CHAPTER 11

コンテンツ ルールへの スティッキ パラメータの設定

この章では、コンテンツ ルールへのスティッキ パラメータの設定方法について説明します。この章の内容は、特に指定のない限り、すべての CSS モデルに共通です。この章の主な内容は次のとおりです。

- 「Sticky-No-Cookie-Found-Action の設定」
- 「電子商取引アプリケーションと他のインターネット アプリケーション用のスティッキ パラメータの設定」
- 「スティッキ アトリビュートの表示」
- 「スティッキ テーブル設定の表示」
- 「スティッキ接続の統計情報の表示」

スティッキの概要

セッション中、CSS はクライアントとサーバの関連付けを維持します。この関連付けはスティッキ性と呼ばれます。セッション中クライアントが同じサーバに留まる必要がある場合に、このスティッキ性によって Web を介したトランザクションが可能になります。CSS は、使用するロード バランシング方式を決めた後に、コンテンツ ルールに応じて、クライアントを適切なサーバに「固定（スティック）」します。

CSS は、クライアントがすでに特定のサービスに固定されていると判断すると、一致したコンテンツ ルールで指定されたロード バランシングの基準に関係なく、クライアント要求をそのサービスに送信します。クライアントが特定のサービスに固定されていないと判断すると、CSS はコンテンツ要求に通常のロード バランシングを適用します。

クライアントのクッキーは、コンテンツを提供するサービスのクライアントを一意に識別します。クッキーは、Web クライアントにデータを配信したり、クライアントに情報の保存を要求したりするために、サーバが使用する小さなデータ構造体です。一部のアプリケーションでは、クライアントとサーバ間の状態を維持するために、クライアントがサーバに情報を返すことがあります。

CSS はコンテンツへの要求を検査し、コンテンツ ルールの照合によってコンテンツが固定していると判断すると、そのコンテンツ要求に存在するすべてのクッキーと URL を検査します。この情報を使用して、適切なサーバにコンテンツ要求を送ります。

CSS スティッキ テーブルのエントリの総数は、CPU メモリのサイズによって異なります。

- CSS 11501 は、128K のスティッキ テーブル（288MB の CPU メモリ）をサポートします。
- CSS 11503 および 11506 は、System Control Module（SCM; システム コントロール モジュール）に搭載されているメモリが 288MB か 144MB かに応じて、128K または 32K のスティッキ テーブルをサポートします。つまり CSS は、メモリが 288MB の場合 128K のスティッキ テーブルをサポートし、メモリが 144MB の場合、32K のスティッキ テーブルをサポートします。

スティッキ テーブルのサイズ（128K または 32K）は、128,000 または 32,000 のユーザが同時に同じサイトを訪れると、そのテーブルが一巡し、最初のユーザが「非固定」になることを意味します。

ここでは、スティッキ性とその用途について説明します。

スティッキを使用する理由

顧客は、電子商取引サイトを訪れると通常、そのサイト内の各所を見て回る、インターネットのウィンドウ ショッピングから始めます。アプリケーションによっては、一度サーバとの接続が確立されると、顧客をその 1 つのサーバに「固定」することを要求するサイトもあれば、顧客がショッピング カートを作るまでは、その要求をしないものもあります。

いずれの場合も、顧客がショッピング カートに商品を入れ始めたら、同じサーバの同じショッピング カートにすべての商品が入るように、顧客のすべての要求が同じサーバに送信されることが重要です。顧客のショッピング カートのインスタンスは、通常、特定の Web サーバに対してローカルであり、複数のサーバ間で複製されません。

電子商取引アプリケーションの他にも、スティッキ性が必要なアプリケーションがあります。銀行業務用のアプリケーションやオンライン トレーディングなどの、クライアント情報を管理するあらゆる Web アプリケーションに、スティッキ性は必要です。

アプリケーション側でユーザやユーザ グループを識別する必要があるため、CSS は特定のユーザを特定の Web サーバに固定する方法を決める必要があります。CSS では、次のようなさまざまな方法を使用できます。

- 送信元 IP アドレス
- 送信元 IP アドレスおよび宛先ポート
- クッキーまたは URL にある文字列
- SSL セッション ID

上記のうちどの方法が電子商取引ベンダーに適切であるかは、電子商取引アプリケーションにより決まります。

レイヤ 3 スティッキの使用

セッションの間、ユーザが一箇所に留まることが必要なアプリケーションでは、レイヤ 3 スティッキを使用します。レイヤ 3 スティッキでは、ユーザの IP アドレスに基づいて、ユーザをサーバに固定します。スティッキ テーブルに含まれるエントリの総数は、CPU メモリのサイズによって異なります（CPU メモリが 288MB の場合は 128K のスティッキ テーブル、CPU メモリが 144MB の場合は 32K のスティッキ テーブル）。

使用しているサイトの大きさが、128K または 32K を超えるユーザ数を同時に収容できる場合や、大部分の顧客がメガプロキシ経由でサイトにアクセスするような場合は、別のスティッキ方式（拡張バランス方式（advanced-balance）の **cookies**、**cookieurl**、または **url** など）を使用するか、スティッキ マスクを増やすことを検討してください。

（注）　**sticky-inact-timeout**　コマンドを使用してスティッキ接続に無活動タイムアウト期間を指定すると、スティッキ テーブルがいっぱいになったが、どのエントリも期限が切れていないときは、CSS はスティッキが必要な後続の要求を拒否します。

デフォルトのスティッキ マスクは、255.255.255.255 です。つまり、スティッキ テーブルの各エントリは個々の IP アドレスを表します。メガプロキシの中には、1 人のユーザが、1 セッション期間中に、あるアドレス範囲内の複数の異なる IP アドレスを使用できるものがあります。このように 1 つのセッションに対して複数のアドレスを使用すると、1 つのトランザクションに対して TCP 接続が別々のサーバに固定されることがあります。そのため、ショッピング カートから商品が失われる可能性があります。このような問題が起こらないようにするために、より高度なスティッキ方式を使用する必要があります。それができない場合は、スティッキ マスク 255.255.240.0 を使用することをお勧めします。

レイヤ 4 スティッキの使用

レイヤ 4 スティッキの機能は、送信元 IP アドレス、プロトコル、および宛先ポートの組み合わせに基づいて固定することを除けば、レイヤ 3 スティッキと同じです。レイヤ 4 スティッキもスティッキ テーブルを使用し、レイヤ 3 スティッキと同じ制約を受けます。

CSS は、2 つの宛先ポートに同じ IP アドレスが使用されていることを確認すると、2 つのエントリを使用します。レイヤ 4 スティッキにもスティッキ マスクを適用できます。

使用しているサイトが、すべての並行セッションを処理できるかどうか不安がある場合は、レイヤ 5 拡張バランス方式の **arrowpoint-cookie**、**cookie**、**cookieurl**、または **url** の使用を検討してください。

レイヤ 5 スティッキの使用

レイヤ 5 スティッキは、宛先 IP アドレス、プロトコル、ポート、および URL の組み合わせを使用します。HTTP クッキーやドメイン名を使うこともできます。レイヤ 5 スティッキは、クッキーまたは URL 内のスティッキ文字列、または SSL バージョン 3 のセッション ID に基づいて機能します。**arrowpoint-cookie**、**cookie**、**cookieurl**、**url** などの拡張バランス方式では、スティッキ テーブルを使用して ID を追跡しません。SSL スティッキの **advanced-balance ssl** 方式でも、スティッキ テーブルを使用します。

（注）　**sticky-inact-timeout** コマンドを使用してスティッキ接続に非アクティブ タイムアウト期間を指定すると、スティッキ テーブルがいっぱいになったが、どのエントリも期限が切れていないときは、CSS はスティッキが必要な後続の要求を拒否します。**sticky-inact-timeout** コマンドを、SSL スティッキを使用するレイヤ 5 コンテンツ ルールに指定した場合、スティッキ テーブルはいっぱいになっても SSL セッションは続行します。ただし、新しいセッションのスティッキ性は失われます。

CSS へのスティッキの設定

CSS にスティッキを設定するには、次の作業を行う必要があります。

- サイトの要件に応じた、使用するスティッキ方式の決定（レイヤ 3、レイヤ 4、文字列方式など）
- フェールオーバー方式の設定

拡張バランス方式の **cookies**、**url**、または **cookieurl** を使用する場合は、次の作業を行う必要もあります。

- 完全文字列一致、またはハッシュのどちらを使用するかの決定とその機能の設定
- 文字列を区切る（設定する）方法の決定

CSS にスティッキを設定するには、次の作業を行います。

1. **advanced-balance** コマンドとそのオプションを使用して、スティッキ方式を設定します。**advanced-balance** コマンドのオプションは、この章の「スティッキ コンテンツへの拡張ロード バランシング方式の指定」で後述します。
   - レイヤ 3 スティッキを設定する場合は、コンテンツ ルールで **advanced-balance sticky-srcip** を使用します。必要に応じて、スティッキ マスクをデフォルトの 255.255.255.255 から変更します。
   - レイヤ 4 スティッキを設定する場合は、コンテンツ ルールで **advanced-balance sticky-srcip-dstport** を使用します。必要に応じて、スティッキ マスクをデフォルトの 255.255.255.255 から変更します。
   - スティッキ クッキーを設定する場合は、コンテンツ ルールで **advanced-balance cookies** を使用します。
   - スティッキ URL を設定する場合は、コンテンツ ルールで **advanced-balance url** を使用します。
   - URL を指定してスティッキ クッキーを設定する場合は、コンテンツ ルールで **advanced-balance cookieurl** を使用します。
2. フェールオーバー方式を設定します。**sticky-serverdown-failover** コマンドを使用して、スティッキ文字列が見つかっても、関連するサービスが失敗したか一時停止している場合に行われる動作を定義します。スティッキ フェールオーバーのデフォルトは、CSS で設定済みのロード バランシング方式を使用することです。**sticky-serverdown-failover** オプションについては、この章で後述する「スティッキ サーバダウン フェールオーバーの設定」を参照してください。

sticky-srcip または **sticky-srcip-dstport** の拡張バランス方式を設定した場合は、これ以降の手順は必要ありません。

拡張バランス方式の **cookies**、**url**、または **cookieurl** を設定している場合は、手順 3 と 4 を行います。

3. 拡張バランス方式の **cookies**、**url** または **cookieurl** を使用している場合は、完全文字列一致またはハッシュのどちらを使用するか決定します。

   完全文字列一致を使用するには、次の操作を行います。

   a. **string operation match-service-cookie** コマンドを入力します（これは **string operation** コマンドのデフォルトです）。

   b. サービス設定ごとに、サービス モードの **string** コマンドを使用して、各サーバに一致させるために使用する一意の文字列を設定します。

      たとえば、3 つのサーバがある場合、照合する文字列として、サービス 1 には serverid111、サービス 2 には serverid112、サービス 3 には serverid113 を設定します。Web サーバ アプリケーションでクッキーやパス パラメータを設定するときに、これらの文字列が使用されるように設定します。

   string operation match-service-cookie コマンドの詳細については、この章で後述する「文字列操作の指定」を参照してください。

   ハッシュ アルゴリズムを使用するには、次の操作を行います。

   a. コンテンツ ルールで **string operation** コマンドを入力します。

   b. 使用するハッシュ方法に応じて、オプション（**hash-a**、**hash-crc32**、または **hash-xor**）を選択します。ハッシュを使用する場合は、各サーバが、他のすべてのサーバに設定されたクッキーを受け入れるようにする必要があります。

      指定する文字列に応じて、**hash-xor** または **hash-crc32** を使用することをお勧めします。文字列がまったく似ていない場合は、**hash-xor** を使用します。文字列が似ている場合は、**hash-crc32** を使用します。たとえば、文字列の値が abc1、abc2、および abc3 の場合、**hash-xor** 方式ではハッシュ値には大きな違いがみられません（つまり、abc1 と abc2 は、同じ値にハッシュされた結果、同じサーバに接続される可能性があります）。

      文字列操作のハッシュ オプションの詳細については、この章で後述する「文字列操作の指定」を参照してください。

4. **advanced-balance cookie**、**url** または **cookieurl** を使用している場合は、文字列を区切る（設定する）方法を決定します。次の owner-content **string** コマンドを使用して、文字列を区切ります。

- **string range** ：文字列範囲を定義して、検索のサイズを制限することができる。デフォルトでは、使用する方式に応じて、クッキー、URL または URL のパラメータの最初の 100 バイトが検索されます。クッキーや URL で文字列が現れる場所が分かっている場合は、その文字列の範囲を定義します。範囲は 1 ～ 2000 で、デフォルトは 1 ～ 100 です。文字列範囲のオプションについては、この章で後述する「文字列範囲の設定」を参照してください。
- **string eos-char** ：文字列の終わりを区切る 3 文字以内の ASCII 文字。文字列の長さが一定でない場合は、このオプションを使用します。**string process-length** は **string eos-char** を無効にするので注意が必要です。どちらのオプションも設定しない場合、CSS では最大 100 バイトを区切り文字として使用します。
- **string prefix** ：文字列のプレフィックス（30 文字まで）を使用して、クッキーや URL 内の文字列範囲が検索される。文字列プレフィックスが見つからない場合は、通常のバランシング方式が使用されます。
- **string process-length** ：プレフィックスの後の文字列範囲内のバイト数に、文字列を区切るときに使用されるスキップの長さを足した後のバイト数を指定する。文字列の長さが一定の場合は、このオプションを使用します。
- **string skip-length** ：プレフィックスの後ろでスキップするバイト数を指定する。範囲は 0 ～ 64 です。

たとえば、ipaddr=192.168.3.6& を使用している場合は、IP アドレスの長さが一定ではないので、**string prefix** を ipaddr=、**string eos-char** を **&** として使用します。

たとえば、server ID=server111 を使用している場合は、文字列の長さは固定されているため、**string prefix** を 「server ID=」、**string process-length** を 8 として使用します。

表 11-1 に、スティッキ ルールとそのコンテンツ ルールへの適用方法を説明します。

表 11-1　スティッキ ルールのコンテンツ ルールへの適用

ルール タイプ	スティッキの設定	スティッキ性の基準
レイヤ 3 コンテンツ ルール	**advanced-balance sticky-srcip**	スティッキ マスクを使用した送信元 IP アドレス
レイヤ 4 コンテンツ ルール	**advanced-balance sticky-srcip-dstport**	スティッキ マスクを使用した送信元 IP アドレスおよび宛先ポート

表 11-1　スティッキ ルールのコンテンツ ルールへの適用（続き）

ルール タイプ	スティッキの設定	スティッキ性の基準
スティッキ文字列を使用しないレイヤ 5 コンテンツ ルール	**advanced-balance sticky-srcip-dstport**	スティッキ マスクを使用した送信元 IP アドレスおよび宛先ポート
スティッキ文字列を使用したレイヤ 5 コンテンツ ルール	**advanced-balance cookies**（または **advanced-balance cookieurl**）	クッキーまたは URL でのスティッキ文字列の検索。クッキーや URL でスティッキ文字列が見つからない場合は、利用可能なサーバ間で各要求のロードバランシング処理が行われます。
SSL を使用したレイヤ 5 コンテンツ ルール	**advanced-balance ssl**	SSL v3 セッション ID。セッション ID が存在しない場合は、送信元 IP アドレスと宛先ポートを使用してスティッキ性が維持されます。

（注）　一部の環境では、URL、cookie 文字列、または他の HTTP ヘッダー情報が複数のパケットにまたがっている場合があります。このような環境では、レイヤ 5 の情報を得るために複数のパケットを解析してから、ロード バランシングの判断を行うことができます。グローバル設定モードの **spanning-packets** コマンドを使用すると、20 パケットまで解析できるようになります（デフォルトは 6）。ロードバランシングの判断は、一致が見つかるとただちに行われるので、設定した数のパケットをすべて解析する必要はありません。複数のパケットを解析すると接続の遅延は必然的に長くなるため、複数のパケットにわたる長い文字列はパフォーマンスに影響を与えます。**spanning-packets** コマンドの使用については、第 10 章「コンテンツ ルールの設定」を参照してください。

スティッキ コンテンツへの拡張ロード バランシング方式の指定

同じユーザやクライアントからの追加セッションが、その最初の接続と同じサービスに送信され、通常のロード バランシングが無効になると、コンテンツ ルールは「スティッキ」になります。デフォルトでは、拡張バランシング方式は無効です。

advanced-balance コマンドを使用して、スティッキ性を持つコンテンツ ルールに、拡張ロード バランシング方式を指定します。**advanced-balance** コマンドのオプション、**cookies**、**cookieurl**、および **url** では、文字列を使用してクライアントをサーバに固定します。文字列方式ではスティッキ テーブルは使用されないため、スティッキ テーブルの上限が、使用しているアプリケーション要件に対して小さすぎる場合、このオプションは便利です。

このコンテンツ モード コマンドのシンタックスとオプションは次のとおりです。

- **advanced-balance arrowpoint-cookie** ：arrowpoint クッキー内の、選択されたサーバの一意のサービス ID 情報に基づいて、クライアントをサーバに固定できるようにコンテンツ ルールを設定する。**(config-service) string** コマンドを使用して、サービス ID を設定します。arrowpoint クッキーの設定の詳細については、この章で後述する「ArrowPoint クッキーの設定」を参照してください。このオプションは、どのレイヤ 5 コンテンツ ルールでも使用できます。

（注）　**advanced-balance** コマンドの **arrowpoint-cookie** オプションを使用している場合は、文字列一致基準を設定せず、**sticky-no-cookie-found-action** コマンドも **sticky-serverdown-failover** コマンドも使用しないでください。

（注）　**arrowpoint-cookie expiration** コマンドと **advanced-balance arrowpoint-cookie** コマンドを設定すると CSS の CPU 使用率が急激に上昇し、パフォーマンスの低下につながる恐れがあります。

- **advanced-balance cookies** ：HTTP クッキー ヘッダーにある設定済み文字列に基づいて、クライアントをサーバに固定できるようにコンテンツ ルールを設定する。このオプションを使用する場合は、コンテンツ ルールでポートを指定する必要があります。これによって、CSS はその接続をスプーフします。スティッキ設定が **advanced-balance cookies** に設定されているコンテンツ ルールでは、すべてのクライアントで、ブラウザのクッキーを有効にする必要があります。

  クライアントが最初に要求を行ったときには、クッキーはありません。クライアントは、クッキーを設定できるサーバにアクセスして、そのサーバからクッキーを受け取ります。次からの各要求には、期限切れにならない限り、そのクッキーが含まれます。クッキーの文字列を使用して、クライアントをサーバに固定できます。サービス モードの **string** コマンドを使用すると、クッキー内で文字列を検索する場所を指定できます。

  CSS では、次の基準を使用してクッキーを処理します。

  － サービスの設定時に設定した完全一致

  － ハッシュ　アルゴリズムのデータ。詳細については、この章で後述する「ハッシュ方式と一致方式の比較」を参照してください。

- **advanced-balance cookieurl** ：**advanced-balance cookies** コマンドに同じ。ただし、HTTP パケットにクッキー ヘッダーが見つからない場合、このタイプは、同じ文字列基準に基づき、フェールオーバーして URL 拡張子（つまり、URL の「？」の後の部分）を検索します。このオプションを使用する場合は、コンテンツ ルールでポートを指定する必要があります。これによって、CSS はその接続をスプーフします。

  このオプションは、Cookie Munger と共に Microsoft IIS Web サーバを使用している場合に便利です。このオプションは、セッションのステート情報を、クライアントがクッキーを受け入れられるかどうかによって、クッキー ヘッダーまたは URL 拡張子に動的に挿入します。

  クライアントのアプリケーションの中には、クッキーを受け入れないものもあります。サイトがクッキーにある情報に依存している場合、管理者はサーバ アプリケーションを変更して、クッキーのデータを URL のパラメータ セクションに付け加えることもあります。このパラメータは、通常、URL のメイン データ セクションの終わりの「？」の後に続きます。

  advanced-balance cookieurl コマンドでは、設定した文字列を次の場所で検索して、クライアントをサーバに固定します。

  － クッキー（クッキーが存在する場合）

  － URL のパラメータ セクション（クッキーが存在しない場合）

  文字列は、完全一致方式、またはハッシュ方式を用いることができます。

- **advanced-balance none** ：コンテンツ ルールで拡張ロード バランシング方式を無効にする（デフォルト）。
- **advanced-balance sticky-srcip** ：クライアントの IP アドレスに基づいてクライアントをサーバに固定できるようにコンテンツ ルールを設定する。レイヤ 3 スティッキ性とも呼びます。このオプションは、レイヤ 3、レイヤ 4、またはレイヤ 5 のコンテンツ ルールで使用できます。
- **advanced-balance sticky-srcip-dstport** ：クライアント IP アドレスとサーバの宛先ポート番号の両方に基づいて、クライアントをサーバに固定できるようにコンテンツ ルールを設定する。レイヤ 4 スティッキ性とも呼ばれます。このオプションは、レイヤ 4 またはレイヤ 5 のコンテンツ ルールで使用できます。
- **advanced-balance sip-call-id** ：Session Initiation Protocol（SIP）Call-ID に基づいて、クライアントをサーバに固定できるようにコンテンツ ルールを設定する。コンテンツ ルールでアプリケーション タイプは **sip**、プロトコルは UDP とする必要があります。SIP の詳細については、「Session Initiation Protocol ロード バランシングの設定」を参照してください。
- **advanced-balance ssl** ：サーバで割り当てられた Secure Socket Layer（SSL）バージョン 3 のセッション ID に基づいて、クライアントをサーバに固定できるようにコンテンツ ルールを設定する。コンテンツ ルールのアプリケーション タイプは、SSL にする必要があります。このオプションを使用する場合は、コンテンツ ルールでポートを指定する必要があります。これによって、CSS はその接続をスプーフします。

  セキュリティのために暗号化が必要なサイトの多くで、SSL が使用されます。SSL にはセッション ID があり、CSS には、このようなセッション ID を使用してクライアントをサーバに固定する機能が用意されています。CSS で SSL スティッキ性を提供できるようにするには、アプリケーションで SSL バージョン 3 のセッション ID を使用する必要があります。スティッキ SSL ではスティッキ テーブルが使用されます。セキュリティは十分だが、同時に使用できるセッションの数に不安がある場合は、**cookies**、**cookieurl**、または **url** オプションの使用を検討してください。

（注）　**ssl-l4-fallback disable** コマンドは、CSS が送信元 IP アドレスと宛先アドレスのペアに基づきレイヤ 4 ハッシュ値をスティッキ テーブルに挿入しないようにする場合に使用します。たとえば、少数のクライアントおよびサーバを使って SSL をテストする実験環境で、再送が行われる場合は、このコマンドの使用が必要になる場合があります。その場合、レイヤ 4 ハッシュ値を使用すると正しいテスト結果

が得られなくなります。詳細については、この章で後述する「SSL レイヤ 4 フォールバックの設定」を参照してください。

ネットワークで SSL バージョン 2 を使用している場合は、**ssl-l4-fallback disable** コマンドを使用しないでください。

- **advanced-balance url** ：HTTP 要求の URL にある設定済みの文字列に基づいてクライアントをサーバに固定できるようにコンテンツ ルールを設定する。このオプションを使用する場合は、コンテンツ ルールでポートを指定する必要があります。これによって、CSS はその接続をスプーフします。

  advanced-balance url コマンドは、**advanced-balance cookies** コマンドと同様です。完全一致方式またはハッシュ アルゴリズムのどちらでも使用できます。文字列は URL のどこに存在しても構いません。

- **advanced-balance wap-msisdn** ：HTTP 要求の MSISDN ヘッダー フィールドに基づいて、クライアントをサーバに固定できるようにレイヤ 5 コンテンツ ルールを設定する。MSISDN は、Wireless Application Protocol（WAP）を使用する無線のクライアントが使用するヘッダー フィールドです。MSISDN フィールドの値には、そのクライアントを一意に識別する電話番号やユーザ ID を指定することができます。このコマンドは、特に電子商取引アプリケーションを使用しているクライアントに役立ちます。

（注）　**advanced-balance wap-msisdn** は、レイヤ 5 コンテンツ ルール（URL 文を使用して設定したルール）でだけ設定することをお勧めします。

HTTP 要求に MSISDN ヘッダーが存在すると、CSS はこの MSISDN ヘッダー フィールドの値に基づいて、ハッシュ値（キー）を生成します。CSS は、このキーを使用してスティッキ テーブルのエントリを検索します。スティッキ テーブルにこのキーが存在すると、CSS はそのテーブル エントリに示されているスティッキ サーバにそのクライアントを送信します。

スティッキ テーブルに該当するエントリが存在しない場合、CSS は次の処理を行います。

a. スティッキ テーブルに新しいエントリを作成する（レイヤ 3、レイヤ 4、および SSL スティッキと同様）。
b. 負荷のバランスを取り、1 つのサーバに要求を送信する。

c. 選択したサーバとキー（MSISDN ヘッダーのハッシュ値）をスティッキ テーブルのエントリに保存する。

この後に同じクライアントから送信される要求については、CSS は、同じテーブル エントリを検索し、このクライアントを同じサーバに送信します。

HTTP 要求に MSISDN ヘッダー フィールドが存在しない場合、CSS は設定されているロード バランシング方式に基づいてクライアントの要求の負荷を分散します。デフォルトのロード バランシング方式は、ラウンドロビンです。

次の例では、192.168.128.151 宛ての TCP ポート 80 のトラフィックは、MSISDN HTTP ヘッダー フィールドの内容に基づいて、server1 または server2 のいずれかに固定されます。

```
owner arrowpoint
   content ruleWapSticky
     vip address 192.168.128.151
     protocol tcp
     port 80
     url "/*"
     add service server1
     add service server2
     advanced-balance wap-msisdn
     active
```

たとえば、コンテンツ ルール *rule1* に **advanced-balance wap-msisdn** を指定するには、次のコマンドを入力します。

```
(config-owner-content[arrowpoint-rule1])# advanced-balance
wap-msisdn
```

（注）　**advanced-balance wap-msisdn** コマンドは、単独で使用することも、MSISDN ヘッダー フィールド タイプと共に使用することもできます。両方を使用する場合の設定例については、この章で後述する「電子商取引アプリケーションを利用する無線ユーザの設定」を参照してください。

拡張ロード バランシング方式を無効にするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# advanced-balance none
```

SSL レイヤ 4 フォールバックの設定

スティッキ テーブルへのレイヤ 4 ハッシュ値の挿入は、3 つより多いフレームがいずれかの方向（クライアントからサーバ、サーバからクライアント）に送信された場合、または、ネットワークで SSL バージョン 2 が使用されている場合に起こります。どちらの場合も、CSS はレイヤ 4 ハッシュ値をスティッキ テーブルに挿入し、SSL バージョン 3 セッション ID のこれ以降の使用は無効になります。**ssl-l4-fallback disable** コマンドは、CSS が送信元 IP アドレスと宛先アドレスのペアに基づいて、スティッキ テーブルにレイヤ 4 ハッシュ値を挿入（CSS のデフォルトの動作）しないようにする場合に使用します。

ssl-l4-fallback コマンドは、コンテンツ ルールに **advanced-balance ssl** 方式が指定され、そのコンテンツ ルールが SSL バージョン 3 のセッション ID に基づいて特定のサーバに強制的に使用される場合だけに適用できます。**ssl-l4-fallback** コマンドは、少数のクライアントとサーバで SSL のテストを行い、ある程度の再送信が発生しうるような実験的な環境で、必要になることがあります。この場合、レイヤ 4 ハッシュ値を使用すると正しいテスト結果が得られなくなります。

（注）　**ssl-l4-fallback** コマンドは、グローバル設定モード コマンドであり、**advanced-balance ssl** 方式を使用するすべてのコンテンツ ルールに影響を与えます。

このグローバル設定モード コマンドのオプションは次のとおりです。

- **ssl-l4-fallback enable** ：CSS は、スティッキ テーブルにレイヤ 4 ハッシュ値を挿入する（デフォルト設定）。
- **ssl-l4-fallback disable** ：CSS は、スティッキ テーブルにレイヤ 4 ハッシュ値を挿入せず、引き続き SSL バージョン 3 セッション ID を検索する。

（注）　ネットワークで SSL バージョン 2 を使用している場合は、**ssl-l4-fallback disable** コマンドを使用しないでください。

たとえば、CSS がスティッキ テーブルにレイヤ 4 ハッシュ値を挿入しないようにするには、次のように入力します。

```
(config)# ssl-l4-fallback disable
```

CSS を、スティッキ テーブルにレイヤ 4 ハッシュ値を挿入するデフォルトの動作にリセットするには、次のように入力します。

```
(config)# ssl-l4-fallback enable
```

スティッキ サーバダウン フェールオーバーの設定

スティッキ フェールオーバーのデフォルト方式では、CSS で設定済みのロード バランシング方式が使用されます。スティッキ文字列が検出され、関連するサービスが失敗または一時停止した場合の動作を、**sticky-serverdown-failover** コマンドを使用して定義します。

（注）　**advanced-balance** コマンドの **arrowpoint-cookie** オプションを使用している場合は、文字列一致基準、**sticky-no-cookie-found-action** コマンド、**sticky-serverdown-failover** コマンドのいずれも設定しないでください。

このコンテンツ モードのコマンドのシンタックスおよびオプションは、次のとおりです。

- **sticky-serverdown-failover balance** ：設定済みのロード バランシング方式に基づいてサービスを使用するフェールオーバー方式を設定する。
- **sticky-serverdown-failover redirect** ：コンテンツ ルールに設定されているリダイレクト文字列を使用するフェールオーバー方式を設定する。このコマンドは 252 文字のリダイレクト文字列（URL）をサポートします。リダイレクト文字列の詳細については、第 10 章「コンテンツ ルールの設定」の「コンテンツへの要求のリダイレクション」を参照してください。コンテンツ ルールにリダイレクト文字列を設定していない場合は、設定済みのロード バランシング方式が使用されます。
- **sticky-serverdown-failover reject** ：コンテンツ要求を拒否する。
- **sticky-serverdown-failover sticky-srcip** ：クライアントの送信元 IP アドレスに基づいてサービスを使用するフェールオーバー方式を設定する。
- **sticky-serverdown-failover sticky-srcip-dstport** ：クライアントの送信元 IP アドレスとサーバの宛先ポートに基づいてサービスを使用するフェールオーバー方式を設定する。

たとえば、スティッキ フェールオーバー方式を **sticky-srcip** に設定するには、次のコマンドを入力します。

```
(config-owner-content[arrowpoint-rule1])sticky-serverdown-failover
sticky-srcip
```

スティッキ フェールオーバー方式を、設定済みロード バランシング方式を使用するデフォルト設定に設定するには、次のコマンドを入力します。

```
(config-owner-content[arrowpoint-rule1])# no
sticky-serverdown-failover
```

スティッキ マスクの設定

クライアントの IP アドレスは、クライアントを一意に識別します。通常のクライアントとサーバ間のセッションでは、IP アドレスは接続の間維持されます。ただし、（高密度のプロキシ フェールオーバーなどが原因で）接続が失われ、クライアントが別の IP アドレスで再接続する場合、CSS はクライアント情報（ショッピング カートや金融セッションなど）を保持している同じサーバにクライアントを再接続する必要があります。

sticky-mask コマンドを使用して、クライアントの送信元 IP アドレスが変わってもクライアントの接続状態を維持できるように、クライアントの IP アドレスのグループをマスクします。スティッキ マスクには、CSS がマスクするクライアントの IP アドレスの部分を指定します。デフォルトのスティッキ サブネット マスクは、255.255.255.255 です。

たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# sticky-mask 255.255.255.0
```

スティッキ サブネット マスクをデフォルトの 255.255.255.255 に戻すには、次のコマンドを入力します。

```
(config-owner-content[arrowpoint-rule1])# no sticky-mask
```

無活動タイムアウトの設定

新しいスティッキ接続には、デフォルトでは最も古い使用済みのスティッキ エントリが使用されます。CSS のスティッキ トラフィックの負荷によっては、スティッキの結合がしばらくの間起こることがあります。**sticky-inact-timeout** コマンドを使用すると、スティッキ エントリがスティッキ テーブルから削除されるまでのスティッキ接続の無活動タイムアウト期間をコンテンツ ルールに指定することができます。この期間を設定すると、指定された期間、スティッキ エントリはスティッキ テーブルに保持されます。期限が切れるまで、このエントリは再使用されません。スティッキ テーブルがいっぱいで、期限切れのエントリがない場合、新しいスティッキ要求は拒否されます。**sticky-inact-timeout** コマンドを、SSL スティッキを使用するレイヤ 5 コンテンツ ルールに指定した場合、スティッキ テーブルがいっぱいになっても SSL セッションは続行します。ただし、新しいセッションのスティッキ性は失われます。

スティッキ接続の期限が切れると、設定済みのロード バランシング方式を使用して、要求に対応できるサーバが選択されます。

この機能が無効の場合、新しいスティッキ接続では最も古い使用済みのスティッキ エントリが使用されます。CSS のスティッキ トラフィックの負荷によっては、スティッキの結合がしばらくの間起こることがあります。

このコマンドのシンタックスは次のとおりです。

sticky-inact-timeout *minutes*

無活動時間（分）を、0 ～ 65535 の整数で入力します。デフォルト値は 0 で、この機能が無効であることを意味します。次に例を示します。

```
(config-owner-content[arrowpoint-rule1])# sticky-inact-timeout 9
```

スティッキ接続の無活動タイムアウト機能を無効にするには、次のコマンドを入力します。

```
(config-owner-content[arrowpoint-rule1])# no sticky-inact-timeout
```

SSL へのスティッキ コンテンツの設定

SSL バージョン 3 のセッション ID に基づいたスティッキ性を使用するには、SSL レイヤ 5 ルールをサービスに個別に設定します。SSL レイヤ 5 ルールをサービスに設定するには、次の操作を行います。

- **(config-owner-content) port** コマンドを使用して、ポートを 443 に設定します。
- **(config-owner-content) advanced-balance ssl** コマンドを使用して、コンテンツ ルールが SSL に基づいてスティッキになるようにします。
- **(config-owner-content) application ssl** コマンドを使用して、SSL アプリケーション タイプを指定します。

（注）　**application ssl** コマンドは、必ず **advanced-balance ssl** コマンドと組み合わせて設定することをお勧めします。**application ssl** コマンドを使用すると、CSS は接続をスプーフしてサーバからの応答が認識できるようにします。**advanced-balance ssl** コマンドを使用すると、CSS はサーバから送られる SSL セッション ID を検索して、そのセッション ID に基づいてクライアントとサーバを接続します。フローが設定されると、**application ssl** コマンドにより、CSS はフローをレイヤ 4 フローとして処理し、フロー内のレイヤ 5 データを検査しません。これにより、暗号化されたデータの解読を誤ることが避けられます。

たとえば、次に示す起動設定（startup-config）の owner 部分には、SSL 用に設定されているコンテンツ ルールが示されています。この例の **url "/*"** コマンドはオプションです。**application ssl** コマンドと **advanced-balance ssl** コマンドの組み合わせによって、ルールがレイヤ 5 に移行します。

```
!************************** OWNER ***************************!
owner arrowpoint
content L5sslsticky
vip address 192.3.6.58
add service server87
add service server88
balance aca
protocol tcp
port 443
url "/*"
advanced-balance ssl
application ssl
active
```

文字列範囲の設定

CSS による文字列検索に使用されるクッキーや URL、URL 拡張子の開始バイトと終了バイトの位置を指定すると、CSS はこの範囲内の情報だけを処理します。これにより、CSS が各クッキー、URL または URL 拡張子を調べる際の情報処理量が制限され、パフォーマンスが向上します。デフォルトでは、文字列範囲は、クッキー、URL または URL のパラメータの最初の 100 バイトです。

（注）　開始位置がクッキー、URL または URL 拡張子の長さを超える場合、文字列機能は実行されません。終了位置がクッキー、URL または URL 拡張子の長さを超える場合、文字列の処理は、該当のヘッダーの終わりで停止します。

string-range コマンドを使用して、指定した文字列を検索するために使用される、クッキー、URL、または URL 拡張子内の開始および終了バイト位置を指定できます。*start_byte* 変数には、ヘッダーの後のクッキー、URL、または URL 拡張子の開始バイト位置を入力します。1 ～ 1999 の整数で入力します。デフォルトは 1 です。開始バイト位置は、必ず終了バイト位置よりも小さくなるようにします。

end_byte 変数には、クッキー、URL、または URL 拡張子の終了バイト位置を入力します。2 ～ 2000 の整数で入力します。デフォルトは 100 です。終了バイト位置は、必ず開始バイト位置よりも大きくなるようにします。

advanced-balance コマンドのオプションに応じて、次のようになります。

- **cookies** ："Cookie: "（コロンの後のスペースも含む）の後からカウントが開始される。
- **url** ：/ の後ろからカウントが開始される。
- **cookieurl** ：文字列「Cookie: 」の直後からカウントが開始される。HTTP 要求内に文字列「Cookie: 」が見つからない場合は、同じ要求の URL の ? の直後からカウントが開始されます。

次に範囲指定の一例を示します。

```
(config-owner-content[arrowpoint-rule1])# string-range 35 to 55
```

文字列の範囲をデフォルトの 1 ～ 100 に戻すには、次のコマンドを入力します。

```
(config-owner-content[arrowpoint-rule1])# no string-range
```

文字列操作の指定

string operation コマンドを使用して、文字列結果の宛先サーバを選択する方式を決定します。CSS は、**string** 基準コマンドの設定から文字列範囲内の文字列結果を取得します。設定したバランス方式を使用するか、または指定したスティッキ ハッシュ タイプにより生成されたハッシュ キーによって、サーバを選択することができます。どちらの方式を使用するかは、Web サーバによって次のように異なります。

- Web サーバが、そのサーバで設定したクッキーしか受け入れない場合は、完全一致方式を使用する必要がある。
- Web　サーバが、クッキー　サーバまたは別のサーバで設定されたどのクッキーでも受け入れることができる場合は、ハッシュ方式を使用できる。

（注）　**advanced-balance** コマンドの **arrowpoint-cookie** オプションを使用している場合は、文字列一致基準、**sticky-no-cookie-found-action** コマンド、**sticky-serverdown-failover** コマンドのいずれも設定しないでください。

ハッシュ方式と一致方式の比較

アプリケーションで完全一致方式が使用されている場合、クライアントが特定のサーバに要求を行うと、サーバは、今後の要求で使用するためにサーバ固有の文字列をクライアントに提供する責任があります。通常、別のサーバで設定された文字列をサーバが要求として受け取ると、その文字列が原因でエラーが発生します。完全一致では、固有の文字列が検索されます。完全一致が見つかると、そのサーバが使用されます。一致が見つからない場合は、設定済みのロード バランシング方式を使用して、クライアントに対応するサーバが選択されます。

アプリケーションでハッシュ アルゴリズムの 1 つが使用される場合、すべてのサーバが他のサーバで設定された文字列を受け入れることができます。このモデルを使うと、最初のログインでクライアントを Web サーバに送信し、そこでクッキーをクライアントに割り当てるように、サイトを設定することができます。CSS がクッキー文字列を持つクライアントから最初の要求を受け取ると、その文字列に対してハッシュ操作を実行して、対応するサーバを選択します。ハッシュ アルゴリズムを使用すると、特定の文字列が常に特定のサーバに送信されますが、完全一致の場合とは異なり、定義済みのサーバでなくてもかまいません。

文字列操作のハッシュ アルゴリズムを使用すると、Web サーバ アプリケーションの変更を行わずに使用することができます。**string operation match-service-cookie** 方式を使用する場合は、各サーバで一意の文字列が生成されるように、Web サーバ アプリケーションを変更する必要があります。ハッシュ アルゴリズムでは、サーバによりすでに生成されている文字列を利用することができる場合があります。

このコンテンツ モードのコマンドのシンタックスおよびオプションは、次のとおりです。

- **string operation match-service-cookie** ：スティッキ文字列のサービス クッキーを照合し、サーバを選択する。これはデフォルト設定です。一致しない場合、設定済みのロード バランシング方式（ラウンドロビンなど）を使用して、サーバが選択されます。
- **string operation** [**hash-a**|**hash-crc32**|**hash-xor**] ：指定されたハッシュ方式で生成されたハッシュ キーを使用して、サーバを選択する。**advanced-balance coookies** でハッシュ アルゴリズムを使用する場合は、同じドメインのすべてのサーバは、クッキーを作成したサーバかどうかにかかわらず、クッキーを受け入れる必要があります。これにより、レイヤ 5 ルールで設定されたすべてのサーバが、HTTP 要求で渡されたクッキーを処理できます。

  ハッシュ方式のキーワードは次のとおりです。

  - **hash-a** ：基本ハッシュ アルゴリズムをハッシュ文字列に適用して、ハッシュ キーを生成する。
  - **hash-crc32** ：CRC32 アルゴリズムをハッシュ文字列に適用して、ハッシュ キーを生成する。
  - **hash-xor** ：ハッシュ文字列の XOR（排他 OR）の各バイトで、最終ハッシュ キーを導き出す。

  選択したサーバがサービスを行っていない場合は、再ハッシュが実行され別のサーバが選択されます。

  指定する文字列に応じて、**hash-xor** または **hash-crc32** を使用することをお勧めします。文字列がまったく似ていない場合は、**hash-xor** を使用します。文字列が似ている場合は、**hash-crc32** を使用します。たとえば、文字列の値が abc1、abc2、および abc3 の場合、**hash-xor** 方式ではハッシュ値には大きな違いがみられません（つまり、abc1 と abc2 は、同じ値にハッシュされた結果、同じサーバに接続される可能性があります）。

たとえば、文字列操作 **hash-crc32** アルゴリズムを使用してサーバを選択する文字列操作を設定するには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# string operation hash-crc32
```

文字列操作を、スティッキ文字列でサービス クッキーを照合してサーバを選択するデフォルト設定にリセットするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no string operation
```

CSS は、文字列結果を次の文字列基準コマンドから導き出します。

- **string ascii-conversion**
- **string match**
- **string eos-char**
- **string prefix**
- **string process-length**
- **string skip-length**

文字列の ASCII 変換の有効化または無効化

デフォルトでは、指定されたスティッキ文字列範囲内のエスケープ特殊文字が、文字列自体の処理前に ASCII 変換されます。この ASCII 変換の有効 / 無効を切り替えるには、**string ascii-conversion** コマンドを使用します。

たとえば、エスケープ特殊文字の ASCII 変換を無効にするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# string ascii-conversion
disable
```

エスケープ特殊文字の ASCII 変換を、再びデフォルトの有効に設定するには、コマンドの **no** 形式、または **enable** オプションを使用します。次に例を示します。

```
(config-owner-content[arrowpoint-rule1])# no string ascii-conversion

(config-owner-content[arrowpoint-rule1])# string ascii-conversion
enable
```

複数の文字列一致の処理

着信した文字列と一致する設定済みサービス文字列が複数検出された場合、CSS は、デフォルトでは最も一致度の高い（最も長い）文字列を選択します。たとえば、CSS サービスが次のように設定されていると仮定します。

```
service s1
string pear

service s2
string grape

service s3
string banana
```

この場合、着信文字列が grapebananapear であれば、文字列 banana が一致として選択されます。

string match コマンドを使用すれば、複数の文字列一致の処理を次のように変更できます。

- **first-string-match** キーワードを指定して、着信文字列の先頭に近い部分に一致する文字列を選択する。この場合は、次のように入力します。

  ```
  (config-owner-content[arrowpoint-rule1])# string match
  first-string-match
  ```

  この例では、文字列 grape が一致した文字列として選択されます。

- **first-service-match** キーワードを使用して、インデックス エントリの順序に従って各サービスを照合し、最初に見つかった一致文字列を選択する。この場合は、次のように入力します。

  ```
  (config-owner-content[arrowpoint-rule1])# string match
  first-service-match
  ```

  この例では、文字列 pear が最初のサービスで一致した文字列として選択されます。

（注）　**string match** コマンドは、**advanced-balance cookies**|**cookiesurl**|**url** コマンドと共に使用します。

デフォルトの扱い（最も一致度の高い文字列を選択）に戻すには、次のコマンドを実行します。

```
(config-owner-content[arrowpoint-rule1])# string match specific
```

文字列終了文字の指定

string eos-char コマンドを使用して、文字列範囲内のスティッキ文字列の区切り文字として 3 文字以内の ASCII 文字を指定します。たとえば、クッキー ヘッダーでは、セミコロン（;）が常に区切り文字として使用され、URL 拡張子では、多くの場合、アンパサンド（&）が区切り文字として使用されます。

(config-owner-content) string process-length コマンドが設定されていない場合は、**string eos-char** の値が使用されます。**(config-owner-content) string process-length** コマンドの方が優先度が高くなります。どちらのコマンドも設定されていない場合は、最終文字列の長さとして、最大 100 バイトが使用されます。スティッキ文字列の文字列終了文字は、3 文字以内のテキスト文字列を引用符で囲んで入力します。

たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# string eos-char ";"
```

文字列終了文字を消去にするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no string eos-char
```

文字列プレフィックスの指定

文字列プレフィックスを設定しない場合は、スティッキ タイプに応じて、クッキー、URL、または URL 拡張子の最初から文字列機能が開始されます。デフォルトでは、文字列範囲は、クッキー、URL または URL のパラメータの最初の 100 バイトです。文字列プレフィックスが指定されていても、文字列範囲内でプレフィックスが見つからない場合は、**sticky-serverdown-failover** コマンドで定義したロード バランシング方式が使用されます。

string prefix コマンドを使用して、文字列の範囲内にある文字列プレフィックスを指定します。文字列プレフィックスは、30 文字以内のテキスト文字列を引用符で囲んで入力します。デフォルトは、プレフィックスなしです（""）。

たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# string prefix "UID="
```

文字列プレフィックスを消去するには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no string prefix
```

文字列の処理の長さの指定

string prefix コマンドで指定されたプレフィックスの末尾から **string skip-length** コマンドで指定されたバイトをスキップした後に位置する、文字列操作対象のバイト長を指定するには、**string process-length** コマンドを使用します。このコマンドは、**string eos-char** コマンドよりも優先されます。どちらのコマンドも設定されていない場合、文字列操作には最大 100 バイトが使用されます。0 ～ 252 のバイト数を入力します。デフォルトは 0 です。

たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# string process-length 16
```

バイト数をデフォルトの 0 に設定するには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no string process-length
```

文字列スキップの長さの指定

文字列結果を検索する、プレフィックスの終わりからスキップする文字列範囲内のバイト数を指定するには、**string skip-length** コマンドを使用します。0 ～ 64 のバイト数を入力します。デフォルトは 0 です。たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# string skip-length 3
```

バイト数をデフォルトの 0 に設定するには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no string skip-length
```

Sticky-No-Cookie-Found-Action の設定

sticky-no-cookie-found-action コマンドを使用すると、クッキー ヘッダーまたは指定されたクッキー文字列が見つからないときにスティッキ クッキー コンテンツ ルールに対して実行するアクションを指定することができます。

（注）　**advanced-balance arrowpoint-cookie** コマンドを使用する場合は、**sticky-no-cookie-found-action** コマンドは設定しないでください。これらのコマンドには互換性がありません。

sticky-no-cookie-found-action コマンドのオプションは次のとおりです。

- **loadbalance**（デフォルト）：クライアント要求にクッキーが見つからない場合、設定済みのバランシング方式が使用される。
- **redirect** *"URL"* ：クライアント要求にクッキーが見つからない場合、クライアント要求は指定された URL 文字列にリダイレクトされる。このオプションを使用する場合は、リダイレクト URL も指定する必要があります。リダイレクト URL は、0 ～ 252 文字以内のテキスト文字を引用符で囲んで指定します。
- **reject** ：要求にクッキーが見つからない場合、クライアント要求は拒否される。
- **service** *service_name* ：要求にクッキーが見つからない場合、指定したサーバにクッキーなしのクライアント要求が送信される。

次にコマンドの使用例を示します。

```
(config-owner-content[arrowpoint-rule1])#
sticky-no-cookie-found-action redirect
"http://www.lml.com/nocookie.html"
```

sticky-no-cookie-found-action をデフォルトの **loadbalance** にリセットするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no
sticky-no-cookie-found-action
```

電子商取引アプリケーションと他のインターネット アプリケーション用のスティッキ パラメータの設定

電子商取引アプリケーション用のスティッキ パラメータを設定すると、クッキー内の文字列に基づいて固定するコンテンツ ルール宛てにクッキーを含まないクライアント要求が送信された場合、クライアント要求の処理方法を CSS に指示できます。また、HTTP ヘッダーによるロード バランシングと **advanced-balance wap-msisdn** コマンドを共に用いることによって、無線のユーザを処理する方法も指示できます。CSS スティッキ テーブルを使用するアプリケーションでは、定義したアクティビティ期間が終わると、スティッキ テーブルからエントリを削除できます。

ここでは、電子商取引アプリケーションと他のインターネット アプリケーション用スティッキ パラメータの設定方法について説明します。

- 「advanced-balance arrowpoint-cookie の設定」
- 「ArrowPoint クッキーの設定」
- 「ロケーション クッキーの設定」
- 「電子商取引アプリケーションを利用する無線ユーザの設定」
- 「Session Initiation Protocol ロード バランシングの設定」

advanced-balance arrowpoint-cookie の設定

advanced-balance arrowpoint-cookie コマンドを使用すると、arrowpoint で生成されたクッキー内にある、選択されたサーバの一意なサービス ID 情報に基づいて、コンテンツ ルールでクライアントをサーバに固定することができます。**(config-service) string** コマンドを使用して、サービス ID を設定します。

（注）　**advanced-balance** コマンドの **arrowpoint-cookie** オプションを使用している場合は、文字列一致基準、**sticky-no-cookie-found-action** コマンド、**sticky-serverdown-failover** コマンドのいずれも設定しないでください。これらのコマンドは、**advanced-balance arrowpoint-cookie** コマンドとは互換性がありません。

文字列の一致基準を設定する必要はありません。arrowpoint で生成されたクッキーの設定については、「ArrowPoint クッキーの設定」を参照してください。このオプションは、どのレイヤ 5 コンテンツ ルールでも使用できます。

（注）　**advanced-balance arrowpoint-cookie** コマンドを使用してコンテンツ ルールに ArrowPoint クッキーを設定する際に、CSS が固定 HTTP 接続で ArrowPoint クッキーなしの次の GET を受信した場合には、CSS は実行設定内の持続性の設定をすべて無視し、バックエンド接続を新しいサーバに再マップしてから新しい ArrowPoint クッキーを挿入します。

たとえば、**advanced-balance arrowpoint-cookie** をコンテンツ rule1 に指定するには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# advanced-balance
arrowpoint-cookie
```

拡張ロード バランシング方式を無効にするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no advanced-balance
```

ArrowPoint クッキーの設定

CSS は、クライアントに対して ArrowPoint クッキーを透過的に生成し、クライアントはそのクッキーを保存して、次の要求で返信します。CSS は後からそのクッキーを使用して、クライアントとサーバ間のスティッキ性を維持します。このクッキーにはスティッキ情報そのものが含まれており、クッキーはスティッキ テーブルを参照しません。

arrowpoint-cookie コマンドを使用して、ArrowPoint クッキーの名前、パス、および有効期限を設定します。コンテンツ ルールに **arrowpoint-cookie** 方式を設定した場合、CSS はクライアント要求を受け取るたびに、ArrowPoint クッキーの存在の有無を確認します。このクッキーが存在しない場合は、サーバ ロード バランシングが実行され、ArrowPoint クッキーが生成されます。

クッキーがクライアント要求に見つかった場合は、クッキー データが解読され、検証されます。次に、クッキーの有効期限が確認されます。クッキーの有効期限が切れている場合は、クライアントが固定されているサーバについての情報を含む新しいクッキーが送信されます。これは、接続を中断せずに行われます。

クッキー フォーマットが有効な場合は、クッキーと CSS 設定の一貫性が確認されます。すべての検証にパスした場合、サーバ ID で指定されたサーバにクライアント要求が転送されます。パスしない場合、この要求は最初の要求として処理されます。

このコンテンツ モード コマンドのオプションは次のとおりです。

- **arrrowpoint-cookie name** ：一意のクッキー ID を指定する。
- **arrowpoint-cookie path** ：クッキー パスを、設定されたパスに設定する。
- **arrowpoint-cookie expiration** ：有効期限を設定する。CSS は、この有効期限と、クッキーに関連付けられている時間を比較します。

（注）　**arrowpoint-cookie expiration** コマンドと **advanced-balance arrowpoint-cookie** コマンドを設定すると CSS の CPU 使用率が急激に上昇し、パフォーマンスの低下につながる恐れがあります。**arrowpoint-cookie expiration** コマンドの設定は、必要な場合だけに限定してください。

- **arrowpoint-cookie browser-expire** ：ブラウザでクッキーを有効期限切れにできるようにする。
- **arrowpoint-cookie expire services** ：クッキーの有効期限が切れたときにサービス情報も無効にする。

arrowpoint クッキー名の設定

arrowpoint-cookie name コマンドを使用すると、最大 4 文字の英数字による一意のクッキー ID を設定できます。このオプションを使用すると、クッキーを挿入する複数の CSS を設定した場合に、1 つの CSS が別の CSS クッキーを上書きすることを防止できます。

この所有者コンテンツ設定モードのコマンドのシンタックスは次のとおりです。

arrowpoint-cookie name *name*

変数 *name* には、1 ～ 31 文字の英数字からなる一意の文字列を指定します。デフォルトは ARPT です。

注意　コンテンツ ルールに新しいクッキー名を設定すると、CSS はそのルールで設定されていたすべての既存のクッキー名を認識できなくなります。そのため、既存のスティッキ性はすべて失われます。

次に設定例を示します。

```
(config-owner-content[arrowpoint-rule1])# arrowpoint-cookie name
abc5678
```

このコマンドの結果として生成されるクッキーは、次のようになります。

```
abc5678=000000SservicenameT0x_0xC1234
```

Arrowpoint クッキー名をデフォルトの ARPT に戻すには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no arrowpoint-cookie name
```

arrowpoint クッキー パスの設定

デフォルトでは、クッキーのデフォルト パスのアトリビュートはスラッシュ（/）に設定されます。**arrowpoint-cookie path** コマンドを使用すると、arrowpoint クッキー パスを、設定済みのパスにすることができます。

この所有者コンテンツ設定モードのコマンドのシンタックスは次のとおりです。

arrowpoint-cookie path "*path_name***"**

path_name にはクッキーを送信する場所を入力します。99 文字以内のテキスト文字列を引用符で囲んで入力します。クッキーのデフォルト パスは、/ です。

たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# arrowpoint-cookie path
"/cgi-bin/"
```

クッキー パスを、デフォルトの / にリセットするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no arrowpoint-cookie path
```

arrowpoint クッキーの有効期限の設定

arrowpoint クッキーの有効期限が切れている場合は、クライアントの固定先のサーバを含む新しいクッキーが送信されます。新しいクッキーを送信しても、接続は中断されません。有効期限を設定しない場合、クッキーはクライアントがブラウザを終了すると同時に期限が切れます。

arrowpoint-cookie expiration コマンドを使用すると、有効期限を設定して、arrowpoint クッキーに関連付けられている時間と比較することができます。この所有者コンテンツ設定モード コマンドのシンタックスは次のとおりです。

arrowpoint-cookie expiration *dd:hh:mm:ss*

変数の内容は次のとおりです。

- *dd* ：日数。有効な数値は 00 ～ 99 です。
- *hh* ：時間数。有効な数値は 00 ～ 99 です。
- *mm* ：分数。有効な数値は 00 ～ 99 です。
- *ss* ：秒数。有効な数値は 00 ～ 99 です。

（注）　日数、時間数、分数および秒数のすべてに 0 を指定しないでください。この値は無効です。

たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# arrowpoint-cookie expiration
08:04:03:06
```

クライアントがブラウザを終了する時間に有効期限が切れるようにリセットするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no arrowpoint-cookie
expiration
```

arrowpoint クッキーのブラウザでの有効期限の設定

arrowpoint-cookie browser-expire コマンドを使用すると、設定されている有効期限に基づいて、ブラウザに arrowpoint クッキーを失効させることができます。有効期限の設定方法については、前の項を参照してください。この所有者コンテンツ設定モードのコマンドのシンタックスは次のとおりです。

arrowpoint-cookie browser-expire

次にコマンドの使用例を示します。

```
(config-owner-content[arrowpoint-rule1])# arrowpoint-cookie
browser-expire
```

CSS でのクッキーの有効期限を有効にするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no arrowpoint-cookie
browser-expire
```

（注）　クッキーの有効期限が切れるとすべてのスティッキ情報が消滅します。

arrowpoint クッキーの有効期限切れサービスの設定

デフォルトでは、arrowpoint クッキーの期限が切れると、CSS は期限の切れたクッキー内のサーバ情報を新しいクッキーに入れて送信します。

arrowpoint-cookie expire-services コマンドを使用すると、新しいクッキーを送信する前にそのクッキーの期限が切れたときに、サービス情報の期限も切れるようにすることができます。この所有者コンテンツ設定モードのコマンドのシンタックスは次のとおりです。

arrowpoint-cookie expire-services

たとえば、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# arrowpoint-cookie
expire-services
```

デフォルトの動作にリセットするには、次のように入力します。

```
(config-owner-content[arrowpoint-rule1])# no arrowpoint-cookie
expire-services
```

arrowpoint クッキー ドメインの設定

cookie-domain コマンドを使用して、arrowpoint クッキーのドメイン名を設定します。詳細については、この章で後述する「ロケーション クッキーのドメイン名の設定」を参照してください。

ロケーション クッキーの設定

複数サイトのネットワーク構成でクライアントが特定の CSS とサーバに固定されている場合、後続の DNS 要求が別の IP アドレスに解決され、クライアントが別の CSS とサーバに送信されることがあります。このような異なる解決は、電子商取引アプリケーションで、特に顧客がショッピング カートにすでに商品を入れている場合に問題となる可能性があります。複数のサイトを使用する環境で、クライアントを元の CSS に戻すには、ロケーション クッキーを設定します。

（注）　CSS は、ロケーション クッキーを含むコンテンツ要求を、スティッキ セッションの一部として認識します。したがって、重みを 0 に設定したサービスをロケーション サービスとしてコンテンツ ルールに追加しても、そのサービスから発信されたロケーション クッキーを含むすべての要求は、引き続きそのサービスに送信されます。

概要

ロケーション クッキーは、ユーザ定義の NAME=VALUE クッキー ペア文字列（サイト固有の CSS で設定）を、サーバからの応答パケットに挿入します。クッキーを挿入した後の接続では、新しい DNS 解決によってクライアントが別の CSS に送信されると、新しい CSS は NAME=VALUE ペアを、設定済みのコンテンツ ルール内の値と照合します。NAME=VALUE ペアに一致する値が見つからない場合、CSS はクッキーの VALUE 部分を、コンテンツ ルールで設定されているロケーション サービス内の情報と比較します。CSS は、ロケーション サービスの情報を使用して、クライアントを元のサイトに戻します。

クライアントを元のサイトに戻すには、CSS で設定したロケーション サービスのタイプに応じて、2 種類の方式があります。1 つ目の方式では、標準のサービス（パススルー サービス）を使用して、すべてのトラフィックを現在の CSS に通過させ、元の CSS に戻します。元の CSS から戻るトラフィックをすべて、クライアントに送信する前に現在の CSS で必ず NAT 変換するには、この同じサービスをソース グループ内の宛先サービスとして設定する必要があります。

2 つ目の方式では、リダイレクト サービスを使用してクライアントに 302 リダイレクトを送信することにより、クライアントを元のサイトに戻します。サービス設定モードの **no prepend http** コマンドを設定しない限り、302 リダイレクトでは URL PUT を使用して http:// を付与した URL を追加します。URL は特定のファイル、またはディレクトリ内のデフォルトファイルを示します。リダイレクトされたサービスは、サービス設定モードのドメイン コマンドを使用して HTTP から HTTPS へリダイレクトすることもできます。302 リダイレクトは、すべての HTTP 方式を HTTP GET に変更することに注意してください。たとえば、あるクライアントが POST 操作を実行しており、現在の CSS がそのクライアントを元のサイトにリダイレクトすると、クライアントは POST 方式内のすべてのデータを失います。

（注）　各サイトは、主サイトの IP アドレスだけでなく、たとえば site1.work.com などの一意な DNS エントリも持つ必要があります。サイトがこのような固有の DNS エントリを持つことで、クライアントは各サイトについて DNS サーバから一意の応答を受け取ることができ、IP アドレスにリダイレクトされることがなくなります。

ロケーション クッキーのクイック スタート

表 11-2、表 11-3、および表 11-4 に、複数サイト設定で各 CSS にロケーション クッキー機能を設定するのに必要な手順の概要を示します。ここでは、3 つのサンプル サイトについてクイック スタート手順を示します。それぞれの手順に、作業を実行するために必要な CLI コマンドも示します。CLI コマンドに関する各機能とすべてのオプションの詳細については、表の後に示す各項を参照してください。

表 11-2　Site1 に対するロケーション クッキー設定のクイック スタート

作業とコマンドの例

1. グローバル設定モードに移行します。
`# config`
`(config)#`

2. ローカル サービスを必要に応じて設定します。サービス設定の詳細については、第 3 章「サービスの設定」を参照してください。次に設定例を示します。
`(config)# service localServ1`
`(config-service[localServ1])# ip address 192.168.2.3`
`(config-service[localServ1])# active`
`(config-service[localServ1])# service localServ2`
`(config-service[localServ2])# ip address 192.168.2.4`
`(config-service[localServ2])# active`

3. CSS がロケーション サービスとして使用するリダイレクト サービスを設定します。
`(config-service[localServ2])# service site2`
`(config-service[site2])# ip address 192.158.128.209`
`(config-service[site2])# string site2`
`(config-service[site2])# type redirect`
`(config-service[site2])# active`

4. CSS がロケーション サービスとして使用する標準サービスを設定します。
`(config-service[site2])# service site3`
`(config-service[site3])# ip address 192.148.128.209`
`(config-service[site3])# string site3`
`(config-service[site3])# active`

表 11-2　Site1 に対するロケーション クッキー設定のクイック スタート（続き）

作業とコマンドの例
5. 所有者を設定します。所有者設定の詳細については、第 9 章「所有者の設定」を参照してください。 (config)# **owner ArrowPoint** (config-owner[ArrowPoint])#
6. この所有者のコンテンツ ルールを作成します。コンテンツ ルール設定の詳細については、第 10 章「コンテンツ ルールの設定」を参照してください。 (config-owner[ArrowPoint])# **content locCookie** (config-owner-content[ArrowPoint-locCookie])#
7. コンテンツ ルールで次のコマンドを設定します。 (config-owner-content[ArrowPoint-locCookie])# **vip address 192.168.128.209** (config-owner-content[ArrowPoint-locCookie])# **add service localServ1** (config-owner-content[ArrowPoint-locCookie])# **add service localServ2** (config-owner-content[ArrowPoint-locCookie])# **protocol tcp** (config-owner-content[ArrowPoint-locCookie])# **port 80** (config-owner-content[ArrowPoint-locCookie])# **url "/*"** (config-owner-content[ArrowPoint-locCookie])# **location-cookie name work value site1** (config-owner-content[ArrowPoint-locCookie])# **cookie-domain ".work.com"** (config-owner-content[ArrowPoint-locCookie])# **add location-service site2** (config-owner-content[ArrowPoint-locCookie])# **add location-service site3** (config-owner-content[ArrowPoint-locCookie])# **active**
8. ソース グループを作成し、サービス site3 を宛先サービスとして追加します。ソース グループ設定の詳細については、第 3 章「サービスの設定」を参照してください。 (config)# **group site1** (config-group[site1])# **add destination service site3** (config-group[site1])# **vip address 192.168.128.210** (config-group[site1])# **active**
9. **show rule sticky** コマンドを使用して、ロケーション クッキーの設定を確認します。 # **show rule sticky**

次の running-config の例は、表 11-2 内の各コマンドを site 1 を対象に実行した結果を示しています。

```
!************************** SERVICE **************************
service localServ1
  ip address 192.168.2.3
  active

service localServ2
  ip address 192.168.2.4
  active

service site2
  ip address 192.158.128.209
  string site2
  type redirect
  active

service site3
  ip address 192.148.128.209
  string site3
  active

!************************** OWNER ****************************
owner ArrowPoint

  content locCookie
    vip address 192.168.128.209
    add service localServ1
    add service localServ2
    protocol tcp
    port 80
    url "/*"
    location-cookie name work value site1
    cookie-domain ".work.com"
    add location-service site2
    add location-service site3
    active

!************************** GROUP ****************************
group site1
  add destination service site3
  vip address 192.168.128.210
  active
```

表 11-3　Site2 に対するロケーション クッキー設定のクイック スタート

作業とコマンドの例

1. グローバル設定モードに移行します。
`# config`
`(config)#`

2. ローカル サービスを必要に応じて設定します。次に設定例を示します。
`(config)# service localServ1`
`(config-service[localServ1])# ip address 192.158.2.3`
`(config-service[localServ1])# active`
`(config-service[localServ1])# service localServ2`
`(config-service[localServ2])# ip address 192.158.2.4`
`(config-service[localServ2])# active`

3. 他の各サイトに対して CSS がロケーション サービスとして使用する標準サービスを設定します。
`(config-service[localServ2])# service site1`
`(config-service[site1])# ip address 192.168.128.209`
`(config-service[site1])# string site1`
`(config-service[site1])# active`
`(config-service[site1])# service site3`
`(config-service[site3])# ip address 192.148.128.209`
`(config-service[site3])# string site3`
`(config-service[site3])# active`

4. 所有者を設定します。
`(config)# owner ArrowPoint`
`(config-owner[ArrowPoint])#`

5. この所有者のコンテンツ ルールを作成します。
`(config-owner[ArrowPoint])# content locCookie`
`(config-owner-content[ArrowPoint-locCookie])#`

表 11-3　Site2 に対するロケーション クッキー設定のクイック スタート（続き）

作業とコマンドの例
6. コンテンツ ルールで次のコマンドを設定します。 (config-owner-content[ArrowPoint-locCookie])# **vip address 192.158.128.209** (config-owner-content[ArrowPoint-locCookie])# **add service localServ1** (config-owner-content[ArrowPoint-locCookie])# **add service localServ2** (config-owner-content[ArrowPoint-locCookie])# **protocol tcp** (config-owner-content[ArrowPoint-locCookie])# **port 80** (config-owner-content[ArrowPoint-locCookie])# **url "/*"** (config-owner-content[ArrowPoint-locCookie])# **location-cookie name work value site2** (config-owner-content[ArrowPoint-locCookie])# **cookie-domain ".work.com"** (config-owner-content[ArrowPoint-locCookie])# **add location-service site1** (config-owner-content[ArrowPoint-locCookie])# **add location-service site3** (config-owner-content[ArrowPoint-locCookie])# **active**
7. ソース グループを作成し、サービス site1 およびサービス site3 を宛先サービスとして追加します。 (config)# **group site2** (config-group[site2])# **add destination service site1** (config-group[site2])# **add destination service site3** (config-group[site2])# **vip address 192.158.128.210** (config-group[site2])# **active**
8. **show rule sticky** コマンドを使用して、ロケーション クッキーの設定を確認します。 # **show rule sticky**

次の running-config の例は、表 11-3 内の各コマンドを site 2 を対象に実行した結果を示しています。

```
!************************** SERVICE **************************
service localServ1
  ip address 192.158.2.3
  active

service localServ2
  ip address 192.158.2.4
  active

service site1
  ip address 192.168.128.209
  string site1
  active

service site3
  ip address 192.148.128.209
  string site3
  active

!************************** OWNER ****************************
owner ArrowPoint

  content locCookie
    vip address 192.158.128.209
    add service localServ1
    add service localServ2
    protocol tcp
    port 80
    url "/*"
    location-cookie name work value site2
    cookie-domain ".work.com"
    add location-service site1
    add location-service site3
    active

!************************** GROUP ****************************
group site2
  add destination service site1
  add destination service site3
  vip address 192.158.128.210
  active
```

表 11-4　Site3 に対するロケーション クッキー設定のクイック スタート

作業とコマンドの例

1. グローバル設定モードに移行します。
`# config`
`(config)#`

2. ローカル サービスを必要に応じて設定します。次に設定例を示します。
`(config)# service localServ1`
`(config-service[localServ1])# ip address 192.148.2.3`
`(config-service[localServ1])# active`
`(config-service[localServ1])# service localServ2`
`(config-service[localServ2])# ip address 192.148.2.4`
`(config-service[localServ2])# active`

3. CSS がロケーション サービスとして使用する標準サービスを設定します。
`(config-service[localServ2])# service site1`
`(config-service[site1])# ip address 192.168.128.209`
`(config-service[site1])# string site1`
`(config-service[site1])# active`

4. CSS がロケーション サービスとして使用するリダイレクト サービスを設定します。
`(config-service[site1])# service site2`
`(config-service[site2])# ip address 192.158.128.209`
`(config-service[site2])# string site2`
`(config-service[site2])# type redirect`
`(config-service[site2])# active`

5. 所有者を設定します。
`(config)# owner ArrowPoint`
`(config-owner[ArrowPoint])#`

6. この所有者のコンテンツ ルールを作成します。
`(config-owner[ArrowPoint])# content locCookie`
`(config-owner-content[ArrowPoint-locCookie])#`

表 11-4　Site3 に対するロケーション クッキー設定のクイック スタート（続き）

作業とコマンドの例
7. コンテンツ ルールで次のコマンドを設定します。 (config-owner-content[ArrowPoint-locCookie])# **vip address 192.148.128.209** (config-owner-content[ArrowPoint-locCookie])# **add service localServ1** (config-owner-content[ArrowPoint-locCookie])# **add service localServ2** (config-owner-content[ArrowPoint-locCookie])# **protocol tcp** (config-owner-content[ArrowPoint-locCookie])# **port 80** (config-owner-content[ArrowPoint-locCookie])# **url "/*"** (config-owner-content[ArrowPoint-locCookie])# **location-cookie name work value site3** (config-owner-content[ArrowPoint-locCookie])# **cookie-domain ".work.com"** (config-owner-content[ArrowPoint-locCookie])# **add location-service site1** (config-owner-content[ArrowPoint-locCookie])# **add location-service site2** (config-owner-content[ArrowPoint-locCookie])# **active**
8. ソース グループを作成し、サービス site1 を宛先サービスとして追加します。 (config)# **group site3** (config-group[site1])# **add destination service site1** (config-group[site1])# **vip address 192.148.128.210** (config-group[site1])# **active**
9. **show rule sticky** コマンドを使用して、ロケーション クッキーの設定を確認します。 # **show rule sticky**

次の running-config の例は、表 11-4 内の各コマンドを site 3 を対象に実行した結果を示しています。

```
!************************** SERVICE **************************
service localServ1
  ip address 192.148.2.3
  active

service localServ2
  ip address 192.148.2.4
  active
service site1
  ip address 192.168.128.209
  string site1
  active

service site2
  ip address 192.158.128.209
  string site2
  type redirect
  active

!************************** OWNER ****************************
owner ArrowPoint

  content locCookie
    vip address 192.148.128.209
    add service localServ1
    add service localServ2
    protocol tcp
    port 80
    url "/*"
    location-cookie name work value site3
    cookie-domain ".work.com"
    add location-service site1
    add location-service site2
    active

!************************** GROUP ****************************
group site3
  add destination service site1
  vip address 192.148.128.210
  active
```

location-cookie コマンドの設定

ローカル サイトの NAME=VALUE クッキー文字列と有効期限を設定するには、**location-cookie** コマンドを使用します。CSS では、このクッキー文字列をサーバからの応答に挿入します。

（注）　ロケーション クッキーには、レイヤ 5（L5）コンテンツ ルールが必要です。「ロケーション クッキーのクイック スタート」で説明したように、ルールには少なくとも url "/*" を設定する必要があります。

この所有者コンテンツ設定モードのコマンドのシンタックスは次のとおりです。

location-cookie name *text* **value** *text* {**expiration** *dd:hh:mm:ss*}

このコマンドのオプションと変数は次のとおりです。

- **name** *text* ：NAME=VALUE クッキー文字列の最初の部分。1 ～ 31 文字のテキスト文字列を引用符で囲まずに入力します。
- **value** *text* ：NAME=VALUE クッキー文字列の 2 番目の部分。1 ～ 31 文字のテキスト文字列を引用符で囲まずに入力します。
- **expiration** *dd:hh:mm:ss* ：（オプション）ロケーション クッキーの有効期限の日付と時刻。この値はクライアント ブラウザに対して、クッキーの有効期限が切れる日時を、クッキーの生成時点からの相対時間で示します。日付と時刻を、次の形式で入力します。
  - *dd* ：日数。有効な数値は 00 ～ 99 です。
  - *hh* ：時間数。有効な数値は 00 ～ 99 です。
  - *mm* ：分数。有効な数値は 00 ～ 99 です。
  - *ss* ：秒数。有効な数値は 00 ～ 99 です。

（注）　ロケーション クッキーの有効期限の日時を設定すると CSS の CPU 使用率が急激に上昇し、パフォーマンスの低下につながる恐れがあります。**expiration** オプションの設定は、必要な場合だけに限定してください。

次に設定例を示します。

```
(config-owner-content[ArrowPoint-rule1])# location-cookie name work
value site1 00:02:30:00
```

ロケーション クッキーを削除するには、次のように入力します。

```
(config-owner-content[ArrowPoint-rule1])# no location-cookie name
```

ロケーション クッキーのドメイン名の設定

クッキー ドメイン名を設定することで、ブラウザは、指定したドメイン名で終わるサイトにクッキーを返送できるようになります。たとえば .work.com というクッキー ドメイン名を指定すると、ブラウザはロケーション クッキーを、site1.work.com、site2.work.com、site3.work.com など、.work.com で終わるすべてのサイトに返送します。

cookie-domain コマンドを使用して、ロケーション クッキーのドメイン名を設定します。この所有者コンテンツ設定モードのコマンドのシンタックスは次のとおりです。

cookie-domain *name*

name 変数では、ロケーション クッキーのドメインの名前を指定します。1 ～ 64 文字のテキスト文字列を引用符で囲んで入力します。

たとえば、次のようになります。

```
(config-owner-content[ArrowPoint-rule1)# cookie-domain ".work.com"
```

クッキー ドメイン名を削除するには、次のように入力します。

```
(config-owner-content[ArrowPoint-rule1)# no cookie-domain
```

ロケーション サービスの設定

クライアントが最初に固定されていたサイトを検索する際に、CSS が使用するコンテンツ ルールにサービスを追加するには、**add location-service** コマンドを使用します。

この所有者コンテンツ設定モードのコマンドのシンタックスは次のとおりです。

add location-service *service_name*

name 変数では、ロケーション クッキーを照合するためのコンテンツ ルールに追加する、標準サービスまたはリダイレクト サービスの名前を指定します。1 ～ 31 文字の名前を入力します。

たとえば、次のように入力します。

```
(config-owner-content[ArrowPoint-rule1)# add location-service site1
```

site1 ロケーション サービスを削除するには、次のように入力します。

```
(config-owner-content[ArrowPoint-rule1)# remove location-service site1
```

ロケーション サービスは、標準サービスまたはリダイレクト サービスとして、最大で 10 件まで設定できます。これらのサービスは、コンテンツ ルールあたりのサービスの最大数である 64 には含まれません。また、コンテンツ ルールのロード バランシング アルゴリズムには関与しません。

（注）　ロケーション サービスが設定されているコンテンツ ルールに、同じクッキー文字列でロケーション サービスを追加することはできません。

ロケーション サービスとして設定したリダイレクト サービスには、リダイレクトされるサイトで設定されているロケーション クッキー コンテンツ ルールの VIP アドレスと同じ IP アドレスを指定する必要があります。また、ロケーション サービスとして使用するすべてのサービスに、文字列を定義する必要があります。この文字列は、ロケーション クッキーの VALUE 部分と一致している必要があります。サービスに対する文字列の設定の詳細については、「拡張ロード バランシング文字列の設定」を参照してください。

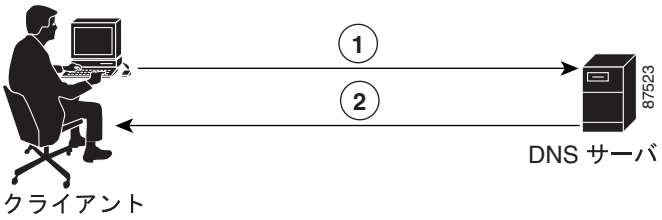
また、コンテンツ ルールでロケーション サービスとして設定した標準サービス（「ロケーション クッキーのクイック スタート」を参照）を、ソース グループ内で宛先サービスとして設定する必要があります。

ロケーション クッキーのフローの例

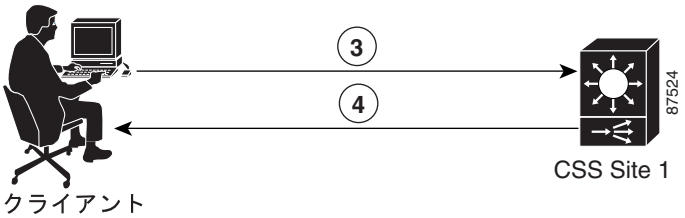
次の例では、元の要求を処理した CSS とサーバにクライアントを戻すために CSS が使用する、2 つのメカニズムを説明します。CSS サイトの具体的な設定情報については、「ロケーション クッキーのクイック スタート」を参照してください。

例 1 ：ロケーション サービスを使用してクライアントを元のサイトに戻す方式（パススルー方式）

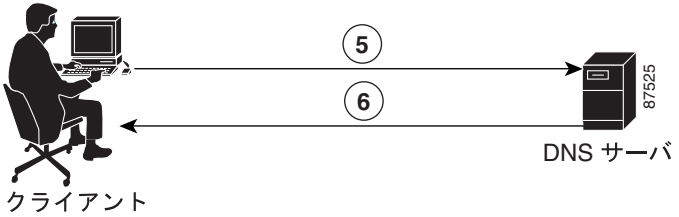
1. クライアントは、my.work.com のルックアップ要求を DNS サーバに送信します。
2. DNS サーバは、応答として 192.168.128.209 という CSS Site1 の VIP アドレスを返送します。



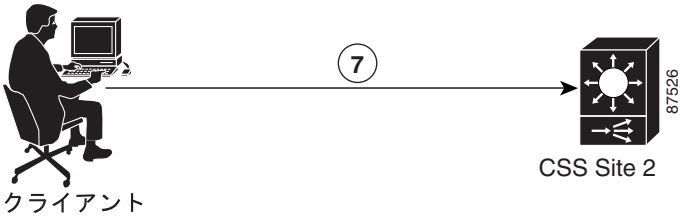
3. クライアントは、192.168.128.209 に GET 要求をクッキーなしで送信します。
4. CSS は、ロケーション クッキー work=site1 を、ローカル サーバからの応答に挿入します。クライアントとサーバの相互対話は通常どおり実行されます。



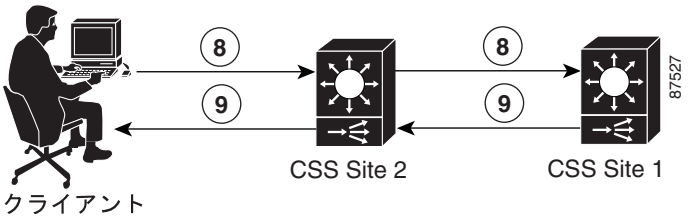
5. しばらくして、クライアントは my.work.com のルックアップ要求をもう一度 DNS サーバに送信します。
6. DNS サーバは、今回は応答として 192.158.128.209 という CSS Site2 の VIP アドレスを返送します。



7. クライアントは、192.158.128.209（CSS Site2）に GET 要求を、work=site1 というロケーション クッキーと共に送信します。このクッキー文字列は、設定されているロケーション クッキー名（work）とは一致しますが、クッキー値（site2）とは一致しません。CSS Site2 は、ロケーション サービスのリストを検索し、設定されている文字列をクッキーの値と照合します。この場合、サービス site1 が一致します。

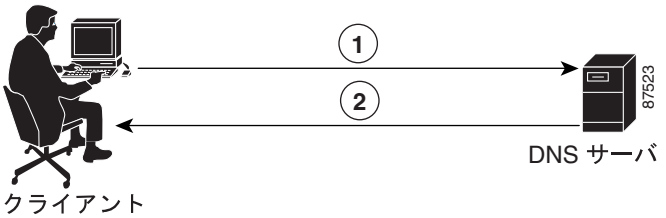


8. サービス site1 は、ソース グループ site2 で宛先サービスとして設定されています。このサービスは、CSS Site2 で設定されている locCookie コンテンツ ルールに一致します。CSS Site2 は、パケットをクライアントから CSS Site1 に転送します。
9. クライアントは、すでにステップ 7 でクッキーを GET 要求と共に送信しているため、別のクッキーを挿入する必要はありません。コンテンツ ルールの処理は、変更なく継続されます。サーバの応答は、CSS Site1 から CSS Site2 を通じてクライアントに返されます。クライアントは、CSS Site2 を通じて元のサイト（site1.work.com）に固定されます。

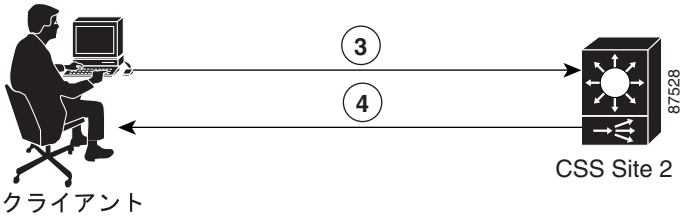


例 2 ：リダイレクト サービスを使用してクライアントを元のサイトに戻す方式

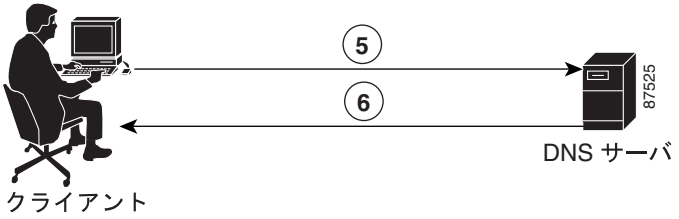
1. クライアントは、my.work.com のルックアップ要求を DNS サーバに送信します。
2. DNS サーバは、応答として 192.158.128.209 という CSS Site2 の VIP アドレスを返送します。



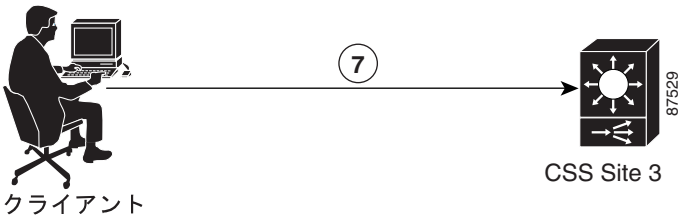
3. クライアントは、192.158.128.209（Site2）に GET 要求をクッキーなしで送信します。
4. CSS は、ロケーション クッキー work=site2 を、ローカル サーバからの応答に挿入します。クライアントとサーバの相互対話は通常どおり実行されます。



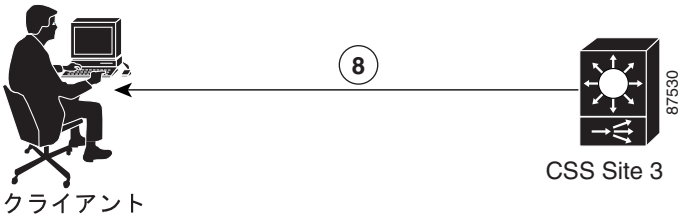
5. しばらくして、クライアントは my.work.com のルックアップ要求をもう一度 DNS サーバに送信します。
6. DNS サーバは、今回は応答として 192.148.128.209 という CSS Site3 の VIP アドレスを返送します。



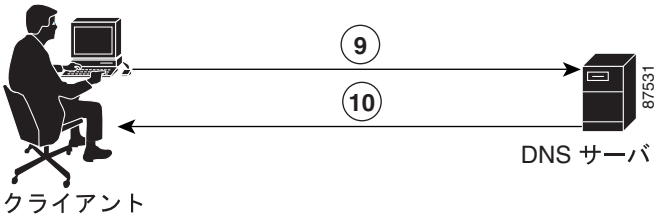
7. クライアントは、192.148.128.209（CSS Site3）に GET 要求を、work=site2 というロケーション クッキーと共に送信します。このクッキー文字列は、CSS Site3 で設定されているロケーション クッキー名（work）とは一致しますが、クッキー値（site3）とは一致しません。CSS Site3 は、ロケーション サービスのリストを検索し、設定されている文字列をクッキーの値と照合します。この場合、サービス site2 が一致します。



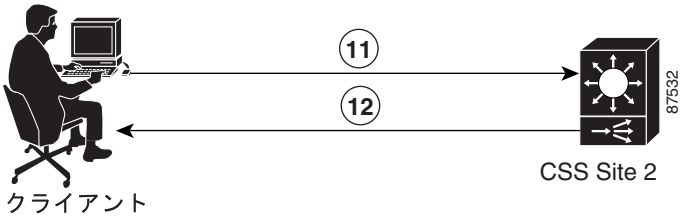
8. サービス site2 はリダイレクト サービスであるため、CSS Site3 はクライアントに、site2.work.com への 302 リダイレクトを送信します。



9. クライアントは、site2.work.com の検索要求を DNS サーバに送信します。
10. DNS サーバは、応答として 192.158.128.209 という CSS Site2 の VIP アドレスを返送します。



11. クライアントは、192.158.128.209（CSS Site2）に GET 要求を、work=site2 というクッキーと共に送信します。
12. クッキーの名前と値が、CSS Site2 で設定されている locCookie コンテンツ ルールに一致します。クライアントは、すでにステップ 11 でクッキーを GET 要求と共に送信しているため、別のクッキーを挿入する必要はありません。コンテンツ ルールの処理は、変更なく継続されます。クライアントは、元のサイト（site2.work.com）に永続的に固定されます。



ロケーション クッキー情報の表示

ロケーション クッキー情報を表示するには、**show rule sticky** コマンドを使用します。詳細については、「スティッキ アトリビュートの表示」を参照してください。

電子商取引アプリケーションを利用する無線ユーザの設定

無線を利用するクライアントは、インターネットのコンテンツにアクセスするのに Wireless Application Protocol（WAP）を使用します。無線を利用するクライアントがコンテンツを要求すると、WAP プロトコル ゲートウェイ（要求を WAP プロトコル スタックから WWW プロトコル スタックに変換する装置）が、MSISDN フィールドを生成して HTTP ヘッダーに追加します。

MSISDN ヘッダー フィールドと、**advanced-balance wap-msisdn** コマンドを使用すると、無線ユーザが電子商取引アプリケーションを利用するように設定することができます。**advanced-balance wap-msisdn** コマンドの詳細については、この章で前述した「スティッキ コンテンツへの拡張ロード バランシング方式の指定」を参照してください。MSISDN ヘッダー フィールドの詳細については、第 12 章「HTTP ヘッダー ロード バランシングの設定」の「ヘッダー フィールド エントリの設定」を参照してください。

次の例では、VIP 192.168.128.151 宛ての TCP ポート 80 のトラフィックで、HTTP ヘッダー フィールドに文字列「012」があるものは、すべてコンテンツ ルール rule012 に一致します。CSS は、このトラフィックを MSISDN フィールドのすべての内容に基づいて、server1 か server2 のいずれかに固定します。

VIP 192.168.128.151 宛ての TCP ポート 80 のトラフィックで、HTTP ヘッダー フィールドに文字列「012」がないものは、すべてコンテンツ ルール ruleNo012 に一致します。CSS は、トラフィックを server21 と server22 にラウンドロビン方式で負荷分散します。

VIP 192.168.128.151 宛ての TCP ポート 80 のトラフィックで、MSISDN HTTP ヘッダー フィールドがないものは、すべてコンテンツ ルール ruleNoWap に一致します。CSS は、トラフィックを server31 と server32 にラウンドロビン方式で負荷分散します。

```
header-field-group wap012
   header-field 1 wap-msisdn contain "012"

header-field-group wapNo012
   header-field 1 wap-msisdn not-contain "012"

owner arrowpoint
   content rule012
     vip address 192.168.128.151
     protocol tcp
     port 80
     url "/*"
     add service server1
     add service server2
     header-field-rule wap012
     advanced-balance wap-msisdn
     active
   content ruleNo012
     vip address 192.168.128.151
     protocol tcp
     port 80
     url "/*"
     add service server21
     add service server22
     header-field-rule wapNo012
     active

   content ruleNoWap
     vip address 192.168.128.151
     protocol tcp
     port 80
     url "/*"
     add service server31
     add service server32
     active
```

Session Initiation Protocol ロード バランシングの設定

Session Initiation Protocol（SIP）はアプリケーション層制御プロトコルで、ユーザ装置とメディア サーバ間のシグナル メカニズムとして機能します。SIP はピアツーピア プロトコルです。エンド デバイス（ユーザ エージェント クライアント）は、このプロトコルを使用して SIP サーバとの対話型通信セッションを起動します。これらの通信セッションには、インターネット マルチメディア会議、インターネット電話（VoIP）、およびマルチメディア配信があります。クライアント装置の例としては、ハードウェア、ソフトウェア、携帯 IP 電話、および personal digital assistant（PDA; 携帯情報端末）があります。

セッション Call-ID は一意の呼び出し識別子で、クライアントから SIP プロキシ サーバに送信された SIP メッセージに含まれます。Call-ID によるスティッキ性は、Call-ID を使用して現在の SIP セッションを特定し、メッセージの内容に基づいて決定を行うコール ステートフル サービスで特に重要です。

クライアントから SIP サーバに送信された SIP メッセージの SIP Call-ID を検出すると、CSS は SIP Call-ID に基づいてキー（ハッシュ値）を生成します。CSS は、生成したキーを使用してスティッキ テーブル内のエントリを検索します。エントリがある場合、CSS はそのテーブル エントリに示されているスティッキ サーバにそのクライアントの要求を送ります。エントリがない場合には、CSS は新しいスティッキ エントリを作成し、SIP Call-ID 値をキーにハッシュした後にそのキーをエントリに保存します。

CSS では、次の SIP メソッドをサポートします。

- INVITE ：ユーザまたはサービスが SIP セッションに参加するように要求されていることを示す。
- ACK :クライアントが INVITE 要求に対する最終応答を受信したことを確認する。
- OPTIONS ：サーバの機能に対して問い合わせが行われる。
- BYE ：クライアントがコールを解放したいことをサーバに示す。
- CANCEL ：同じ Call-ID、To、From、および Cseq ヘッダー フィールド値を持つ保留要求を取り消す。ただし、完了済み要求には影響しません。
- REGISTER ：To ヘッダー フィールドに表示されたアドレスを SIP サーバに登録する。

設定要件と制限事項

CSS の SIP ロード バランシングの設定には、次の設定要件と制限事項が適用されます。

- CSS は UDP でだけ SIP をサポートする。
- SIP プロキシ サーバからの UDP 応答を CSS を介してクライアントに返して NAT 変換する場合は、ソース グループに送信先サービスを設定する。送信先サービスは、クライアントの IP アドレスを CSS VIP へ NAT 変換して、サーバからの応答パケットを CSS を介してクライアントに返します。
- **application sip** コンテンツ設定モード コマンドを入力すると、SIP ポートが 5060 に、プロトコルが UDP に自動的に設定される。SIP ポートを設定から削除すると、SIP コンテンツ ルールのアクティブ化が失敗し、その失敗の理由を示すメッセージが送信されます。SIP ポートが設定されていることを確認するには、**show rule** コマンドを入力します。
- **application sip** コマンドは **url** コマンド、および **url**、**urlhash**、**domain**、**domainhash** ロード バランシング方式とは互換性がない。
- デフォルトでは、SIP のフローが設定される。SIP フローを無効にするには、**flow-state 5060 udp flow-disable nat-enable** グローバル設定モード コマンドを使用します。SIP フローを無効にすると、同一フロー内で異なる Call-ID が使用できなくなります。スティッキ性は、SIP フローが有効になっている場合と同じように動作します。SIP フローを復元するには、**no flow-state 5060 udp** コマンドを入力します。

SIP ロード バランシングの設定のクイック スタート

表 11-5 に、複数サイト構成で各 CSS に SIP ロード バランシング機能を設定するのに必要な手順の概要を示します。それぞれの手順に、作業を実行するために必要な CLI コマンドも示します。CLI コマンドに関する各機能とすべてのオプションの詳細については、この表に続く各項を参照してください。

表 11-5　SIP 設定のクイック スタート

作業とコマンドの例
1. グローバル設定モードに移行します。 **# config** (config)#
2. SIP プロキシ サーバのサービスを設定します。サービス設定の詳細については、第 3 章「サービスの設定」を参照してください。次に設定例を示します。 (config)# **service sipServer** (config-service[sipServer])# **ip address 192.168.2.3** (config-service[sipServer])# **active**
3. 所有者を設定します。所有者設定の詳細については、第 9 章「所有者の設定」を参照してください。 (config)# **owner sipOwner** (config-owner[sipOwner])#
4. この所有者の SIP コンテンツ ルールを作成します。コンテンツ ルール設定の詳細については、第 10 章「コンテンツ ルールの設定」を参照してください。 (config-owner[sipOwner])# **content sipRule** (config-owner-content[sipOwner-sipRule])#

表 11-5　SIP 設定のクイック スタート（続き）

作業とコマンドの例

5. コンテンツ ルールで次のコマンドを設定します。**application sip** コマンドについては、第 10 章「コンテンツ ルールの設定」の「アプリケーション タイプの指定」を参照してください。**advanced-balance sip-call-id** コマンドについては、この章で前述した「スティッキ コンテンツへの拡張ロード バランシング方式の指定」を参照してください。
 	
（注）　プロトコルとポートがまだ設定されていない場合、**application sip** コマンドを入力すると、プロトコルが UDP に、ポート番号が 5060 に自動的に設定されます。
 	
`(config-owner-content[sipOwner-sipRule])# vip address 192.168.128.191`
`(config-owner-content[sipOwner-sipRule])# application sip`
`(config-owner-content[sipOwner-sipRule])# advanced-balance sip-call-id`
`(config-owner-content[sipOwner-sipRule])# add service sipServer`
`(config-owner-content[sipOwner-sipRule])# active`

6. アプリケーションにとって、VIP（CSS）からのサービス応答をクライアントが受信する必要がある場合は、ソース グループを設定する必要があります。この場合、ソース グループは、「received=」フィールドを使用するようにサーバが設定されている場合にだけ動作します。そうでない場合には、この手順はスキップしてください。ソース グループ設定の詳細については、第 5 章「サービスのソース グループの設定」を参照してください。
 	
`(config)# group sipGroup`
`(config-group[sipGroup])# vip address 192.168.1.228`
`(config-group[sipGroup])# add destination service sipServer`
`(config-group[sipGroup])# active`

7. （推奨）SIP 設定を確認するには、**show rule sticky**、**show sticky-table all-sticky**、**show sticky-table call-id-sticky**、および **show sticky-stats** コマンドを使用します。
 	
`# show rule sticky`
`# show sticky-table all-sticky`
`# show sticky-table call-id-sticky`
`# show sticky-stats`

表 11-5 に示した各コマンドを実行すると、次のような実行設定が得られます。

```
!************************** CIRCUIT **************************
circuit VLAN1

  ip address 192.168.1.191 255.255.255.0
  ip address 192.168.2.191 255.255.254.0
!************************** SERVICE **************************
service sipServer
  ip address 192.168.2.3
  active

!************************** OWNER ****************************
owner sipOwner

  content sipRule
    vip address 192.168.128.191
    protocol udp
    port 5060
    application sip
    advanced-balance sip-call-id
    add service sipServer
    active
!************************** GROUP ****************************
group sipGroup
  vip address 192.168.1.228
  add destination service sipServer
  active
```

スティッキ アトリビュートの表示

コンテンツ ルールのスティッキ アトリビュートを表示するには、**sticky** オプションを指定して **show rule** コマンドを使用します。**show rule** コマンドのシンタックスは次のとおりです。

show rule {*owner_name* {*content_rule_name*
{**acl**|**all**|**dns**|**header-field**|**hot-list**|**services**|**statistics**|**sticky**}}}

このコマンドは、スーパーユーザ、ユーザ、グローバル設定モード、または他の設定モードで使用できます。

次にコマンドの使用例を示します。

```
(config)# show rule sipOwner sipRule sticky
```

たとえば、コンテンツ設定モードで、次のように入力します。

```
(config-owner-content[sipOwner-sipRule])# show rule sticky
```

表 11-6 に、**show rule sticky** コマンドで表示されるフィールドについて説明します。

表 11-6　show rule sticky コマンドの出力フィールド

フィールド	説明
Balance	コンテンツ ルールのロード バランシング アルゴリズム。次の値があります。 • **ACA** ：Arrowpoint Content Awareness（ACA）アルゴリズム。CSS はコンテンツ要求の頻度とサーバのキャッシュ サイズとの相関関係を使って、そのサーバのキャッシュ ヒット率を高めます。 • **destip** ：宛先 IP アドレスによる分配アルゴリズム。CSS は、同じ宛先 IP アドレスを持つすべてのクライアント要求を同じサービスに送ります。 • **domain** ：ドメイン名による分配アルゴリズム。CSS は要求 URI のドメイン名を使用して、クライアント要求を適切なサービスに送ります。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
Balance（続き）	• **domainhash** ：ドメイン文字列に基づく、CSS の内部ハッシュ アルゴリズム。CSS はこのアルゴリズムを使用して、ドメイン文字列全体に対してハッシュをとります。CSS はこのハッシュ結果を使用してサーバを選択します。 • **leastconn** ：最小接続数によるアルゴリズム。CSS は、実行中のサービスのうち、最も接続数が少ないサービスを選択します。 • **roundrobin** ：ラウンドロビン アルゴリズム（デフォルト）。 • **srcip** ：送信元 IP アドレスによる分配アルゴリズム。CSS は、同じ送信元 IP アドレスを持つすべてのクライアント要求を同じサービスに送ります。 • **url** ：URL による分配アルゴリズム。CSS は、リダイレクト URL でその URL（先頭のスラッシュは除外）を使用して、クライアント要求を適切なサービスに送ります。 • **urlhash** ：URL 文字列に基づく、CSS の内部ハッシュ アルゴリズム。CSS はこのアルゴリズムを使用して、URL 文字列全体に対してハッシュをとります。CSS はこのハッシュ結果を使用してサーバを選択します。 • **weightedrr** ：加重ラウンドロビン アルゴリズム。CSS は、ラウンドロビン アルゴリズムを使用しますが、サービスに設定された重みに基づいて、サービスに重み付けを行います。サービスをルールに追加するときに、サービスの重みを設定または変更できます。コンテンツ ルールで設定されたサービスの重みは、そのコンテンツ ルールのサービスで設定された重みだけを無効にします。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
Advanced Balance	スティッキ性を含む、コンテンツ ルールの高度なロード バランシング方式。次の値があります。 • **arrowpoint-cookie** ：コンテンツ ルールで、クッキー内にある、選択されたサーバの一意のサービス識別子情報に基づいて、クライアントをサーバに固定できるようにする。 • **cookies** ：コンテンツ ルールで、HTTP クッキー ヘッダー内にある設定済みの文字列に基づいて、クライアントをサーバに固定できるようにする。このオプションを使用する場合は、コンテンツ ルールでポートを指定する必要があります。これによって、CSS はその接続をスプーフします。 • **cookieurl** : **advanced-balance cookies** オプションと同じだが、HTTP パケット内でクッキー ヘッダーが見つからない場合、このタイプのフェールオーバーでは、同じ文字列基準に基づいて、URL 拡張子（つまり URL の「?」の後の部分）を検索する。このオプションは、すべてのレイヤ 5 HTTP コンテンツ ルールで使用できます。 • **none** ：ルールに対して高度なバランシング方式を無効にする。これがデフォルト設定です。 • **sip call-ID** ：Session Initiation Protocol（SIP）セッション Call-ID に基づいて、クライアントをサーバに固定できるようにコンテンツ ルールを設定する。コンテンツ ルールでアプリケーション タイプは SIP、プロトコルは UDP とする必要があります。 • **sticky-srcip** ：コンテンツ ルールで、クライアントの IP アドレスに基づいてクライアントをサーバに固定できるようにする。これは、レイヤ 3 のスティッキ性とも呼ばれます。このオプションは、レイヤ 3、4、または 5 のコンテンツ ルールで使用できます。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
Advanced Balance（続き）	• **sticky-srcip-dstport** ：コンテンツ ルールで、クライアントの IP アドレスとサーバの宛先ポート番号の両方に基づいて、クライアントをサーバに固定できるようにする。これは、レイヤ 4 のスティッキ性とも呼ばれます。このオプションは、レイヤ 4 またはレイヤ 5 のコンテンツ ルールで使用できます。 • **ssl** ：コンテンツ ルールで、サーバが割り当てた Secure Socket Layer（SSL）バージョン 3 のセッション ID に基づいて、クライアントをサーバに固定できるようにする。コンテンツ ルールのアプリケーション タイプは、SSL にする必要があります。このオプションを使用する場合は、コンテンツ ルールでポートを指定する必要があります。これによって、CSS はその接続をスプーフします。 • **url** ：コンテンツ ルールで、HTTP 要求の URL 内にある設定済みの文字列に基づいて、クライアントをサーバに固定できるようにする。このオプションを使用する場合は、コンテンツ ルールでポートを指定する必要があります。これによって、CSS はその接続をスプーフします。
Sticky Mask	スティッキに使用するサブネット マスク。デフォルトは 255.255.255.255 です。
Sticky Inactivity timeout	コンテンツ ルールのスティッキ接続がアイドル状態でいられる期間。このタイムアウト期間が経過すると、CSS はスティッキ テーブルからそのエントリを削除します。範囲は 0 ～ 65535 分です。デフォルト値は 0 で、この機能が無効であることを示します。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
Sticky No Cookie Found Action	CSS が、クライアント要求にクッキー ヘッダーや指定したクッキー文字列を見つけられないときに、スティッキ クッキーのコンテンツ ルールに対して実行する必要があるアクション。次の値があります。 • **loadbalance** ：クライアント要求にクッキーが見つからない場合、CSS は設定済みのバランス方式を使用する。これがデフォルト設定です。 • **redirect "URL"** ：クライアント要求にクッキーが見つからない場合、CSS はクライアント要求を指定した URL 文字列にリダイレクトする。このオプションを使用する場合は、リダイレクト URL も指定する必要があります。リダイレクト URL は、0 ～ 64 文字のテキスト文字列を引用符で囲んで入力します。 • **reject** ：クライアント要求にクッキーが見つからない場合、CSS はクライアント要求を拒否する。 • **service name** ：クライアント要求にクッキーが見つからない場合、CSS は指定したサーバにクッキーなしのクライアント要求を送信する。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
Sticky Server Down Failover	スティッキ文字列は見つかったが、関連のサービスに障害が発生した場合または一時停止の場合に CSS が実行しなければならないアクション。次の値があります。 • **Balance** ：このフェールオーバー方式では、設定済みのロード バランシング方式に基づいてサービスを使用する（デフォルト）。 • **Redirect** ：このフェールオーバー方式では、現在設定されているリダイレクト文字列に基づいてサービスを使用する。リダイレクト文字列を設定していない場合は、ロード バランシング方式が使用されます。 • **Reject** ：このフェールオーバー方式では、コンテンツ要求を拒否する。 • **Sticky-srcip** ：このフェールオーバー方式では、クライアントの IP アドレスに基づいてサービスを使用する。これは、スティッキ設定に依存します。 • **Sticky-srcip-dstport** ：このフェールオーバー方式では、クライアントの IP アドレスとサーバの宛先ポートに基づいてサービスを使用する。これは、スティッキ設定に依存します。
ArrowPoint Cookie Path	クッキーを送信するパス名。クッキーのデフォルト パスは、/ です。
ArrowPoint Cookie Expiration	クッキーに関連付けられている時間と比較される有効期限。有効期限を設定しない場合、クッキーはクライアントがブラウザを終了すると同時に期限が切れます。
ArrowPoint Cookie CSS/Browser Expired	ブラウザが有効期限に基づいてクッキーを失効させることができるように、**arrowpoint-cookie browser-expire** コマンドを有効にしているかどうかを示します。このコマンドが有効である場合、フィールドには「CSS」の代わりに「Browser」と表示されます。デフォルトは「CSS」です。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
ArrowPoint Cookie Service	クッキーの有効期限が新しいクッキーの送信前に失効した場合、**arrowpoint-cookie expire-services** コマンドを発行して、サービス情報を失効させるどうかを指定する。デフォルトでは、クッキーの期限が切れると、CSS は期限の切れたクッキーからのサーバ情報を持つ新しいクッキーを送信します。
ArrowPoint Cookie Advanced	**advanced-balance arrowpoint-cookie** コマンドを発行して、コンテンツ ルールが、arrowpoint クッキーに含まれている選択されたサーバの一意なサービス ID に基づき、クライアントをサーバに固定できるようにするかどうかを指定する。
ArrowPoint Cookie Format	arrowpoint クッキーの有効期限の形式について、RFC 2822 に準拠した形式を有効にするか無効にするかを指定する。**arrowpoint-cookie rfc2822-compliant** コマンドは、arrowpoint クッキーの有効期限のシンタックスを RFC 2822 準拠に設定します。このコマンドを使用すると、arrowpoint-cookie の有効期限のシンタックスは、曜日が 3 文字だけで表され（たとえば「Tues」ではなく「Tue」）、月の初めの文字だけが大文字（たとえば「JAN」ではなく「Jan」）になります。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
String Match Criteria	文字列結果と、その宛先サーバを選択する方式を決定する文字列基準。文字列結果は、設定されているスティッキ タイプに従って、クッキー ヘッダー、URL、または URL 拡張子のいずれかに存在するスティッキ文字列になります。以下のフィールドを参照してください。
String Range	クライアントから送られたクッキー、URL、URL 拡張子内の開始バイト位置と終了バイト位置。バイトの範囲を指定すると、CSS はその範囲内にある情報だけを処理します。 • 範囲は 1 ～ 1999。デフォルトの開始バイトの位置は 1 です。 • 範囲は 2 ～ 2000。デフォルトの終了バイトの位置は 100 です。
String Prefix	スティッキ範囲内の文字列プレフィックス。文字列プレフィックスを設定しない場合、文字列はスティッキ タイプに応じて、クッキー、URL、または URL 拡張子の先頭から有効になります。文字列プレフィックスが設定されているが、スティッキの範囲内で指定されていない場合、ロードバランシング方式はデフォルトのラウンドロビン方式になります。デフォルトはプレフィックスなしです（""）。
String Eos-Char	スティッキ文字列の区切り文字として使用される ASCII 文字
String Ascii-Conversion	文字列に処理を適用する前に、特定のスティッキ文字列範囲内のエスケープした特殊文字の ASCII 変換を有効または無効にするかを指定する。デフォルトでは、ACSII 変換は有効です。
String Skip-Len	文字列結果を検索する際に、スキップするプレフィックスの後のバイト数。デフォルトは 0 です。範囲は 0 ～ 64 です。
String Process-Len	文字列操作で使用されるもので、**string prefix** コマンドで指定したプレフィックスの後から、**string skip-length** で指定したバイト数だけスキップする。範囲は 0 ～ 64 です。デフォルトは 0 です。

表 11-6　show rule sticky コマンドの出力フィールド（続き）

フィールド	説明
String Operation	文字列結果の宛先サーバを選択する方式。文字列結果は、文字列基準コマンドの設定によって決まります。次の値があります。 • **match-service-cookie** ：スティッキ文字列内のサービス クッキーをマッチングしてサーバを選択する。これがデフォルト設定です。一致するものがない場合は、設定されたロード バランシング方式（ラウンドロビンなど）を使用してサーバが選択されます。これは、デフォルトの方式です。 • **hash-a** ：ハッシュ文字列に基本ハッシュ アルゴリズムを適用して、ハッシュ キーを生成する。 • **hash-crc32** ：ハッシュ文字列に CRC32 アルゴリズムを適用して、ハッシュ キーを生成する。 • **hash-xor** ：ハッシュ文字列の各バイトに対し XOR（排他的 OR）を実行して、最終的なハッシュ キーを取得する。
Location-Cookie	ロケーション クッキー文字列の形式（NAME=VALUE）
Location-Cookie Expiration	ロケーション クッキーの有効期限の日付と時刻（dd:hh:mm:ss）。この値は、クッキーの有効期限が切れる時間をクライアント ブラウザに知らせます。
Cookie-Domain	ロケーション クッキーのドメイン名（たとえば .site.com）。クッキー ドメイン名を設定することで、ブラウザは、指定したドメイン名で終わるサイトにクッキーを返送できるようになります。

スティッキ テーブル設定の表示

show sticky-table コマンドを使用して、コンテンツ ルールの高度なロード バランシング方式に基づき、CSS スティッキ テーブルの内容を表示します。CSS スティッキ テーブルに含まれているスティッキ情報を表示するには、任意のモードで、次の **show sticky-table** コマンドを使用します。

- **show sticky-table all-sticky** ：すべてのレイヤ 3、レイヤ 4、SIP Call-ID、SSL、および WAP MSISDN スティッキ エントリを表示する。
- **show sticky-table l3-sticky** ：レイヤ 3 エントリを表示する。
- **show sticky-table l4-sticky** ：レイヤ 4 エントリを表示する。
- **show sticky-table sip-callid-sticky** ：SIP Call-ID エントリを表示する。
- **show sticky-table ssl-sticky** ：SSL エントリを表示する。
- **show sticky-table wap-sticky** ：WAP MSISDN エントリを表示する。

コンテンツのスティッキ設定を表示するには、コンテンツ モードで **show rule sticky** コマンドを使用します。**show rule sticky** コマンドの詳細については、「スティッキ アトリビュートの表示」を参照してください。

CSS スティッキ テーブルに含まれているすべてのスティッキ エントリを表示するには、次のように入力します。

```
(config)# show sticky-table all-sticky
```

表 11-7 に、**show sticky-table all-sticky** コマンドで表示されるすべてのフィールドについて説明します。

表 11-7　show sticky-table all-sticky コマンドの出力フィールド

フィールド	説明
All Sticky List on Slot *n*, Subslot *n*	CSS 内で SP が装着されているスロットとサブスロットの番号。複数の SP を装着している場合、スティッキ テーブル情報はスロット番号別に表示されます。
Entries for Page	スティッキ テーブルの情報のページ番号 表示画面には、1 ページ最大 100 件のスティッキ テーブル エントリが表示されます。デフォルト値はページ 1 です。
Entry Number	表示されたスティッキ テーブル エントリの行番号

表 11-7　show sticky-table all-sticky コマンドの出力フィールド（続き）

フィールド	説明
Hash Value	CSS が別のスティッキ データ タイプ用に生成したキー。ハッシュ値は、スティッキ テーブルに挿入されたクライアント固有情報の表示形式で、インデックス、またはスティッキ テーブルへのエントリ キーとして機能します。CSS では、さまざまなスティッキ データ タイプに対して、次のハッシュ値を生成します。 • レイヤ 3 ：送信元 IP アドレス • レイヤ 4 ：送信元 IP アドレスと宛先ポートの組み合わせ • SIP ：Session Initiation Protocol（SIP）Call-ID（CID） • SSL ：SSL バージョン 3.0 セッション ID（SID） • WAP ：MSISDN ヘッダー フィールド
Rule Index	CSS がルールに割り当てた一意の数値インデックス。これは、**show rule summary** コマンドで表示されるインデックスです。
Rule State	ルールの状態。ACT（アクティブ）または SUSP（一時停止）です。
Srv Index	CSS がサービスに割り当てた一意の数値インデックス。これは、**show service summary** コマンドで表示されるインデックスです。
Srv State	サービスの状態。このフィールドには、サービスが Alive、Dying、または Down のいずれかの状態であることが示されます。
Time (Sec) Elapsed	スティッキ テーブル内のエントリが最後に参照されてからアイドル状態のまま経過した時間を示す。カウンタは 0 から始まり、スティッキ テーブル エントリが再び使用されるまで増加します。
Hit Cnt	CSS スティッキ テーブル内のエントリに対して、CSS がクライアントから受信したトランザクションの回数。

表 11-7　show sticky-table all-sticky コマンドの出力フィールド（続き）

フィールド	説明
Col Cnt	CSS スティッキ テーブル内のエントリに対して、CSS が別のクライアントから受信した同じハッシュ値を持つトランザクションの回数。このフィールドは、**show sticky-table ssl-sticky** コマンドでだけ使用されます。
Elem Type	コンテンツ ルールに関連付けられているスティッキ タイプ。要素タイプには、次の種類があります。 • レイヤ 3 • レイヤ 4 • SIP ：Session Initiation Protocol（SIP）Call-ID（CID） • SSL ：SSL バージョン 3.0 セッション ID（SID） • WAP ：MSISDN ヘッダー フィールド
Inact Cfg (Min)	エントリに関連付けられたコンテンツ ルールに対して設定した無活動タイムアウト期間。このフィールドは、スティッキ エントリがスティッキ テーブルに保持されるアイドル時間の長さを示します。値 0（デフォルト値、この機能が無効であることを示す）は、そのエントリがスティッキ テーブルからタイム アウトされないことを示します。テーブル内で使用頻度の最も少ないエントリは、スティッキ テーブルがいっぱいになり新しいエントリの追加が必要になった時点で、スティッキ テーブルから削除されます。
Total Number of Entries Found	問い合せに対してスティッキ テーブル内に見つかったエントリの総数。この総数の値は、特定のスティッキ データ タイプ（レイヤ 3、レイヤ 4、SIP、SSL、または WAP）ごとに表示することもできます。

レイヤ 3 スティッキ テーブル情報の表示

show sticky-table l3-sticky コマンドを使用して、CSS スティッキ テーブルに含まれているレイヤ 3 スティッキ エントリを表示します。レイヤ 3 は、送信元 IP アドレスに基づいてユーザをサーバに固定します。

このグローバル設定モードのコマンドのシンタックスは次のとおりです。

show sticky-table l3-sticky [**page** {*value*}|**ipaddress** {*ip_address sticky_mask*}]

show sticky-table l3-sticky コマンドでは、次のオプションおよび変数がサポートされています。

- **page** *value* : スティッキ テーブル内の特定のページのレイヤ 3 スティッキ エントリを、1 ページ 100 エントリずつ表示する。1 ～ 5000 の値を入力して、スティッキ テーブルで表示するエントリのページを選択します。表示するページを決めるには、**show sticky-stats** コマンドの出力に含まれる **Total Number of Used Entries Found** の値を 100（ページあたりのエントリ数）で除算します。
- **ipaddress** *ip_address sticky_mask* ：表示するレイヤ 3 スティッキ テーブル エントリの IP アドレス。IP アドレスは、ドット付き 10 進表記（192.168.2.5 など）で入力します。この IP アドレスのコンテンツ ルールのスティッキ マスクをドット付き 10 進表記（255.255.255.0 など）で指定します。コンテンツ ルールのデフォルトのスティッキ マスクは、255.255.255.255 です。

たとえば、スティッキ テーブルの 60 ページからレイヤ 3 スティッキ エントリを表示するには、次のように入力します。

```
(config)# show sticky-table l3-sticky page 60
```

たとえば、スティッキ テーブルで特定の IP アドレスとスティッキ マスクのレイヤ 3 スティッキ エントリを表示するには、次のように入力します。

```
(config)# show sticky-table l3-sticky ipaddress 192.168.2.5
255.255.255.255
```

show sticky-table l3-sticky コマンドで表示されるフィールドの説明については、表 11-7 を参照してください。

レイヤ 4 スティッキ テーブル情報の表示

show sticky-table l4-sticky コマンドを使用して、CSS スティッキ テーブルに含まれているレイヤ 4 スティッキ エントリを表示します。レイヤ 4 スティッキの機能は、送信元 IP アドレスと宛先ポートの組み合わせに基づいてユーザをサーバに固定することを除けば、レイヤ 3 スティッキと同じです。

グローバル設定モードのコマンドのシンタックスは次のとおりです。

show sticky-table l4-sticky [**page** {*value*}|**ipaddress** {*ip_address sticky_mask*} {*port*}]

show sticky-table l4-sticky コマンドでは、次のオプションおよび変数がサポートされています。

- **page** *value* : スティッキ テーブル内の特定のページのレイヤ 4 スティッキ エントリを、1 ページ 100 エントリずつ表示する。1 ～ 5000 の値を入力して、スティッキ テーブルで表示するエントリのページを選択します。表示するページを決めるには、**show sticky-stats** コマンドの出力に含まれる **Total Number of Used Entries Found** の値を 100（ページあたりのエントリ数）で除算します。
- **ipaddress** *ip_address sticky_mask* ：表示するレイヤ 3 スティッキ テーブル エントリの IP アドレス。IP アドレスは、ドット付き 10 進表記（192.168.2.5 など）で入力します。この IP アドレスのコンテンツ ルールのスティッキ マスクをドット付き 10 進表記（255.255.255.0 など）で指定します。コンテンツ ルールのデフォルトのスティッキ マスクは、255.255.255.255 です。
- *port* ：表示するエントリの宛先ポート

たとえば、宛先ポート 80 について、スティッキ テーブルで特定の IP アドレスとスティッキ マスクのレイヤ 4 スティッキ エントリを表示するには、次のように入力します。

```
(config)# show sticky-table l4-sticky ipaddress 192.168.2.5
255.255.255.255 80
```

show sticky-table l4-sticky コマンドで表示されるフィールドの説明については、表 11-7 を参照してください。

SIP Call-ID スティッキ テーブル情報の表示

スティッキ テーブルに含まれているエントリをセッション Call-ID に基づいて表示するには、**show sticky-table sip-callid-sticky** コマンドを使用します。Call-ID は、クライアントから SIP サーバに送信された SIP メッセージに含まれる、一意の呼び出し識別子です。

このグローバル設定モードのコマンドのシンタックスは次のとおりです。

show sticky-table sip-callid-sticky [**page** {*value*}|**Call-ID** {*sip_callid*}]

show sticky-table sip-callid-sticky コマンドでは、次のオプションおよび変数がサポートされています。

- **page** *value* ：スティッキ テーブル内の特定のページの SIP Call-ID スティッキ エントリを、1 ページに 100 エントリずつ表示する。1 ～ 5000 の値を入力して、スティッキ テーブルで表示するエントリのページを選択します。表示するページを決めるには、**show sticky-stats** コマンドの出力に含まれる **Total Number of Used Entries Found** の値を 100（ページあたりのエントリ数）で除算します。
- **Call-ID** *sip_callid* ：スティッキ テーブルで表示する特定の Call-ID を指定する。Call-ID 番号を検索するには、パケット トレースを実行します。

たとえば、スティッキ テーブルで特定の Call-ID のスティッキ エントリを表示するには、次のように入力します。

```
(config)# show sticky-table sip-calliD-sticky 12345600@here.com
```

show sticky-table sip-callid-sticky コマンドで表示されるフィールドの説明については、表 11-7 を参照してください。

SSL スティッキ テーブル情報の表示

show sticky-table ssl-sticky コマンドを使用して、スティッキ テーブルに含まれている SSL エントリを表示します。

このグローバル設定モード コマンドのシンタックスは次のとおりです。

show sticky-table ssl-sticky [**rule** {*index*}{**page** {*value*}}|**time** {*number*} {**page** {*value*}}|**sid** {*text*}|**collision**|**page** {*value*}]

show sticky-table ssl-sticky コマンドでは、次のオプションおよび変数がサポートされています。

- **rule** *index* ：コンテンツ ルールに関する、スティッキ テーブル内の SSL エントリを表示する。SSL スティッキ コンテンツ ルールのインデックス番号を入力します。コンテンツ ルールのインデックス番号は、**show rule summary** コマンドで表示されます。
- **page** *value* ：スティッキ テーブル内の特定のページのエントリを、1 ページに 100 エントリずつ表示する。1 ～ 5000 の値を入力して、スティッキ テーブルで表示するエントリのページを選択します。表示するページを決めるには、**show sticky-stats** コマンドで表示される Total Number of Used Entries Found の値を 100（ページあたりのエントリ数）で除算します。
- **time** *number* :経過時間の長さを秒単位で指定する。該当するスティッキ テーブルのエントリが表示対象になります。指定した時間内に参照された、テーブル内のすべてのスティッキ エントリが表示されます。時間は秒単位で入力します。
- **sid** *text* ：スティッキ テーブル内のエントリを、SSL セッション ID（SID）に基づいて表示する。SID 値を、0x プレフィックスを付けない 16 進 ASCII 文字列として入力します。SID 番号を検索するには、パケット トレースを実行します。
- **collision** ：スティッキ テーブル内の、コリジョン カウント（Col Cnt）が 0 より大きいエントリを表示する。

たとえば、スティッキ テーブル内の SSL エントリを、コンテンツ ルール インデックス番号とスティッキ テーブル内でのページ番号に基づいて表示するには、次のように入力します。

```
(config)# show sticky-table ssl-sticky rule 4 page 33
```

show sticky-table ssl-sticky コマンドで表示されるフィールドの説明については、表 11-7 を参照してください。

WAP スティッキ テーブル情報の表示

show sticky-table wap-sticky コマンドを使用して、スティッキ テーブルに含まれているエントリを MSISDN ヘッダー フィールドに基づいて表示します。MSISDN は、Wireless Application Protocol（WAP）を使用する無線のクライアントが使用するヘッダー フィールドです。

グローバル設定モードのコマンドのシンタックスは次のとおりです。

show sticky-table wap-sticky [**page** {*value*}|**msisdn** {*msisdn_header*}]

show sticky-table wap-sticky コマンドでは、次のオプションおよび変数がサポートされています。

- **page** *value* ：スティッキ テーブル内の特定のページの MSISDN スティッキ エントリを、1 ページに 100 エントリずつ表示する。1 ～ 5000 の値を入力して、スティッキ テーブルで表示するエントリのページを選択します。表示するページを決めるには、**show sticky-table all-sticky** コマンドまたは **show sticky-stats** コマンドで表示される Total Number of Entries Found の値リストを 100（ページあたりのエントリ数）で除算します。
- **msisdn** *msisdn_header* ：スティッキ テーブルから表示する MSISDN ヘッダー フィールドを指定する。*msisdn_header* をテキスト文字列として入力します。通常、MSISDN ヘッダー フィールドには無線電話番号が含まれています。MSISDN ヘッダーを検索するには、パケット トレースを実行します。

たとえば、スティッキ テーブル内の MSISDN スティッキ エントリを MSISDN ヘッダーに基づいて表示するには、次のように入力します。

```
(config)# show sticky-table wap-sticky msisdn 6079979410
```

show sticky-table wap-sticky コマンドで表示されるフィールドの説明については、表 11-7 を参照してください。

スティッキ接続の統計情報の表示

show sticky-stats コマンドを使用して、CSS のスティッキ接続に関する統計情報の要約を表示します。

コンテンツのスティッキ設定を表示するには、コンテンツ モードで **show rule sticky** コマンドを使用することもできます。**show rule sticky** コマンドの詳細については、「スティッキ アトリビュートの表示」を参照してください。

次に設定例を示します。

```
(config-owner-content[arrowpoint-rule1])# show sticky-stats
```

表 11-8 に、**show sticky-stats** コマンドで表示されるフィールドについて説明します。

表 11-8　show sticky-stats コマンドの出力フィールド

フィールド	説明
Total Number of New Sticky Entries	スティッキ テーブルで使用する一意のエントリの総数。一意のエントリがスティッキ テーブル内で作成されるたびに、このカウンタが増加します。
Total Number of Sticky Table Hits	CSS スティッキ テーブル内のエントリと一致する要求を CSS がクライアントから受信した回数の合計。CSS がクライアント エントリを受信し、ハッシュ値がスティッキ テーブルに存在する場合に、このカウンタが増加します。CSS では、スティッキ テーブルの検索を実行します。一致するものが見つからない場合、そのエントリは新しいスティッキ カウントと見なされます。
Total Number of Sticky Rejects (No Entry)	CSS がスティッキ要求を拒否した回数の合計。スティッキ テーブルがいっぱいになり、スティッキ テーブルで期限切れになったエントリがない場合、それ以降のスティッキ要求は拒否されます。
Total Number of Sticky Collisions	CSS スティッキ テーブル内のエントリに対して同じハッシュ値を持つ要求を CSS がクライアントから受信し、ロード バランシング サーバが解決されなかった回数の合計

表 11-8　show sticky-stats コマンドの出力フィールド（続き）

フィールド	説明
Total Number of Available Sticky Entries	スティッキ テーブル内で使用可能なスティッキ エントリの総数
Total Number of Used Sticky Entries	スティッキ テーブル内で現在使用されているエントリの総数 CSS は、128K のスティッキ テーブル（CPU メモリが 288MB の場合）、または 32K のスティッキ テーブル（CPU メモリが 144MB の場合）をサポートします。
Total Number of L3 Sticky Entries	スティッキ テーブル内のレイヤ 3 スティッキ エントリの総数
Total Number of L4 Sticky Entries	スティッキ テーブル内のレイヤ 4 スティッキ エントリの総数
Total Number of SSL Sticky Entries	スティッキ テーブル内の SSL セッション ID スティッキ エントリの総数
Total Number of WAP Sticky Entries	スティッキ テーブル内の WAP MSISDN ヘッダー スティッキ エントリの総数
Total Number of SIP Sticky Entries	スティッキ テーブル内の SIP Call-ID スティッキ エントリの総数