



設定テンプレートの管理

この章では、Broadband Access Center (BAC) でサポートされる、デバイス構成およびデバイス管理用のテンプレートについて説明します。この章は、次の項で構成されています。

- [テンプレート ファイル: 概要 \(P.5-2\)](#)
- [テンプレート文法 \(P.5-2\)](#)
 - [SNMP VarBind \(P.5-7\)](#)
 - [マクロ変数 \(P.5-9\)](#)
 - [SNMP TLV \(P.5-11\)](#)
 - [定義済みオプションの符号化タイプ \(P.5-15\)](#)
- [設定ファイルユーティリティの使用方法 \(P.5-23\)](#)

テンプレート ファイル：概要

BAC が使用するテンプレートは、動的 PacketCable、DOCSIS、および CableHome ファイルを配備するときに役立ちます。テンプレートを使用して、読みやすい形式のテンプレート ファイルを作成し、迅速かつ簡単に編集することができます。テンプレートは、有効な PacketCable、DOCSIS、または CableHome ファイルを生成するときに使用する PacketCable、DOCSIS、または CableHome のオプションおよび値を表す ASCII テキスト ファイルです。BAC は *.tmpl* 拡張子を使用してテンプレート ファイルを識別します。サービス クラスでテンプレート ファイルを参照する前に、管理者のユーザ インターフェイスまたは Application Programming Interface (API; アプリケーション プログラミング インターフェイス) を使用して、ファイルとしてそのテンプレート ファイルを RDU に追加する必要があります。

BAC RDU コンポーネントをインストールするときに、いくつかのサンプル テンプレート ファイルが *BPR_HOME/rdu/templates* ディレクトリにコピーされます。

テンプレートを作成または編集するために必要なツールは単純なテキスト エディタですが、独自のテンプレート ファイルを作成する前に、次の情報を十分に理解しておく必要があります。

- BAC プロビジョニングのフロー
- DOCSIS 1.0、1.1、2.0、および 3.0 RFI の仕様
- DOCSIS レイヤ 2 バーチャル プライベート ネットワークの仕様
- PacketCable 1.0、1.1、および 1.5 の仕様
- Multimedia Terminal Adapter (MTA; マルチメディア ターミナル アダプタ) デバイスのプロビジョニングの仕様
- CableHome 1.0 の仕様
- ケーブル デバイス用の SNMP MIB (DOCS-CABLE-DEVICE-MIB など)

テンプレート 文法

テンプレートは、次の種類の文で構成されます。

- コメント (P.5-3)
- インクルード (P.5-3)
- オプション (P.5-4)
- インスタンス修飾子 (P.5-5)
- OUI 修飾子 (P.5-5)

`comment` 文で、テンプレートに説明を入れることができます。 `include` 文で、他のテンプレートで使用するビルディング ブロック テンプレートを作成できます。オプションを使用して、説明のために PacketCable、DOCSIS、または CableHome の Type Length Value (TLV) を指定します。インスタンス修飾子を使用して、複合オプションを特定の個別 TLV にグループ化できます。OUI 修飾子を使用すると、ベンダー固有の情報を含めることができます。表 5-1 に、利用可能なテンプレート文法のオプションを示します。

表 5-1 テンプレートの文法

オプション	説明
<code><comment></code>	<code>::= #[ascii-string]</code>
<code><include></code>	<code>::= include "<filename.tmpl>"</code>
<code><option-description></code>	<code>::= option <option-num> [instance <instance-num>] [oui <oui>] <option-value></code>
<code><option-num></code>	<code>::= <unsigned-byte>[.<unsigned-byte>]*</code>

表 5-1 テンプレートの文法 (続き)

オプション	説明
<option-value>	::= <well-defined-value> <custom-value>
<well-defined-value>	::= <option-value-string>[,<option-value-string>]*
<custom-value>	::= <ascii-value> <hex-value> <ip-value> <snmp-value>
<ascii-value>	::= ascii <ascii-string>
<hex-value>	::= hex <hex-string>
<ip-value>	::= ip <ip-string>
<instance-num>	::= <unsigned integer>
<template>	::= <template-statement>*
<template-statement>	::= <comment> <include> <option-description>
<snmp-value>	::= <snmpvar-oid>,<snmpvar-type>,<snmpvar-value>

コメント

コメントは情報のみを提供し、常にシャープ記号 (#) から行末までの間に配置されます。例 5-1 に、コメントの使用方法的例を示します。

例 5-1 コメントの使用方法的例

```
#
# Template for gold service
#

option 3 1 # enabling network access
```

インクルード

インクルードファイルを使用すると、似ているが少し異なるテンプレートの階層を構築できます。これは、多くのサービス クラスで共通のオプションを定義する場合に、複数のテンプレートでオプションを重複させる必要がなくなり、とても便利です。

単一のテンプレートで複数の include 文を使用できますが、テンプレートの中での include 文の位置は重要です。インクルードファイルの内容は、テンプレートの中に include 文が見つかった場所で読み込まれます。インクルードするテンプレートは、使用する前にファイルとして RDU に追加する必要があります。テンプレートは RDU データベースにパス情報がない状態で格納されるため、インクルードするファイルには ../ などの位置修飾子を含めることができません。例 5-2 と例 5-3 は、それぞれ、インクルード オプションの正しい使用方法と誤った使用方法を示しています。

例 5-2 include 文の正しい使用方法

```
# Valid, including common options
include "common_options.tpl"
```

例 5-3 include 文の誤った使用方法

```
# Invalid, using location modifier
include "../common_options.tpl"

# Invalid, using incorrect file suffix
include "common_options.common"

# Invalid, not using double quotes
include common_options.tpl
```

オプション

PacketCable、DOCSIS、および CableHome の設定ファイルは、適切に符号化されたオプション ID と値のペアで構成されます。サポートされるオプションの形式には、定義済みとカスタムの 2 種類があります。

- オプションを正しく定義するには、オプション番号と値が必要です。値は、オプション番号の符号化タイプに基づいて符号化されます。
- カスタム オプションには、オプション番号、明示的な値の符号化タイプ、および値が必要です。

Option 43 などの複合オプションを使用するときは、インスタンス修飾子を使用して、TLV グループを指定できます。P.5-5 の「[インスタンス修飾子](#)」を参照してください。

テンプレートで定義済みオプションのいずれかを指定するときは、値の値符号化を指定する必要はありません。定義済み符号化タイプの詳細については、P.5-15 の「[定義済みオプションの符号化タイプ](#)」および P.B-2 の「[DOCSIS オプションのサポート](#)」を参照してください。

カスタム オプション (Option 43 など) を指定するときは、オプションの符号化タイプを指定する必要があります。利用可能な符号化タイプは、次のとおりです。

- ASCII : ASCII タイプは、指定された任意の値を NULL ターミネータなしの ASCII 文字列として符号化します。値にスペースを含める場合は、二重引用符で囲む必要があります。
- hex : 値は有効な 16 進数で、各オクテットに正確に 2 文字入るようにする必要があります。値として 01 を指定した場合は、符号化で正確に 1 オクテットが使用されます。値として 0001 を指定した場合は、符号化で正確に 2 オクテットが使用されます。
- IP アドレス : IP アドレス タイプでは、指定された任意の値が 4 オクテットに符号化されます。たとえば、IP アドレス 10.10.10.1 は 0A0A0A01 に符号化されます。
- SNMPVarBind : SNMP OID 文字列、タイプ、および値。これらはそれぞれカンマで区切られます。

1 行に複数の値があるオプションでは、カンマを使用して値を区切ります。それぞれの値は別個に扱われるので、場合によっては値の 1 つを二重引用符で囲む必要がありますが、他の値を囲む必要はありません。複数の値を持つオプションの例として、Option 11 (SNMP VarBind) があります。詳細については、P.5-7 の「[SNMP VarBind](#)」を参照してください。

複合オプションを指定するときは、トップ レベル オプション (Option 4.1 を指定するときの Option 4 など) を指定する必要はありません。例 5-4 と例 5-5 は、それぞれ option 文の正しい使用方法と誤った使用方法を示しています。

例 5-4 option 文の正しい使用方法

```
# Valid, specifying the number for well known option 3
option 3 1

# Valid, specifying the number for option 4 sub-option 1
option 4.1 1

# Valid, specifying a vendor option as hex
option 43.200 hex 00000C

# Valid, specifying a vendor option as ascii
option 43.201 ascii "enable log"

# Valid, specifying a vendor option as IP
option 43.202 ip 10.4.2.1
```

例 5-5 option 文の誤った使用方法

```
# Invalid, using hex with incorrect hex separator
option 43.200 hex 00.00.0C

# Invalid, not using double quotes when needed
option 43.201 ascii enable log

# Invalid, not specifying IP address correctly
option 43.202 ip 10-10-10-1

# Invalid, specifying the description for option "Network Access Control"
option "Network Access Control" 1

# Invalid, specifying top level option
option 4
```

インスタンス修飾子

インスタンス修飾子は、複合オプションを特定の個別 TLV にグループ化するために使用します。例 5-6 と例 5-7 は、それぞれ、個別の TLV を作成する正しい方式と誤った方式を示しています。IOS コマンドを 2 つの個別のコマンドとして解釈するために、IOS DOCSIS モデムをイネーブルにする必要があります。

例 5-6 正しい IOS コマンドライン入力

```
# Valid, each IOS command gets its own TLV
option 43.8 instance 1 00-00-0C
option 43.131 instance 1 ascii "login"
option 43.8 instance 2 00-00-0C
option 43.131 instance 2 ascii "password cable"
```

例 5-7 誤った IOS コマンドライン入力

```
# Invalid, IOS commands are grouped into one TLV
option 43.8 00-00-0C
option 43.131 ascii "login"
option 43.131 ascii "password cable"

# Invalid, using instance on non-compound options
option 3 instance 1 1
```



(注) Option 43.8 の符号化タイプは、Organizationally Unique Identifier (OUI) です。例 5-4 に示されている例とは異なり、このタイプは 00-00-0C 形式のみ受け入れます。

OUI 修飾子

OUI 修飾子は、Option 43 とそのサブオプションを使用して、マルチベンダーのサポートを強化します。

BAC 4.0 では、単一のテンプレートを使用して、多くのベンダーのさまざまな TLV 43 を指定できます。例 5-8 は、OUI 形式を XX-XX-XX として指定します。

- FF-FF-FF : DOCSIS 一般拡張の符号化を指定するベンダー ID を示します。
- 00-00-0C : シスコ固有のケーブル モデム Option 43 とそのサブオプションを指定するシスコベンダー ID を示します。

例 5-8 は、L2VPN のアップストリーム トラフィックを分類するためにケーブル モデム設定ファイルを使用する、L2VPN の BAC サポートを示しています。このテンプレートの内容を使用して、次のサブ TLV を生成できます。

- OUI=FF-FF-FF を使用する、DOCSIS 一般拡張の符号化の 43.5.1 と 43.5.2.2。
- OUI=00-00-0C を使用する、シスコ固有の Option 43 の 43.1。

ただし、DOCSIS の仕様に準拠するために、TLV 43 の最初の サブ TLV として次のいずれかを挿入する必要があります。

- 一般拡張情報を符号化するために DOCSIS 拡張フィールドを使用する場合は、0xFFFFFFFF。
- シスコ固有の subTLV を生成する場合は、0x00000C。

例 5-8 OUI 修飾子の正しい使用方法

```
# Upstream L2VPN Classifier Example

# This example shows how to classify upstream traffic from a specific CPE
# onto an upstream L2VPN service flow, in which other CPE attached to
# the cable modem forward to the non-L2VPN forwarder, as depicted below.

# This example also demonstrates that when using the DOCSIS extension
# field (TLV 43) to encode general extension information (GEI), you do
# not need to specify oui=FF-FF-FF. You only need to specify the OUI tag when
# general extension encoding is not used and vendor-specific encoding is used.

# Upstream L2VPN Classifier Cable Modem Config File

# (43) Per-CM L2VPN Encoding
# GEI (43.8) Vendor ID : 0xFFFFFFFF for GEI
option 43.8 instance 1 ff-ff-ff

# GEI (43.5) for L2VPN Encoding
# GEI (43.5.1) VPNID Subtype
option 43.5.1 instance 1 0234560003

# GEI (43.5) for L2VPN Encoding
# GEI (43.5.2) IEEE 802.1Q Format Subtype
# VLAN ID 25
option 43.5.2.2 instance 1 25

# Cisco Specific Vendor Option Encodings
# (43.8) Vendor ID : 00-00-0C (Cisco Vendor ID)
option 43.8 instance 2 00-00-0C

# Cisco Vendor Specific option (43.1)
# Static Downstream Frequency
# Frequency 402750000
option 43.1 instance 2 oui 00-00-0C 402750000

# Cisco Specific Vendor Option Encodings
# (43.8) Vendor ID : 00-00-0C (Cisco Vendor ID)
option 43.8 instance 3 00-00-0C

# Cisco Vendor Specific option (43.3)
# Update Boot Monitor Image
# image name (boot_monitor_image.bin)
option 43.3 instance 3 oui 00-00-0C boot_monitor_image.bin
```

例 5-9 と例 5-10 は、OUI 修飾子の誤った使用方法を示しています。

例 5-9 OUI 修飾子の誤った使用方法

```
# Invalid, OUI tag needs to be present for each 43 suboption if/when general extension
# encoding is not used and vendor-specific encoding is used.
```

```
option 43.8 00-00-0C
```

```
option 43.3 boot_monitor_image.bin
```

例 5-10 OUI 修飾子の誤った使用方法

```
# Invalid, when both OUI and instance modifier are used in authoring a template,
# "instance" modifier needs to occur before "oui" modifier.
```

```
option 43.8 instance 1 00-00-0C
```

```
option 43.3 oui 00-00-0C instance 1 boot_monitor_image.bin
```

SNMP VarBind

DOCSIS Option 11、PacketCable Option 64、または CableHome Option 28 を指定するときは、Object Identifier (OID) を使用する必要があります。OID を含む MIB は、RDU がロードする次の MIB のいずれかに存在する必要があります。オブジェクトを一意に識別するために必要な数の OID を指定する必要があります。OID の名前または番号を使用できます。RDU は、次の MIB を自動的にロードします。

- SNMPv2-SMI
- SNMPv2-TC
- CISCO-SMI
- CISCO-TC
- SNMPv2-MIB
- RFC1213-MIB
- IANAifType-MIB
- IF-MIB

DOCSIS MIB

これらの DOCSIS MIB は、次の RDU にロードされます。

- DOCS-IF-MIB
- DOCS-BPI-MIB
- CISCO-CABLE-SPECTRUM-MIB
- CISCO-DOCS-EXT-MIB
- SNMP-FRAMEWORK-MIB
- DOCS-CABLE-DEVICE-MIB
- DOCS-CABLE-DEVICE-MIB-OBSOLETE
- CISCO-CABLE-MODEM-MIB

RDU にロードされる DOCS-CABLE-DEVICE MIB には、2 つのバージョンがあります。

- DOCS-CABLE-DEVICE-MIB-OBSOLETE (実験的ブランチ)
- DOCS-CABLE-DEVICE-MIB (mib2 ブランチ)

完全修飾された MIB OID (.experimental...) は常に、MIB OID を一意に識別します。

DOCS-CABLE-DEVICE-MIB のうち、完全修飾されていない MIB OID を使用する場合、MIB OID は常に、デフォルトで DOCS-CABLE-DEVICE-MIB に設定されます (DOCS-CABLE-DEVICE-MIB-OBSOLETE には設定されません)。

例 5-11 と例 5-12 は、それぞれ、完全修飾された MIB OID と完全修飾されていない MIB OID の使用方法を示しています。

例 5-11 完全修飾された MIB OID

```
# Valid, uniquely identifying an OID
option 11 .experimental.docsDev.docsDevMIBObjects.docsDevNmAccessTable.docsDevNmAccess
Entry.docsDevNmAccessStatus.1, Integer, 4
```

例 5-12 完全修飾されていない MIB OID (デフォルトで DOCS-CABLE-DEVICE-MIB に設定される)

```
# Valid, NonFully Qualified MIB OID.
option 11 .docsDevNmAccessStatus.1, Integer, 4
```

配備中の DOCSIS CM が DOCS-CABLE-DEVICE-MIB-OBSOLETE を要求しない場合は、常に短縮形の MIB OID を使用できます。

PacketCable MIB

次の PacketCable (北米版) MIB が RDU にロードされます。

- CLAB-DEF-MIB
- PKTC-MTA-MIB
- PKTC-SIG-MIB
- PKTC-EVENT-MIB

CableHome MIB

次の CableHome MIB が RDU にロードされます。

- CABH-CAP-MIB
- CABH-CDP-MIB
- CABH-CTP-MIB
- CABH-PS-DEV-MIB
- CABH-QOS-MIB
- CABH-SEC-MIB

次の追加 MIB が必要ですが、BAC 製品の一部ではありません。

- CABH-CTP-MIB には RMON2-MIB、TOKEN-RING-RMON-MIB が必要です。
- CABH-SEC-MIB には DOCS-BPI2-MIB が必要です。

マクロ変数

マクロ変数はテンプレートで値として指定され、これを使用してデバイス固有のオプション値を指定できます。マクロ変数がテンプレートにあると、プロパティ階層でマクロ変数名が検索され、変数の値が代入されます。変数名はカスタム プロパティで、事前に RDU に定義します。スペースは使用できません。



(注)

マクロ変数のカスタム プロパティを使用する場合は、`DataType.STRING` を使用する必要があります。

カスタム プロパティを定義すると、次のプロパティ階層で使用できるようになります。

- デバイス プロパティ
- プロビジョニング グループ プロパティ
- サービス クラス プロパティ
- DHCP 基準プロパティ
- テクノロジー デフォルト (PacketCable、DOCSIS、CableHome など)
- システム デフォルト

テンプレート パーサーは、階層の下から上にプロパティを検索し (最初はデバイス、次にサービス クラス)、テンプレート オプション構文に変換します。マクロ変数をサポートする構文は次のとおりです。

- `${var-name}` : この構文は、単純な代入です。変数が見つからない場合、パーサーはエラーを生成します。
- `${var-name, ignore}` : この構文では、変数値がプロパティ階層で見つからなかった場合、テンプレート パーサーはこのオプションを無視します。
- `${var-name, default-value}` : この構文では、変数がプロパティ階層で見つからなかった場合に、デフォルト値が使用されます。

例 5-13 と例 5-14 は、それぞれ Option 11 の正しい使用方法と誤った使用方法を示しています。

例 5-13 マクロ変数の正しい使用方法

```
# Valid, using macro variable for max CPE's, straight substitution
option 18 ${MAX_CPES}

# Valid, using macro variable for max CPE's, ignore option if variable not found
# option 18 will not be defined in the DOCSIS configuration file if MAX_CPES
# is not found in the properties hierarchy
option 18 ${MAX_CPES, ignore}

# Valid, using macro variable for max CPE's with a default value
option 18 ${MAX_CPES, 1}

# Valid, using macro variable for vendor option
option 43.200 hex ${MACRO_VAR_HEX}

# Valid, using macro variable for vendor option
option 43.201 ascii ${MACRO_VAR_ASCII}

# Valid, using macro variable for vendor option
option 43.202 ip ${MACRO_VAR_IP}

# Valid, using macro variable in double quotes
option 18 "${MAX_CPES}"

# Valid, using macro variable within a value
option 43.131 ascii "hostname ${HOSTNAME}"

# Valid, using macro variables in multi-valued options
option 11 ${ACCESS_CONTROL_MIB,
.mib-2.docsDev.docsDevMIBObjects.docsDevNmAccessTable.docsDevNmAccessEntry.docsDevNmAc
cessControl.1}, Integer, ${ACCESS_CONTROL_VAL, 3}

# Valid, using macro variable in an include statement
include "${EXTRA_TEMPLATE}"

# Valid, using macro variable in an include statement with a default value
include "${EXTRA_TEMPLATE, modem_reset.tpl}"

# Valid, using macro variable in an include statement with a default value
include "${EXTRA_TEMPLATE, modem_reset}.tpl"

# Valid, using macro variable in an include statement with an ignore clause
include "${MY_TEMPLATE, ignore}"
```

例 5-14 マクロ変数の誤った使用方法

```
# Invalid, using macro variable as the option number
option ${MAX_CPES} 1

# Invalid, using macro variable with space in name
option 18 ${MAX CPES}
```

SNMP TLV

BAC は、Option 11 および 64 を使用して、動的テンプレート ファイルの SNMP TLV をサポートします。次のものが対象です。

- DOCSIS : Broadband Access Center for Cable (BACC) バージョン 2.0 以降。
- PacketCable : BACC バージョン 2.5 以降。
- CableHome : BACC バージョン 2.6 以降。

これらのテンプレート ファイルの SNMP TLV の構文を検証する場合、BAC は SNMP TLV で参照される対応する SNMP OID を含む MIB ファイルを必要とします。テンプレートに、MIB で検出できない SNMP OID を備えた SNMP TLV が含まれている場合、SNMP TLV は構文エラーを生成します。

次の各項では、MIB を使用しない SNMP TLV、またはベンダー固有の MIB を使用した SNMP TLV の追加方法について説明します。

MIB を使用しない SNMP TLV の追加

RDU で MIB をロードしなくても、動的設定ファイル (DOCSIS、PacketCable、CableHome) に SNMP TLV を追加できます。次の方法を使用すると、RDU 設定の拡張から、DOCSISOptionFactory インターフェイスを使用して機能にアクセスできます。

```
public OptionValue createOptionValue(OptionSyntax syntax, String optionNumStr,
String[] optionValueList)
```

上記の方法では、`public OptionSyntax.SNMP` 列挙値を、OID、Type、Value という値のセットを含む `optionValueList` と組み合わせて使用できます。

RDU 動的設定テンプレートから、次の構文を使用して、RDU MIB に対して検証されていない SNMP TLV を指定します。

```
option option-number snmp OID, Type, Value
```

例 :

```
# DOCS-CABLE-DEVICE-MIB:
option 11 snmp .docsDevNmAccessIp.1, IPADDRESS, 192.168.1.1

# Arris vendor specific SNMP TLV (OID numbers only, mix names/numbers)
option 11 snmp .1.3.6.1.4.1.4115.1.3.1.1.2.3.2.0, INTEGER, 6
option 11 snmp .enterprises.4115.1.3.1.1.2.3.2.0, INTEGER, 6

# NOTE: trailing colon required for single octet
option 11 snmp .1.3.6.1.2.1.69.1.2.1.6.3, STRING, 'c0:'
```

表 5-2 は、利用可能な SNMP 変数タイプの名前です。

表 5-2 SNMP 変数タイプ

IETF 標準の SMI データ タイプ	SNMP API の名前
Integer32	INTEGER
Integer (Enumerated)	INTEGER
Unsigned32	UNSIGNED32
Gauge32	GAUGE
Counter32	COUNTER
Counter64	COUNTER64

表 5-2 SNMP 変数タイプ (続き)

IETF 標準の SMI データ タイプ	SNMP API の名前
Timeticks	TIMETICKS
OCTET STRING	STRING
OBJECT IDENTIFIER	OBJID
IpAddress	IPADDRESS
BITS	STRING

たとえば、SMI Integer32 タイプを指定する場合は、Integer32 および INTEGER タイプが利用可能です (大文字と小文字は区別されません)。

OCTET STRING タイプの場合は、OCTET STRING、OCTETSTRING、または STRING タイプがすべて利用可能です。

カスタム SNMP TLV テンプレート オプションを使用して、任意の SNMP TLV (RDU MIB に存在するものを含む) を指定することができます。カスタム SNMP TLV エラー チェックはあまり厳しくないため、誤ったスカラー/テーブルの参照は検出されません (たとえば、OID 名における .0 と .n の区別)。

ベンダー固有の MIB を使用した SNMP TLV の追加

MIB を RDU に追加すると、テンプレートで人間が判読可能な SNMP OID を使用できるようになり、さらに、マクロ変数に SNMP TLV 値を使用できるようになります。

BACC 2.6 以前

使用する SNMP OID に対応する MIB を保持している場合、MIB ファイルを BAC RDU に追加できます。MIB を追加後、新しい MIB で参照される SNMP OID を使用する SNMP TLV が認識されます。

新しい MIB を RDU に追加するには、次の手順に従います。

ステップ 1 新しい MIB ファイルを `BPR_HOME/rdu/mibs` ディレクトリにコピーします。

ステップ 2 `/docsis/mibs/custom/mibList` プロパティを次の場所にコピーします。このプロパティ値には、MIB ファイル名のカンマ区切りリストが含まれます。

- a. `rdu.properties` ファイル。このファイルは、RDU と管理者のユーザ インターフェイスが使用します。このファイルは `BPR_HOME/rdu/conf` ディレクトリにあります。
- b. `api.properties` ファイル。このファイルは設定ファイル ユーティリティ (`runCfgUtil.sh` ツール) が使用します。



(注) `api.properties` ファイルは、BAC のインストール処理では作成されません。このファイルは、初めて使用するときに任意のテキスト エディタを使用して手動で作成する必要があります。このファイルは `BPR_HOME/rdu/conf` ディレクトリに置いてください。

`api.properties` ファイルには、`/docsis/mibs/custom/mibList` が含まれます。これは、Arris embedded MTA (eMTA) で使用できる MIB のセット用に設定されます。

ステップ 3 `/etc/init.d/bprAgent restart rdu` コマンドを使用して、BAC プロセス ウォッチドッグ経由で RDU を再起動します。

次の例では、ARRIS MTA を設定するためにテンプレートで使用する追加の ARRIS MIB について説明しています。

Arris ベンダー固有の SNMP TLV 使用すると仮定します。

```
option 11 .ppCfgMtaCountryTemplate.0, INTEGER, 9
```

次の MIB ファイルが利用可能になります。

- ARRIS-MIB
- ARRIS-CM-CAPABILITY-MIB
- ARRIS-CM-DEVICE-MIB
- ARRIS-MTA-DEVICE-MIB
- PACKETPORT-MIB

MIB ファイルを *BPR_HOME/rdu/mibs* ディレクトリにコピーし、次のプロパティを *api.properties* ファイルと *rdu.properties* ファイルに挿入する必要があります。

```
/docsis/mibs/custom/mibList=ARRIS-MIB,ARRIS-CM-CAPABILITY-MIB,ARRIS-CM-DEVICE-MIB,ARRIS-MTA-DEVICE-MIB,PACKETPORT-MIB
```

BACC 2.7 以降



(注)

BACC 2.7 以降では、*/docsis/mibs/custom/mibList* プロパティは、*/snmp/mibs/mibList* に名前が変更されています。

使用する SNMP OID に対応する MIB を保持している場合、MIB ファイルを BAC RDU に追加できます。MIB を追加後、新しい MIB で参照される SNMP OID を使用する SNMP TLV が認識されます。

新しい MIB を BAC RDU に追加するには、次の手順に従います。

-
- ステップ 1** BAC 管理者のユーザ インターフェイスを起動します。
 - ステップ 2** ナビゲーション バーで、**Configuration > Defaults** をクリックします。
 - ステップ 3** 表示される Configure Defaults ページの左ペインにある System Defaults リンクをクリックします。
 - ステップ 4** MIB List フィールドの末尾に新しい MIB の内容を貼り付けます。
 - ステップ 5** **Submit** をクリックします。



(注)

バージョン 2.7 以降では、MIB の解析ツールは強化されています。その結果、以前はエラーなしで解析されていた MIB のバージョンで時々エラーが返されるようになりました。エラーが発生し、新しい MIB を編集することで解決できない場合は、Cisco Technical Assistance Center にお問い合わせください。

MIB のロード順のデバッグ

一般に、ベンダーが提供するさまざまな MIB は、MIB 間の依存関係を満たすために特定の順番でロードする必要があります。ただし、多くの場合、ベンダーは正しいロード順を提供しないので、ユーザ自身が正しいロード順を決定する必要があります。この項では、BAC のデバッグ情報を使用して、MIB のロード順の問題を解決する方法について説明します。



(注) BAC 内の MIB のロード順は、次のプロパティにリストされている MIB の順番で設定されます。

- `/docsis/mibs/custom/mibList` プロパティ (BACC 2.6.x 以前のリリースを使用している場合)。
- `/snmp/mibs/MibList` プロパティ (BACC 2.7.x 以降のリリースを使用している場合)。

`runCfgUtil.sh` ツールを使用して、`api.properties` ファイルで指定されているプロパティの正しいロード順を判断できます。`runCfgUtil.sh` ツールは、`BPR_HOME/rdu/bin` ディレクトリにあります。



(注) この手順では、BACC 2.7.x 以降のリリースで使用する `/snmp/mibs/MibList` プロパティを参照します。2.6.x 以前のリリースを実行している場合は、`/docsis/mibs/custom/mibList` プロパティを使用してください。

ステップ 1 `api.properties` ファイルを使用して `Configure runCfgUtil.sh` を設定し、このステップで説明されているのと同様の設定内容を使用します。`api.properties` ファイルは、BAC トレースをイネーブルにし、MIB デバッグ情報をユーザ コンソールに送信します。

```
#
# Enable logging to the console
#
/server/log/1/level=Info
/server/log/1/properties=level
/server/log/1/service=com.cisco.csrc.logging.SystemLogService
/server/log/1/name=Console
#
# Enable trace categories
#
/server/log/trace/rduserver/enable=enabled
#
# The list of MIBs to be added.
#
/snmp/mibs/MibList=arrishdr.mib,arris_cm_capability.mib,arris_mta_device.mib,arris_sip
.mib,arris_cm.mib,pp.mib,blp2.mib,dev0.mib,docs_evtnt.mib,qos.mib,test.mib,usb.mib,snmp
v2_conf.mib,rfc1493.mib,rfc1907.mib,rfc2011.mib,rfc2013.mib,rfc2233.mib,rfc2571.mib,rf
c2572.mib,rfc2573.mib,rfc2574.mib,rfc2575.mib,rfc2576.mib,rfc2665.mib,rfc2669.mib,rfc2
670.mib,rfc2786.mib,rfc2851.mib,rfc2933.mib,rfc 3083.mib
```

ステップ 2 `runCfgUtil.sh` をこのように設定し、ツールを実行して、Option 11 または Option 64 (SNMP 符号化) を含んでいる任意のテンプレートを符号化します。ツールは、`/snmp/mibs/MibList` 内で指定されている MIB のロードを試み、MIB のロードエラーとともに完全なデバッグ情報をユーザ コンソールに送信します。

ステップ 3 エラー情報を使用して、MIB の全セットがエラーなしでロードされ、ファイルの符号化が成功するまで、`/snmp/mibs/MibList` 内で指定されている MIB の順番を変更します。

ステップ 4 正常なロード順を決定したら、使用している BACC のバージョンに基づき、このステップで説明されている手順を実行します。

BACC 2.7 以降

- a. 管理者のユーザ インターフェイスから、**Configuration > Defaults** をクリックして、**System Defaults** リンクをクリックします。
- b. MIB List フィールドに、ロード順の情報をコピーします。
これで、RDU はベンダー提供の MIB を使用してテンプレートを符号化するように設定されます。



(注) RDU を再起動する必要はありません。

api.properties ファイルと MIB List フィールドでは、*/snmp/mibs/mibList* 文字列を使用してください。

BACC 2.6 以前

- a. *rdu.properties* ファイルの */docsis/mibs/custom/mibList* プロパティにロード順の情報をコピーします。このファイルは *BPR_HOME/rdu/conf* ディレクトリにあります。
- b. */etc/init.d/bprAgent restart rdu* コマンドを使用して、BAC プロセス ウォッチドッグ経由で RDU を再起動します。
これで、RDU はベンダー提供の MIB を使用してテンプレートを符号化するように設定されます。

定義済みオプションの符号化タイプ

表 5-3 に、定義済み符号化タイプを持つオプションを示します。

表 5-3 定義済みオプション符号化タイプ

符号化	入力	例
認可アクション	<p>8 ビット符号なし整数または説明のための文字列。</p> <p>認可を許可する場合の値は次のとおりです。</p> <ul style="list-style-type: none"> • 0 • permit <p>認可を拒否する場合の値は次のとおりです。</p> <ul style="list-style-type: none"> • 1 • deny 	<p>0 1</p> <p>permit deny</p>

表 5-3 定義済みオプション符号化タイプ (続き)

符号化	入力	例
アクセス ビュー コントロール	<p>8 ビット符号なし整数または説明のための文字列。</p> <p>アクセス ビューの SNMPv3 Access View サブツリーを含める場合の値は次のとおりです。</p> <ul style="list-style-type: none"> 1 included <p>アクセス ビューの SNMPv3 Access View サブツリーを除外する場合の値は次のとおりです。</p> <ul style="list-style-type: none"> 2 excluded 	<p>1 2</p> <p>included excluded</p>
アクセス ビュー タイプ	<p>8 ビット符号なし整数または説明のための文字列。</p> <p>読み取り専用アクセスをイネーブルにする場合の値は次のとおりです。</p> <ul style="list-style-type: none"> 1 read-only <p>読み取りと書き込みアクセスをイネーブルにする場合の値は次のとおりです。</p> <ul style="list-style-type: none"> 2 read-write 	<p>1 2</p> <p>Read-only Read-write</p>
ActInact	<p>8 ビット符号なし整数または説明のための文字列。</p> <p>TLV をディセーブルにする場合の値は次のとおりです。</p> <ul style="list-style-type: none"> 0 inactive <p>TLV をイネーブルにする場合の値は次のとおりです。</p> <ul style="list-style-type: none"> 0 active 	<p>0 1</p> <p>Inactive Active</p>
BitFlag8	8 ビット符号なし整数。出力は、値を表す 16 進数の文字列です。	0xFE
BitFlag32	32 ビット符号なし整数。出力は、値を表す 16 進数の文字列です。	0xFFFFF000
ブール値	0 は false、1 は true です。	0 1

表 5-3 定義済みオプション符号化タイプ (続き)

符号化	入力	例
Byte16	32 文字の 16 進数文字列で指定される 16 バイト。これは通常、ケーブル モデムと CMTS の MIC オプションを表すために使用されます。0x プレフィックスは使用できません。	なし。 BAC はケーブル モデムと CMTS MIC オプションのハッシュを自動的に計算します。
バイト	一連の 16 進数オクテット。各オクテットは 2 文字にする必要があります。	000102030405060708
CPE アクセス コントロール	8 ビット符号なし整数または説明のための文字列。 デバイス アクセス コントロールをディセーブルにする場合の値は次のとおりです。 <ul style="list-style-type: none"> 0 ディセーブル デバイス アクセス コントロールをイネーブルにする場合の値は次のとおりです。 <ul style="list-style-type: none"> 1 イネーブル 	0 1 Disabled Enabled
DSC 分類子	8 ビット符号なし整数または DSC 分類子の文字列名。符号なし整数には次のものがあります。 <ul style="list-style-type: none"> 0 : DSC 追加分類子 1 : DSC 置換分類子 2 : DSC 削除分類子 	0
EnableDisable	8 ビット符号なし整数または説明のための文字列。 ディセーブルにする場合の値は次のとおりです。 <ul style="list-style-type: none"> 0 ディセーブル イネーブルにする場合の値は次のとおりです。 <ul style="list-style-type: none"> 1 イネーブル 	0 1 Disabled Enabled

表 5-3 定義済みオプション符号化タイプ (続き)

符号化	入力	例
Inet アドレス ピ ア	1 バイトの InetAddressTypeCode。 <ul style="list-style-type: none"> • 1 は IPv4 • 2 は IPv6 この値の後に IPv4 または IPv6 インターネットアドレスが続きます。 その結果、この長さは IPv4 用が 5 バイト (1+4)、IPv6 用が 17 バイト (1+16) になります。	1,10.112.125.111 2,0:0:0:0:0:ffff:8190:3426
IP アドレス	ドット (.) で区切られた 4 つの符号なし整数 8。	10.10.10.1
IPv6 アドレス	IPv6 アドレス x:x:x:x:x:x:x:x を表す文字列。ここで、x は、アドレスの 8 つの 16 ビット部分を表す 1 ~ 4 桁の 16 進数です。	2001:db8:0:0:8:800:200c:417a
IPv4 または IPv6 アドレス	IPv4 または IPv6 アドレスを表す文字列。	10.112.125.111 0:0:0:0:0:ffff:8190:3426
複数の IP アド レス	IP アドレスのカンマ区切りリスト。	10.11.12.13,10.11.12.14
複数の IPv6 ア ドレス	IPv6 アドレスのカンマ区切りリスト。	2001:db8:0:0:8:800:200c:417a,f f01:0:0:0:0:0:0:101
MAC アドレス	コロン (:) またはダッシュ (-) で区切られた 6 つの 16 進数オクテット。各オクテットは正確に 2 文字にする必要があります。コロンとダッシュを同時に使用することはできません。	00:01:02:03:04:05 00-01-02-03-04-05
MAC アドレスと マスク	コロン (:) またはダッシュ (-) で区切られた 12 のオクテット。各オクテットは 2 文字にする必要があります。コロンとダッシュを同時に使用することはできません。最初の 6 つのオクテットは MAC アドレスを表し、残りの 6 つは MAC アドレスのマスクを表します。	00:01:02:03:04:05:06:07:08:09: 0A:0B 00-01-02-03-04-05-06-07-08-09- 0A-0B
NoLV	タイプのみ。値や長さは含みません。	null
NVTASCII	ASCII 文字列。符号化文字列を NULL で終わることはできません。	This is an ASCII string
OID	SNMP OID 文字列。	sysinfo.0
OIDCF	SNMP OID 文字列とカンマで区切られた符号なし整数 (0 または 1)。	sysinfo.0,1

表 5-3 定義済みオプション符号化タイプ (続き)

符号化	入力	例
OnOff	<p>8 ビット符号なし整数。</p> <p>TLV をオンにする場合の値は次のとおりです。</p> <ul style="list-style-type: none"> • 0 • On <p>TLV をオフする場合の値は次のとおりです。</p> <ul style="list-style-type: none"> • 1 • Off 	<p>0</p> <p>1</p> <p>On</p> <p>Off</p>
OUI	<p>コロン (:) またはダッシュ (-) で区切られた 3 つの 16 進数オクテット。各オクテットは 2 文字にする必要があります。</p>	00-00-0C
RFC868 時刻	<p>RFC868 時刻を表す 32 ビット符号なし整数。出力は、<i>MM/dd/yyyy HH:mm:ss</i> の形式の日付と時刻の文字列です。</p>	<p>0</p> <p>("12/31/1899 19:00:00" を表します)</p> <p>4294967295</p> <p>("02/07/2036 01:28:15" を表します)</p>
サービス フロー	<p>8 ビット符号なし整数またはサービス フローの説明のための文字列。出力は次の内容を示すサービス フローです。</p> <ul style="list-style-type: none"> • 0: 予約済み • 1: 未定義 (CMTS の実装に依存) • 2: ベスト エフォート • 3: 非リアルタイム ポーリング サービス • 4: リアルタイム ポーリング サービス • 5: 任意の認可サービス アクティビティ検出 • 6: 任意の認可サービス 	0

表 5-3 定義済みオプション符号化タイプ (続き)



符号化	入力	例
SNMPVarBind	<p>SNMP OID 文字列、タイプ、値。これらはそれぞれカンマで区切られます。有効な値は次のとおりです。</p> <ul style="list-style-type: none"> • BITS • Counter • Counter32 • Counter64 • Gauge • Gauge32 • INTEGER • Integer32 • IpAddress • OCTETSTRING • OBJECTIDENTIFIER • Opaque • TimeTicks • Unsigned32 <p> (注) OCTETSTRING は、末尾に NULL を含まない 16 進表記に変換される文字列 (オクテット文字列など) または引用符で囲まれた 16 進表記 ('aa:bb:cc' など) です。</p>	<pre>.experimental.docsDev.docsDevM IBObjects.docsDevNmAccessTable .docsDevNmAccessEntry.docsDevN mAccessStatus.1, INTEGER, 4</pre>
SrvChangeAct	<p>0 ~ 3 の範囲に限定された 8 ビット符号なし整数または SrvChangeAct の説明。説明のための文字列の出力は次のとおりです。</p> <ul style="list-style-type: none"> • 0 : PHS ルールの追加 • 1 : PHS ルールの設定 • 2 : PHS ルールの削除 • 3 : すべての PHS ルールの削除 	0
サブタイプ	1 つまたは 2 つのカンマで区切られた符号なし整数 8。	12 12, 14

表 5-3 定義済みオプション符号化タイプ (続き)

符号化	入力	例
転送アドレスとマスク	IPv4 の場合、ドット付き表記の 4 つのオクテット IP アドレスと、カンマ (,) で区切られたポート番号。 IPv6 の場合、次のようなドット付き表記または文字列です。 <ul style="list-style-type: none"> ドット付き表記の有効な IPv6 アドレスと、カンマ (,) で区切られたポート番号。 IPv6 アドレスを表す文字列と、カンマ (,) で区切られたポート番号。たとえば、<code>x:x:x:x:x:x,x,1234</code> の場合、<code>x</code> は、アドレスの 8 つの 16 ビット部分を表す 1 ~ 4 桁の 16 進数です。 	IPv4 <code>10.112.125.111,5678</code> IPv6 <code>2001.db8.0.0.8.800.200c.417a,5678</code> <code>2001:db8:0:0:8:800:200c:417a,5678</code>
8 ビット符号なし整数	0 ~ 255	14
16 ビット符号なし整数	0 ~ 65535	1244
32 ビット符号なし整数	0 ~ 4294967295	3455335
8 ビット符号なし整数と 16 ビット符号なし整数	カンマで区切られた 1 つの符号なし整数 8 と 1 つの符号なし整数 16。	3,12324
8 ビット符号なし整数のペア	カンマで区切られた 2 つの符号なし整数 8。	1,3
3 ビット バイトの 8 ビット符号なし整数	カンマで区切られた 3 つの符号なし整数 8。	1,2,3
検証	8 ビット符号なし整数 検証をイネーブルにする場合の値は次のとおりです。 <ul style="list-style-type: none"> 0 検証 検証をディゼーブルにする場合の値は次のとおりです。 <ul style="list-style-type: none"> 1 Don't Verify  (注) Verify TLV の true と false の定義は、DOCSIS 1.1 の仕様 (Option 26.11) と一致しています。	0 = verify 1 = don't verify
ZTASCII	ASCII 文字列。符号化文字列は NULL で終わります。	This is an ASCII string

BITS 値の構文

BITS 型を使用する場合は、ラベル（「interval1 interval2 interval3」）または数値による位置（「0 1 2」）を指定する必要があります。ラベル値は 1 ベースで、ビット値は 0 ベースであることに注意してください。

ビット番号を使用する構文の例を示します。

```
option 11 .pktcSigDevR0Cadence.0,STRING,"0 1 2 3 4 5 6 7 8 9 10 11 12 13 14"
```

ラベルを使用するカスタマー オクテット文字列（FFFE000000000000）の構文を示します。

```
option 11 .pktcSigDevR0Cadence.0,STRING,"interval1 interval2 interval3  
interval4 interval5 interval6 interval7 interval8 interval9 interval10  
interval11 interval12 interval13 interval14 interval15"
```

OCTETSTRING の構文

OCTETSTRING は、末尾に NULL を含まない 16 進表記に変換される文字列（オクテット文字列など）または一重引用符で囲まれた 16 進表記（'aa:bb:cc' など）です。

設定ファイルユーティリティの使用法

設定ファイルユーティリティを使用して、PacketCable 1.0/1.1/1.5、DOCSIS 1.0/1.1/2.0/3.0、および CableHome のテンプレートファイルと設定ファイルをテスト、検証、および表示できます。これらの作業は、独自の設定ファイルを正常に展開するために重要です。テンプレートの詳細については、[P.5-2 の「テンプレートファイル：概要」](#)を参照してください。

設定ファイルユーティリティは、RDU をインストールしたときのみ利用可能です。このユーティリティは `BPR_HOME/rdu/bin` ディレクトリにインストールされます。

符号化するテンプレートファイルとデコードするバイナリファイルの両方が、設定ファイルユーティリティを起動するディレクトリに存在する必要があります。

この項のすべての例では、RDU が運用中で、次の条件が適用されていることを前提にしています。

- BAC アプリケーションは、デフォルトのホームディレクトリ (`/opt/CSCObac`) にインストールされています。
- RDU ログイン名は **admin** です。
- RDU ログインパスワードは **changeme** です。



(注)

この項の例では、一部が出力例にとって重要でない場合に、その部分を省略していることがあります。その場合は、例中のサマリーの直前に省略記号 (...) を示しています。

この項では、次のトピックについて取り上げます。

- [設定ファイルユーティリティの実行 \(P.5-24\)](#)
- [BAC へのテンプレートの追加 \(P.5-25\)](#)
- [テンプレートファイルへのバイナリファイルの変換 \(P.5-26\)](#)
- [ローカルテンプレートファイルのテンプレート処理のテスト \(P.5-28\)](#)
- [外部テンプレートファイルのテンプレート処理のテスト \(P.5-29\)](#)
- [コマンドラインでのマクロ変数の指定 \(P.5-32\)](#)
- [マクロ変数のデバイスの指定 \(P.5-33\)](#)
- [バイナリファイルへの出力の指定 \(P.5-34\)](#)
- [ローカルバイナリファイルの表示 \(P.5-35\)](#)
- [外部バイナリファイルの表示 \(P.5-36\)](#)
- [PacketCable Basic フローの有効化 \(P.5-37\)](#)
- [マルチベンダーをサポートするための TLV 43 の生成 \(P.5-39\)](#)

設定ファイルユーティリティの実行

次の手順と例で、「設定ファイルユーティリティを実行する」というフレーズは、指定されたディレクトリから `runCfgUtil.sh` コマンドを入力することを意味します。設定ファイルユーティリティを実行するには、`BPR_HOME/rdu/bin` ディレクトリから次のコマンドを実行します。

`runCfgUtil.sh options`

利用可能な `options` は、次のとおりです。

- `-c secret` : DOCSIS テンプレート ファイルを解析するときの CMTS 共有秘密情報を指定します。デフォルトの共有秘密情報を指定するには、`-c cisco` と入力します。
- `-cablehome` : 入力ファイルが CableHome ポータル サービス設定ファイルであることを示します。`-docsis` または `-pkt` オプションと同時に使用することはできません。
- `-d` : バイナリ入力ファイルをデコードします。`-e` オプションと同時に使用することはできません。
- `-docsis` : 入力ファイルが DOCSIS 設定ファイルであることを指定します。このデフォルトを `-pkt` オプションと同時に使用することはできません。
- `-v version` : 使用している DOCSIS のバージョンを指定します。たとえば、DOCSIS 1.1 を使用している場合は、`-v 1.1` と入力します。バージョン番号を指定しない場合、コマンドはデフォルトで DOCSIS 2.0 を使用します。BAC がサポートする値は 1.0、1.1、2.0、および 3.0 です。
- `-e` : テンプレート入力ファイルを符号化します。このデフォルトを `-d` オプションと同時に使用することはできません。
- `-g` : DOCSIS、PacketCable、または CableHome バイナリ ファイルからテンプレート ファイルを生成します。
- `-h host:port` : ホストとポート番号を指定します。デフォルトのポート番号は 49187 です。
- `-i device-id` : テンプレート解析中にマクロ変数を代入するときに使用するデバイスを示します。たとえば、デバイス MAC アドレスが 1,6,00:00:00:00:00:01 の場合は、`-i 1,6,00:00:00:00:00:01` と入力し、デバイス DUID が 00:03:00:01:00:18:68:52:75:c0 の場合は、`-i 00:03:00:01:00:18:68:52:75:c0` と入力します。このオプションを使用するときは、`-u` および `-p` オプションを使用して、それぞれユーザ名とパスワードを指定する必要もあります。`-m` オプションと同時に使用することはできません。
- `-l filename` : 入力ファイルがローカル ファイル システムにあることを示します。たとえば、入力ファイルの名前が `any_file` の場合、`-l any_file` と入力します。`-r` オプションと同時に使用することはできません。
- `-loc` : PacketCable のロケールを `na` (北アメリカ) または `euro` (ヨーロッパ) に指定します。デフォルトは `na` です。MTA が `euro-MTA` の場合は、ロケールを `euro` に設定する必要があります。
- `-m macros` : マクロ変数のキーと値のペアを指定します。形式は `key=value` です。複数のマクロ変数が必要な場合は、`key_1=value_1,key_2=value_2` のように、キーと値のペアを2つのカンマで区切ります。`-i` オプションと同時に使用することはできません。
- `-p password` : RDU に接続するとき使用するパスワードを指定します。たとえば、パスワードが 123456 の場合は、`-p 123456` と入力します。
- `-o filename` : 解析したテンプレート ファイルをバイナリ ファイルとして保存します。たとえば、出力を `op_file` という名前のファイルに保存するには、`-o op_file` と入力します。
- `-pkt` : 入力ファイルが PacketCable MTA 設定ファイルであることを示します。`-docsis` オプションと同時に使用することはできません。
- `-r filename` : 入力ファイルが RDU に追加したりモート ファイルであることを示します。たとえば、ファイル名が `file25` の場合は、`-r file25` と入力します。このオプションを使用するときは、`-u` および `-p` オプションを使用して、それぞれユーザ名とパスワードを指定する必要もあります。`-l` オプションと同時に使用することはできません。
- `-s` : 解析されたテンプレートまたはバイナリ ファイルの内容を人間が判読可能な形式で表示します。
- `-t` : PacketCable 符号化タイプを `Secure` または `Basic` に指定します (デフォルトは `Secure`)。

- **-u username** : RDU に接続するとき使用するユーザ名を指定します。たとえば、ユーザ名が **admin** の場合、**-u admin** と入力します。



(注)

設定ファイルユーティリティでは、テンプレートファイルに **Option 19** (TFTP サーバ タイムスタンプ) と **Option 20** (TFTP サーバのプロビジョニングされたモデム アドレス) は含まれません。ただし、BAC TFTP 混在では含まれます。また、**options 6** (CM MIC) および **7** (CMTS MIC) はどちらも、符号化されたテンプレートファイルに自動的に挿入されます。そのため、これらの **Message Integrity Check (MIC; メッセージ完全性チェック)** を指定する必要はありません。

BAC へのテンプレートの追加

設定ファイルユーティリティを使用して、BAC テンプレートをテストするには、次の手順に従います。

- ステップ 1** P.5-2 の「**テンプレート ファイル: 概要**」の説明に従い、テンプレートを作成します。テンプレートに他のテンプレートを含める場合は、参照されるテンプレートすべてが同一のディレクトリにあることを確認します。
- ステップ 2** ローカル ファイル システムで設定ファイルユーティリティを実行します。テンプレートの構文をチェックするか、または CRS と同じ方法で設定ファイルユーティリティにテンプレートを処理させた後、出力を返すことができます。

テンプレートにマクロ変数が含まれる場合、指定された順番で次の操作を実行します。
 - a. コマンドラインの代入を使用してテストします。
 - b. RDU に追加したデバイスを使用してテストします。
- ステップ 3** テンプレート (および、そのテンプレートにインクルードするテンプレート) を RDU に追加します。
- ステップ 4** 設定ファイルユーティリティを実行して、ファイルを解析します。P.5-29 の「**外部テンプレート ファイルのテンプレート処理のテスト**」を参照してください。

テンプレートにマクロ変数が含まれる場合、指定された順番で次の操作を実行します。
 - a. コマンドラインの代入を使用してテストします。
 - b. RDU に追加したデバイスを使用してテストします。
- ステップ 5** テストが成功したら、そのテンプレートを使用するサービス クラスを設定します。

テンプレート ファイルへのバイナリ ファイルの変換

`runCfgUtil.sh` コマンドを使用して、バイナリ設定メモリ ファイルをテンプレート ファイルに変換します。BAC の動的構成生成は、作成されるテンプレートに基づきます。既存のテスト済みバイナリ ファイルをテンプレート ファイルに自動的に変換すると、プロセスの速度が向上し、エラーが発生する可能性は低下します。

シンタックスの説明 `runCfgUtil.sh -g -l binary_file -o template_file`

- `-g` : 入力バイナリ ファイルからテンプレート ファイルを生成する必要があることを指定します。
- `-l binary_file` : ローカル入力ファイル (パス名を含む) を指定します。すべての場合において、入力バイナリ ファイルの名前には `.cm` ファイル拡張子が割り当てられます (たとえば、`bronze.cm`)。
- `-o template_file` : 出力テンプレート ファイル (パス名を含む) を指定します。すべての場合において、出力テンプレート ファイルの名前には `.tmpl` ファイル拡張子が割り当てられます (たとえば、`test.tmpl`)。

バイナリ ファイルをテンプレート ファイルに変換するには、次の手順に従います。

ステップ 1 `/opt/CSCObac/rdu/samples/docsis` にディレクトリを変更します。

ステップ 2 使用するテンプレート ファイルを選択します。この例では、既存のバイナリ ファイル `unprov.cm` を使用します。

ステップ 3 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -g -l unprov.cm -o test.tmpl -docsis
```

`-docsis` : 入力ファイルを DOCSIS 設定ファイルにすることを指定します。

ユーティリティを実行すると、次のような結果が表示されます。

```
Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

#####
## Template File Generator
## Generated on Fri Oct 12 16:12:51 EST 2007
#####

#####
## Each generated option will be represented by the following:
## The first line will represent a description of the
## generated option
## The second line will represent the generated option
## The third line will represent the custom version
## of the generated option
#####

# (3) Network Access Control
Option 3 01
# Option 3 hex 01

# (4.1) Class ID
Option 4.1 1
# Option 4.1 hex 01

# (4.2) Maximum Downstream Rate
Option 4.2 128000
# Option 4.2 hex 0001F400

# (4.3) Maximum Upstream Rate
Option 4.3 64000
# Option 4.3 hex 0000FA00

# (4.4) Upstream Channel Priority
Option 4.4 1
# Option 4.4 hex 01

# (4.5) Guaranteed Minimum Upstream Channel Data Rate
Option 4.5 0
# Option 4.5 hex 00000000

# (4.6) Maximum Upstream Channel Transmit Burst
Option 4.6 1600
# Option 4.6 hex 0640

# (4.7) Class-of-Service Privacy Enable
Option 4.7 00
# Option 4.7 hex 00

# (11) SNMP MIB Object
Option 11
.iso.org.dod.internet.experimental.docsDev.docsDevMIBObjects.docsDevNmAccessTable.docs
DevNmAccessEntry.docsDevNmAccessStatus.1, INTEGER, createAndGo
# Option 11 hex 3082000F060A2B060103530102010701020104

...

# (18) Maximum Number of CPEs
Option 18 1
# Option 18 hex 01
```

ローカル テンプレート ファイルのテンプレート処理のテスト

`runCfgUtil.sh` コマンドを使用して、ローカル ファイル システムに格納されているテンプレート ファイルの処理をテストします。

シンタックスの説明

`runCfgUtil.sh -pkt -l file`

- `-pkt` : 入力ファイルが PacketCable MTA ファイルであることを示します。
- `-l` : 入力ファイルがローカル ファイル システムにあることを指定します。
- `file` : 解析する入力テンプレート ファイルを示します。

ローカル ファイル システムにあるテンプレート ファイルを解析するには、次の手順に従います。

ステップ 1 `/opt/CSCObac/rdu/samples/packet_cable` にディレクトリを変更します。

ステップ 2 使用するテンプレート ファイルを選択します。この例では、既存のテンプレート ファイル `unprov_packet_cable.tmpl` を使用します。これは PacketCable MTA テンプレートであるため、`-pkt` オプションを使用します。

ステップ 3 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -pkt -l unprov_packet_cable.tmpl
```

`unprov_packet_cable.tmpl` : 解析する入力テンプレート ファイルを示します。

ユーティリティを実行すると、次のような結果が表示されます。

```
Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off   File Bytes      Option  Description                    Value
-----
0     FE0101          254    Telephony Config File         1
      Start/End

3     0B153013060E   11     SNMP MIB Object                .iso.org.dod.internet.privat
      2B06010401A30B                             e.enterprises.cableLabs.clab
      0202010101                                Project.clabProjPacketCable.
      0700020102                                pktcMtaMib.pktcMtaMibObjects
                                                  .pktcMtaDevBase.
                                                  pktcMtaDevEnabled.0, INTEGER,
                                                  false(2)

...

0 error(s), 0 warning(s) detected. Parsing of unprov_packet_cable.tmpl was successful.
The file unprov_packet_cable.tmpl was parsed successfully in 434 ms.
The parser initialization time was 92 ms.
The parser parse time was 342 ms.
```

外部テンプレート ファイルのテンプレート処理のテスト

`runCfgUtil.sh` コマンドを使用して、外部テンプレート ファイルの処理をテストします。

シンタックスの説明 `runCfgUtil.sh -docsis -r file -u username -p password`

- `-r` : 入力ファイルが RDU に追加したファイルであることを示します。
- `file` : 解析する入力テンプレート ファイルを示します。
- `-u username` : RDU に接続するとき使用するユーザ名を指定します。
- `-p password` : RDU に接続するとき使用するパスワードを指定します。
- `-docsis` : ファイルが DOCSIS テンプレートであることを示します。

RDU に追加したテンプレート ファイルを解析するには、次の手順に従います。

ステップ 1 使用するテンプレート ファイルを選択します。この例では、既存のテンプレート ファイル `unprov.tmpl` を使用します。DOCSIS テンプレートを使用するため、`-docsis` オプションを使用します。

ステップ 2 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -docsis -r unprov.tmpl -u admin -p changeme
```

- `unprov.tmpl` : 入力ファイルを示します。
- `admin` : ユーザ名を示します。
- `changeme` : パスワードを示します。

ユーティリティを実行すると、次のような結果が表示されます。



(注) ここに表示されている結果は説明のためだけのものであり、簡潔にするために省略されています。

```

Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off   File Bytes   Option   Description                               Value
0     030101       3        Network Access Control                   On
3     041F         4        Class of Service
5     010101       4.1      Class ID                                 1
8     02040000FA00 4.2      Maximum Downstream Rate                  128000 bits/sec
14    03040000FA00 4.3      Maximum Upstream Rate                    64000 bits/sec
20    040101       4.4      Upstream Channel Priority                 1
...
252   06108506547F 6        CM MIC Configuration Setting             8506547FC9152B44
      C9152B44DB95                               DB955420843EF6FE
      5420843EF6FE
270   0710644B675B 7        CMTS MIC Configuration Setting          644B675B70B7BD3E
      70B7BD3E09AC                               09AC210F794A1E8F
      210F794A1E8F
288   FF           255     End-of-Data Marker
289   00           0        PAD
290   00           0        PAD
291   00           0        PAD

0 error(s), 0 warning(s) detected. Parsing of unprov.tmpl was successful.
The file unprov.tmpl was parsed successfully in 375 ms.
The parser initialization time was 63 ms.
The parser parse time was 312 ms.

```

ローカル テンプレート ファイルのテンプレート処理のテストと共有秘密情報の追加

`runCfgUtil.sh` コマンドを使用して、テンプレート ファイルの処理をテストし、指定する共有秘密情報を追加します。

シンタックスの説明 `runCfgUtil.sh -e -docsis -l file -c secret`

- `-e` : 符号化オプションを示します。
- `-docsis` : 入力ファイルが DOCSIS テンプレート ファイルであることを示します。
- `-l` : 入力ファイルがローカル ファイル システムにあることを指定します。
- `file` : 解析する入力テンプレート ファイルを示します。
- `-c` : DOCSIS テンプレート ファイルを解析するときの CMTS 共有秘密情報を指定します。
- `secret` : 新しい共有秘密情報を示します。デフォルトの共有秘密情報は `cisco` です。

ローカルに保存したテンプレート ファイルを解析し、ユーザ固有の共有秘密情報を設定するには、次の手順に従います。

ステップ 1 `/opt/CSCObac/rdu/templates` にディレクトリを変更します。

ステップ 2 解析するテンプレート ファイルを選択します。この例では、既存のテンプレート ファイル `unprov.tmpl` を使用します。これは DOCSIS テンプレート であるため、`-docsis` オプションを使用します。

ステップ 3 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -e -docsis -l unprov.tmpl -c shared
```

- **unprov.tmpl** : ローカルファイルシステムの入力ファイルを示します。
- **shared** : 新しい共有秘密情報を示します。

ユーティリティを実行すると、次のような結果が表示されます。

```
Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off   File Bytes      Option   Description                               Value
-----
0     030100          3       Network Access Control                   Off
3     041F            4       Class of Service
5     010101          4.1     Class ID                                 1
8     02040001F400    4.2     Maximum Downstream Rate                  128000 bits/sec
14    03040000FA00    4.3     Maximum Upstream Rate                    64000 bits/sec
20    040101          4.4     Upstream Channel Priority                 1
...
252   06108506547F    6       CM MIC Configuration Setting             8506547FC9152B44
      C9152B44DB95                                         DB955420843EF6FE
      5420843EF6FE
270   0710644B675B    7       CMTS MIC Configuration Setting           644B675B70B7BD3E
      70B7BD3E09AC                                         09AC210F794A1E8F
      210F794A1E8F
288   FF              255     End-of-Data Marker
289   00              0       PAD
290   00              0       PAD
291   00              0       PAD

0 error(s), 0 warning(s) detected. Parsing of unprov.tmpl was successful.
The file unprov.tmpl was parsed successfully in 375 ms.
The parser initialization time was 63 ms.
The parser parse time was 312 ms.
```

コマンドラインでのマクロ変数の指定

`runCfgUtil.sh` コマンドを使用して、マクロ変数を指定します。

シンタックスの説明 `runCfgUtil.sh -e -l file -m "macros"`

- `-e` : 符号化オプションを示します。
- `-l` : 入力ファイルがローカルファイルシステムにあることを指定します。
- `file` : 解析する入力テンプレートファイルを示します。
- `-m` : テンプレートを解析するときに代入するマクロ変数を指定します。
- `"macros"` : 目的のマクロを示します。複数のマクロ変数が必要な場合は、各マクロの間に2つのカンマを挿入します。

コマンドラインでマクロ変数の値を指定するには、次の手順に従います。

- ステップ 1** `/opt/CSCObac/rdu/templates` にディレクトリを変更します。
- ステップ 2** 使用するテンプレートファイルを選択します。
- ステップ 3** テンプレートのマクロ変数を特定します。この例のマクロ変数は、`macro1` (option 3) と `macro11` (option 4.2) です。
- ステップ 4** マクロ変数の値を特定します。`macro1` の値を 1 に設定し、`macro11` の値を 64000 に設定します。
- ステップ 5** 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -e -l macro.tmpl -m "macro1=1,,macro11=64000"
```

- `macro.tmpl` : 入力ファイルを示します。
- `macro1=1,,macro11=64000` : マクロ変数のキーと値のペアを示します。複数のマクロ変数が必要なため、キーと値のペアの間に2つのカンマを挿入して区切ります。

ユーティリティを実行すると、次のような結果が表示されます。

```
Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off      File Bytes      Option  Description                               Value
-----
0        030101          3      Network Access Control                   On
3        041F            4      Class of Service
5        010101          4.1    Class ID                                 1
8        02040000FA00    4.2    Maximum Downstream Rate                  64000 bits/sec
14       03040000FA00    4.3    Maximum Upstream Rate                   64000 bits/sec
20       040101          4.4    Upstream Channel Priority                1
...

0 error(s), 0 warning(s) detected. Parsing of macro.tmpl was successful.
The file macro.tmpl was parsed successfully in 854 ms.
The parser initialization time was 76 ms.
The parser parse time was 778 ms.
```


マクロ変数のデバイスの指定

`runCfgUtil.sh` コマンドを使用して、マクロ変数のデバイスを指定します。

シンタックスの説明 `runCfgUtil.sh -e -r file -i MAC -u username -p password`

- `-e` : 符号化オプションを示します。
- `-r` : 入力ファイルが RDU に追加したファイルであることを示します。
- `file` : 解析する入力テンプレート ファイルを示します。
- `-i` : マクロ変数を解析するときに使用するデバイスを指定します。
- `MAC` : デバイスの MAC アドレスを示します。
- `-u username` : RDU に接続するときに使用するユーザ名を指定します。
- `-p password` : RDU に接続するときに使用するパスワードを指定します。

マクロ変数の代入に使用するデバイスを指定するには、次の手順に従います。

-
- ステップ 1** 使用するテンプレート ファイルを選択します。この例では、既存のテンプレート ファイル `macro.tmpl` を使用します。
- ステップ 2** テンプレートのマクロ変数を特定します。この例のマクロ変数は、`macro1` (option 3) と `macro11` (option 4.2) です。
- ステップ 3** 使用するデバイスを調べます。この例では、デバイスが RDU に存在し、マクロ変数がプロパティとして設定されているものとします。`macro1` の値を 1 に設定し、`macro11` の値を 64000 に設定します。
- ステップ 4** 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -e -r macro.tmpl -i "1,6,00:01:02:03:04:05" -u admin -p changeme
```

- `macro.tmpl` : 入力ファイルを示します。
- `1,6,00:01:02:03:04:05` : デバイスの MAC アドレスを示します。この MAC アドレスは、例として示す目的でのみ使用しています。
- `admin` : デフォルトのユーザ名を示します。
- `changeme` : デフォルトのパスワードを示します。

ユーティリティを実行すると、次のような結果が表示されます。

```

Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off      File Bytes      Option      Description      Value
0        030101          3           Network Access Control      On
3        041F            4           Class of Service
5        010101          4.1        Class ID          1
8        02040000FA00    4.2        Maximum Downstream Rate      64000 bits/sec
14       03040000FA00    4.3        Maximum Upstream Rate        64000 bits/sec
20       040101          4.4        Upstream Channel Priority     1
...

0 error(s), 0 warning(s) detected. Parsing of macro.tmpl was successful.
The file macro.tmpl was parsed successfully in 159 ms.
The parser initialization time was 42 ms.
The parser parse time was 117 ms.

```

バイナリ ファイルへの出力の指定

runCfgUtil.sh コマンドを使用して、解析するテンプレートの出力をバイナリ ファイルとして指定します。

シンタックスの説明 *runCfgUtil.sh -I input_file -o output_file*

- **-I** : 入力ファイルがローカル ファイル システムにあることを指定します。
- **input_file** : 解析する入力テンプレート ファイルを示します。
- **-o** : 解析するテンプレート ファイルをバイナリ ファイルとして保存することを指定します。
- **output_file** : 解析するテンプレート ファイルのバイナリ コンテンツを格納するファイル名を示します。

テンプレートを解析してバイナリ ファイルに出力するように指定するには、次の手順に従います。

- ステップ 1** `/opt/CSCObac/rdu/templates` にディレクトリを変更します。
- ステップ 2** 使用するテンプレート ファイルを選択します。
- ステップ 3** 出力ファイルの名前を指定します。この例では `unprov.cm` を使用します。
- ステップ 4** 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -I unprov.tmpl -o unprov.cm
```

- **unprov.tmpl** : バイナリ ファイルに解析結果を出力する、既存のテンプレート ファイルを示します。
- **unprov.cm** : 使用する出力ファイル名を示します。

ユーティリティを実行すると、次のような結果が表示されます。

```
Broadband Access Center Configuration Utility
Version: 4.0

0 error(s), 0 warning(s) detected. Parsing of unprov.tmpl was successful.
The file unprov.tmpl was parsed successfully in 595 ms.
The parser initialization time was 262 ms.
The parser parse time was 333 ms.
```

ローカル バイナリ ファイルの表示

runCfgUtil.sh コマンドを使用して、ローカル システムに格納されているバイナリ ファイルを表示します。

シンタックスの説明 *runCfgUtil.sh -d -l file*

- **-d** : このコマンドでバイナリ入力ファイルを表示するためにデコードすることを指定します。
- **-l** : 入力ファイルがローカル ファイル システムにあることを示します。
- **file** : 表示する既存のバイナリ入力ファイルを示します。

ローカル ファイル システムにあるバイナリ ファイルを表示するには、次の手順に従います。

ステップ 1 /opt/CSCObac/rdu/samples/packet_cable にディレクトリを変更します。

ステップ 2 表示するバイナリ ファイルを選択します。

ステップ 3 次のコマンドを使用して、設定ファイル ユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -d -l unprov_packet_cable.bin
```

unprov_packet_cable.bin : 表示する既存のバイナリ入力ファイルを示します。

ユーティリティを実行すると、次のような結果が表示されます。

```

Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Warning: Expecting config file of type docsis, but input file is of type pktc1.0.
Decoding as pktc1.0

Off   File Bytes      Option  Description          Value
0     FE0101           254    Telephony Config File 1
      Start/End
3     0B153013060E    11     SNMP MIB Object      .iso.org.dod.internet.privat
      2B06010401A30B02    e.enterprises.cableLabs.clab
      0201010107000201    Project.clabProjPacketCable.
      02                   pktcMtaMib.pktcMtaMibObjects
                        .pktcMtaDevBase.pktcMtaDevEn
                        abled.0,INTEGER,fals e(2)

...

```



(注) この例の警告は、デフォルトの入力ファイルが DOCSIS で、この例ではバイナリ PacketCable ファイルを使用しているために表示されます。**-pkt** オプションを使用して、入力ファイルを PacketCable ファイルとして指定する場合、警告は表示されません。次に例を示します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -d -pkt -l unprov_packet_cable.bin
```

外部バイナリ ファイルの表示

runCfgUtil.sh コマンドを使用して、外部バイナリ ファイルを表示します。

シンタックスの説明 `runCfgUtil.sh -d -r file -u username -p password`

- **-d** : このコマンドでバイナリ入力ファイルを表示するためにデコードすることを指定します。
- **-r** : 入力ファイルが RDU に追加したファイルであることを示します。
- **file** : RDU にある既存のバイナリ ファイルを示します。
- **-u username** : RDU に接続するとき使用するユーザ名を指定します。
- **-p password** : RDU に接続するとき使用するパスワードを指定します。

RDU に追加したバイナリ ファイルを表示するには、次の手順に従います。

ステップ 1 表示するバイナリ ファイルを選択します。この例では、既存のバイナリ ファイル `unprov.cm` を使用し、RDU が `localhost:49187` であることを前提にしています。

ステップ 2 次のコマンドを使用して、設定ファイル ユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -d -r unprov.cm -u admin -p changeme
```

- **unprov.cm** : RDU にある既存のバイナリ ファイルを示します。
- **admin** : デフォルトのユーザ名を示します。
- **changeme** : デフォルトのパスワードを示します。

ユーティリティを実行すると、次のような結果が表示されます。

```

Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off   File Bytes      Option   Description                               Value
0     030100             3       Network Access Control                    Off
3     041F               4       Class of Service
5     010101             4.1     Class ID                                  1
8     02040001F400       4.2     Maximum Downstream Rate                  128000 bits/sec
14    03040000FA00       4.3     Maximum Upstream Rate                   64000 bits/sec
20    040101             4.4     Upstream Channel Priority                1
...
252   06108506547F       6       CM MIC Configuration Setting             8506547FC9152B44
      C9152B44DB95
      5420843EF6FE
270   0710644B675B       7       CMTS MIC Configuration Setting          644B675B70B7BD3E
      70B7BD3E09AC
      210F794A1E8F
288   FF                255    End-of-Data Marker
289   00                0      PAD
290   00                0      PAD
291   00                0      PAD

0 error(s), 0 warning(s) detected. Parsing of unprov.tpl was successful.
The file unprov.tpl was parsed successfully in 375 ms.
The parser initialization time was 63 ms.
The parser parse time was 312 ms.

```

PacketCable Basic フローの有効化

`runCfgUtil.sh` コマンドを使用して、PacketCable Basic フローの完全性ハッシュを生成し、BASIC フローの静的設定ファイルに挿入することをサポートします。

シンタックスの説明

```
runCfgUtil.sh -t {basic | secure} -pkt -r filename -u username -p password
```

- **basic** : PacketCable Basic フローの完全性ハッシュを計算し、MTA 静的設定ファイルに挿入します。
- **secure** : PacketCable Basic フローの完全性ハッシュを MTA 静的設定ファイルに挿入しません。これはデフォルト設定です。
- **-r** : 入力ファイルが RDU に追加したファイルであることを示します。
- **filename** : 入力ファイルを示します。
- **-u username** : RDU に接続するとき使用するユーザ名を指定します。
- **-p password** : RDU に接続するとき使用するパスワードを指定します。
- **-pkt** : 入力ファイルが PacketCable MTA 設定ファイルであることを示します。

PacketCable Basic フローの完全性ハッシュを生成し、Basic フローの静的設定ファイルに挿入することをサポートするには、次の手順に従います。

- ステップ 1** PacketCable Basic フローの完全性ハッシュを挿入する Basic フローの静的設定ファイルを選択します。この例では `example_mta_config.tpl` を使用します。

ステップ 2 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -t basic -pkt -r example_mta_config.tmpl -u admin
-p changeme
```

- **example_mta_config.tmpl** : Basic フローの静的設定ファイルを示します。
- **admin** : デフォルトのユーザ名を示します。
- **changeme** : デフォルトのパスワードを示します。

ユーティリティを実行すると、次のような結果が表示されます。

```
Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off   File Bytes   Option   Description   Value
-----
0     FE0101       254     Telephony Config File Start/End     1

3     0B153013060E 11      SNMP MIB Object   .iso.org.dod.internet.privat
e.enterprises.cableLabs.clab
Project.clabProjPacketCable.
pktcMtaMib.pktcMtaMibObjects
.pktcMtaDevBase.pktcMtaDevEn
abled.0, INTEGER, true(1)

26    0B2530230610 11      SNMP MIB Object   .iso.org.dod.internet.privat
e.enterprises.cableLabs.clab
Project.clabProjPacketCable.
pktcSigMib.pktcSigMibObjects
.pktcNcsEndPntConfigObjects.
pktcNcsEndPntConfigTable.pkt
cNcsEndPntConfigEntry.pktcNc
sEndPntConfigCallAgentId.9,S
TRING,CMS.IPFONIX.COM

...

371   FE01FF       254     Telephony Config File Start/End     255

0 error(s), 0 warning(s) detected. Parsing of example_mta_config.tmpl was successful.
The file example_mta_config.tmpl was parsed successfully in 100 ms.
The parser initialization time was 44 ms.
The parser parse time was 56 ms.
```

.tmpl 拡張子付きのファイルは、動的設定テンプレートであると見なされます。このテンプレートに対し、テンプレート処理中に Basic ハッシュの計算と挿入が透過的に発生します。その結果、同じテンプレートを Secure モードと Basic モードのプロビジョニングで使用できます。

ただし、ハッシュを挿入する前に、Secure 静的バイナリ設定ファイルを Basic 静的設定ファイルに変換する場合は、次の手順に従います。

- 次のコマンドを使用して、Secure 静的ファイルをテンプレートに変換します。

```
# runCfgUtil -l input_static_filename -pkt -g -o output_template_filename
```

- 次のコマンドを使用して、Secure 静的テンプレートを Basic 静的設定ファイルに変換します。

```
# runCfgUtil -t basic -l input_template_name -pkt -o output_Basic_static_filename
```

このコマンドは、Basic の完全性ハッシュを計算して、Basic 静的設定ファイルに挿入します。

マルチベンダーをサポートするための TLV 43 の生成

`runCfgUtil.sh` コマンドを使用して、マルチベンダーのサポートを提供するために TLV 43 を生成します。

シンタックスの説明

```
runCfgUtil.sh -docsis -r filename -u username -p password
```

- `-docsis` : 入力ファイルが DOCSIS テンプレート ファイルであることを示します。
- `filename` : 解析する入力テンプレート ファイルを示します。
- `-r` : 入力ファイルが RDU に追加したファイルであることを示します。
- `-u username` : RDU に接続するとき使用するユーザ名を指定します。
- `-p password` : RDU に接続するとき使用するパスワードを指定します。

RDU に追加したテンプレート ファイルを使用して TLV 43 を生成するには、次の手順に従います。

ステップ 1 使用するテンプレート ファイルを選択します。この例では、既存のテンプレート ファイル `test.tmpl` を使用します。DOCSIS テンプレートを使用するため、`-docsis` オプションを使用します。

ステップ 2 次のコマンドを使用して、設定ファイルユーティリティを実行します。

```
/opt/CSCObac/rdu/bin# runCfgUtil.sh -docsis -r test.tmpl -u admin -p changeme
```

- `test.tmpl` : DOCSIS 設定ファイルを示します。
- `admin` : デフォルトのユーザ名を示します。
- `changeme` : デフォルトのパスワードを示します。

ユーティリティを実行すると、次のような結果が表示されます。

```

Broadband Access Center Configuration Utility
Version: 4.0, Revision: 1.26

Off  File Bytes      Option  Description                               Value
-----
0    2B14              43      DOCSIS Extension Field                    FF-FF-FF
2    0803FFFFFF        43.8    Vendor ID                                 0234560003
7    050D              43.5    L2VPN Encoding
9    010502345600     43.5.1  VPNID Subtype
    03
16   0204              43.5.2  NSI Encapsulation Subtype
18   02020019         43.5.2.2 IEEE 802.1Q Format Subtype                25
22   2B0B              43      DOCSIS Extension Field
24   080300000C       43.8    Vendor ID                                 00-00-0C (CISCO
    SYSTEMS, INC.)
29   010418017A30     43.1    Static Downstream Frequency              402750000
35   2B1D              43      DOCSIS Extension Field
37   080300000C       43.8    Vendor ID                                 00-00-0C (CISCO
    SYSTEMS, INC.)
42   0316626F6F74     43.3    Update Boot Monitor Image                boot_monitor_image.bin
    5F6D6F6E6974
    6F725F696D61
    67652E62696E
66   061071E79068     6        CM MIC Configuration Setting              71E790683DE8B995
    3DE8B9950536
    8936F4C5312F
84   0710DB0EED14     7        CMTS MIC Configuration Setting            DB0EED14B5B3428D
    B5B3428D2B15
    ODA582B41A54
102  FF                255     End-of-Data Marker
103  00                0        PAD

0 error(s), 0 warning(s) detected. Parsing of test.tpl was successful.
The file test.tpl was parsed successfully in 250 ms.
The parser initialization time was 109 ms.
The parser parse time was 141 ms.

```
