサーバにリダイレクトするように Content Engine を設定します。


```
ContentEngine(config)# rule action redirect http://linux.org/(.*) pattern-list 3
```

```
ContentEngine(config)# rule protocol http
```
- rule action no-auth** グローバル設定コマンドは、LDAP、RADIUS、SSH、TACACS+ などの認証と許可の機能をバイパスする特定のログイン要求とコンテンツ要求を許可します。次の例では、送信元 IP アドレス (src-ip) 172.16.53.88 からの要求はどれも認証されません。


```
ContentEngine(config)# rule enable
```

```
ContentEngine(config)# rule action no-auth pattern-list 1 protocol all
```

```
ContentEngine(config)# rule pattern-list 1 src-ip 172.16.53.88 255.255.255.255
```

- 認証と SmartFilter URL フィルタリング用に ACNS 5.x ソフトウェアを設定している場合には、認証をバイパスすることが許可されている要求は、SmartFilter URL フィルタもバイパスします。172.22.73.34 の送信先 IP アドレス (dst-ip) に対するどの要求も認証しないように Content Engine を設定します。

```
ContentEngine(config)# rule action no-auth pattern-list 2 protocol all
ContentEngine(config)# rule pattern-list 2 dst-ip 172.22.73.34 255.255.255.255
```

- 9090 の送信先 IP ポート (dst-port) に対するどの要求も認証しないように Content Engine を設定します。

```
ContentEngine(config)# rule action no-auth pattern-list 3 protocol all
ContentEngine(config)# rule pattern-list 3 dst-port 9090
```

- cgi-bin という文字列を含んだ URL 要求を認証しないように Content Engine を設定します。

```
ContentEngine(config)# rule action no-auth pattern-list 4 protocol all
ContentEngine(config)# rule pattern-list 4 url-regex .*cgi-bin.*
```

- ドメインとして cisco.com を含んだどの要求も認証しないように Content Engine を設定します。たとえば、roti.cisco.com や badal.cisco.com に対する要求が Content Engine 認証から除外されます。

```
ContentEngine(config)# rule action no-auth pattern-list 5 protocol all
ContentEngine(config)# rule pattern-list 5 domain cisco.com
```

設定済みルールの統計情報の表示

スタンドアロン Content Engine に設定済みルールの統計情報を表示するには、**show statistics rule EXEC** コマンドを使用します。ACNS 5.3.1 またはそれ以降のソフトウェア リリースでは、**show statistics rule** コマンドのオプションが変更され RTSP および WMT RTSP ルールの統計情報を表示することが可能になりました。

ACNS 5.2.x およびそれ以前のソフトウェアでは、**show statistics rule** コマンドのコマンド オプションは次のとおりでした。

```
ContentEngine# show statistics rule ?
  all   Display statistics of all the Rules
  http  Display statistics of http/https/all Rules
  wmt   Display statistics of wmt Rules
```

ACNS 5.3.1 およびそれ以降のソフトウェアでは、**show statistics rule** コマンドのコマンド オプションは次のとおりです。

```
ContentEngine# show statistics rule ?
  all   Display statistics of all the Rules
  http  Display statistics of http/https/wmt-http Rules
  mms   Display statistics of mms Rules
  rtsp  Display statistics of rtsp/wmt-rtsp Rules
```

ACNS 5.3.1 ソフトウェア リリースでは、**wmt** オプションは **mms** および **rtsp** オプションに置き換えられました。

たとえば、**show statistics rule rtsp** コマンドを入力して、RTSP ルール (RealMedia Player からの RTSP 要求のルール [RTSP rule] および Windows Media 9 Player からの RTSP 要求のルール [WMT-RTSP rule]) を表示します。

ルール ヒット カウント総数を表示するには、**show statistics rule all EXEC** コマンドを入力します。

```
ContentEngine# show statistics rule all
Rules Template Statistics
-----
Rule hit count = 0   Rule: rule action no-auth pattern-list 1 protocol all
Rule hit count = 0   Rule: rule action rewrite pattern-list 2 protocol all
```

HTTP 要求と HTTPS 要求に適用されるすべての設定済みルールに関する統計情報を表示するには、**show statistics rule http EXEC** コマンドを入力します。

特定の設定済みルールに関する統計情報を表示するには、**show statistics rule http action rule action name** を入力します。たとえば、**no-url-filtering rule action** についての統計情報を表示するには **show statistics rule http action no-url-filtering EXEC** コマンドを入力します。

設定済みルールの統計情報のクリア

スタンドアロン Content Engine で設定されたルールの統計情報をクリアするには、ACNS 5.3.1 ソフトウェア リリースの **clear statistics rule EXEC** コマンドを使用します。**rtsp** オプションが **clear statistics rule** コマンドに追加され、設定済み RTSP および WMT RTSP ルールの統計情報が表示できるようになりました。

ACNS 5.2.x およびそれ以前のソフトウェアでは、**clear statistics rule** コマンドのコマンド オプションは次のとおりでした。

```
ContentEngine# clear statistics rule ?
  action  Clear statistics of all the rules with same action
  all     Clear statistics of all the rules
```

ACNS 5.3.1 およびそれ以降のソフトウェアでは、**clear statistics rule** コマンドのコマンド オプションは次のとおりです。

```
ContentEngine# clear statistics rule ?
  action  Clear statistics of all the rules with same action
  all     Clear statistics of all the rules
  rtsp    Clear statistics of rtsp/wmt-rtsp rules
```

たとえば、**clear statistics rule rtsp** コマンドを使用して、設定済み RTSP ルール (RealMedia Player からの RTSP 要求のルール [RTSP rule] および Windows Media 9 Player からの RTSP 要求のルール [WMT-RTSP rule]) をクリアします。