

Como adicionar, modificar e remover VLANs em um Catalyst usando SNMP

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Componentes](#)

[Conventions](#)

[Background](#)

[Detalhes das variáveis MIB — incluindo identificadores de objeto \(OIDs\)](#)

[Adicionar uma VLAN a um switch Cisco Catalyst com SNMP](#)

[Step-by-Step Instructions](#)

[Adicionar uma VLAN a um switch Cisco Catalyst com SNMP](#)

[Instruções de uma etapa](#)

[Excluir uma VLAN de um switch Cisco Catalyst com SNMP](#)

[Step-by-Step Instructions](#)

[Adicionar uma porta a uma VLAN em um switch Cisco Catalyst com SNMP](#)

[Como alterar uma porta de uma VLAN para outra VLAN](#)

[Informações Relacionadas](#)

[Introduction](#)

Este documento descreve como criar e excluir VLANs em um switch Cisco Catalyst que usa o Simple Network Management Protocol (SNMP). Ele também descreve como adicionar portas a uma VLAN com SNMP.

[Prerequisites](#)


[Requirements](#)

Antes de usar as informações neste documento, certifique-se de que entende:

- Como funciona o ifTable e ifIndexes
- Como as VLANs funcionam nos switches Cisco Catalyst
- Como visualizar informações de VLAN nos switches Cisco Catalyst
- O uso geral dos comandos **get**, **set** e **walk** SNMP

[Componentes](#)

Este documento destina-se aos Catalyst Switches que executam Catalyst OS ou Catalyst IOS regulares que suportam IF-MIB, CISCO-VTP-MIB e CISCO-VLAN-MEMBERSHIP-MIB. As informações neste documento são baseadas nestas versões de software e hardware:

- Catalyst 3524XL executando CatIOS 12.0(5)WC5a
- NET-SNMP version 5.0.6 available at <http://www.net-snmp.org/> 

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. All of the devices used in this document started with a cleared (default) configuration. Se você estiver trabalhando em uma rede ativa, antes de usar qualquer comando, certifique-se de que entende o impacto potencial de qualquer comando.

Conventions

Para obter mais informações sobre convenções de documento, consulte as [Convenções de dicas técnicas Cisco](#).

Background

Detalhes das variáveis MIB — incluindo identificadores de objeto (OIDs)

1.3.6.1.4.1.9.9.46.1.3.1.1.2 (CISCO-VTP-MIB)

```
vtpVlanState OBJECT-TYPE
    SYNTAX      INTEGER { operational(1),
                        suspended(2),
                        mtuTooBigForDevice(3),
                        mtuTooBigForTrunk(4) }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION  "The state of this VLAN.
```

The state 'mtuTooBigForDevice' indicates that this device cannot participate in this VLAN because the VLAN's MTU is larger than the device can support.

The state 'mtuTooBigForTrunk' indicates that while this VLAN's MTU is supported by this device, it is too large for one or more of the device's trunk ports."

```
::= { vtpVlanEntry 2 }
```

1.3.6.1.4.1.9.9.46.1.4.1.1.1 (CISCO-VTP-MIB)

```
vtpVlanEditOperation OBJECT-TYPE
    SYNTAX      INTEGER { none(1),
                        copy(2),
                        apply(3),
                        release(4),
                        restartTimer(5)
                    }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION  "This object always has the value 'none' when read. When
                written, each value causes the appropriate action:
```

'copy' - causes the creation of rows in the vtpVlanEditTable exactly corresponding to the current global

VLAN information for this management domain. If the Edit Buffer (for this management domain) is not currently empty, a copy operation fails. A successful copy operation starts the deadman-timer.

'apply' - first performs a consistent check on the the modified information contained in the Edit Buffer, and if consistent, then tries to instanciate the modified information as the new global VLAN information. Note that an empty Edit Buffer (for the management domain) would always result in an inconsistency since the default VLANs are required to be present.

'release' - flushes the Edit Buffer (for this management domain), clears the Owner information, and aborts the deadman-timer. A release is generated automatically if the deadman-timer ever expires.

'restartTimer' - restarts the deadman-timer.

'none' - no operation is performed."

```
::= { vtpEditControlEntry 1 }
```

1.3.6.1.4.1.9.9.46.1.4.1.1.3 (CISCO-VTP-MIB)

vtpVlanEditBufferOwner OBJECT-TYPE

SYNTAX OwnerString

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The management station which is currently using the Edit Buffer for this management domain. When the Edit Buffer for a management domain is not currently in use, the value of this object is the zero-length string. Note that it is also the zero-length string if a manager fails to set this object when invoking a copy operation."

```
::= { vtpEditControlEntry 3 }
```

1.3.6.1.4.1.9.9.46.1.4.2.1.11 (CISCO-VTP-MIB)

vtpVlanEditRowStatus OBJECT-TYPE

SYNTAX RowStatus

1:active

2:notInService

3:notReady

4:createAndGo

5:createAndWait

6:destroy

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The status of this row. Any and all columnar objects in an existing row can be modified irrespective of the status of the row.

A row is not qualified for activation until instances of at least its vtpVlanEditType, vtpVlanEditName and vtpVlanEditDot10Said columns have appropriate values.

The management station should endeavor to make all rows consistent in the table before 'apply'ing the buffer. An inconsistent entry in the table will cause the entire buffer to be rejected with the vtpVlanApplyStatus object set to the appropriate error value."

```
::= { vtpVlanEditEntry 11 }
```

1.3.6.1.4.1.9.9.46.1.4.2.1.3.1.48 (CISCO-VTP-MIB)

vtpVlanEditType OBJECT-TYPE

SYNTAX VlanType

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The type which this VLAN would have.
An implementation may restrict access to this object."

DEFVAL { ethernet }

::= { vtpVlanEditEntry 3 }

1.3.6.1.4.1.9.9.46.1.4.2.1.4.1.48 (CISCO-VTP-MIB)

vtpVlanEditName OBJECT-TYPE

SYNTAX DisplayString (SIZE (1..32))

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The name which this VLAN would have. This name would be
used as the ELAN-name for an ATM LAN-Emulation segment of
this VLAN.

An implementation may restrict access to this object."
::= { vtpVlanEditEntry 4 }

1.3.6.1.4.1.9.9.46.1.4.2.1.6.1.48 (CISCO-VTP-MIB)

vtpVlanEditDot10Said OBJECT-TYPE

SYNTAX OCTET STRING (SIZE (4))

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The value of the 802.10 SAID field which would be used for
this VLAN.

An implementation may restrict access to this object."
::= { vtpVlanEditEntry 6 }

1.3.6.1.4.1.9.9.46.1.4.1.1.2.1 (CISCO-VTP-MIB)

vtpVlanApplyStatus OBJECT-TYPE

SYNTAX INTEGER { inProgress(1),
succeeded(2),
configNumberError(3),
inconsistentEdit(4),
tooBig(5),
localNVStoreFail(6),
remoteNVStoreFail(7),
editBufferEmpty(8),
someOtherError(9)
}

MAX-ACCESS read-only

STATUS current

DESCRIPTION "The current status of an 'apply' operation to instantiate
the Edit Buffer as the new global VLAN information (for this
management domain). If no apply is currently active, the
status represented is that of the most recently completed
apply. The possible values are:

inProgress - 'apply' operation in progress;

succeeded - the 'apply' was successful (this value is
also used when no apply has been invoked since the
last time the local system restarted);

configNumberError - the apply failed because the value of

```

vtpVlanEditConfigRevNumber was less or equal to
the value of current value of
managementDomainConfigRevNumber;

inconsistentEdit - the apply failed because the modified
information was not self-consistent;

tooBig - the apply failed because the modified
information was too large to fit in this VTP
Server's non-volatile storage location;

localNVStoreFail - the apply failed in trying to store
the new information in a local non-volatile
storage location;

remoteNVStoreFail - the apply failed in trying to store
the new information in a remote non-volatile
storage location;

editBufferEmpty - the apply failed because the Edit
Buffer was empty (for this management domain).

someOtherError - the apply failed for some other reason
(e.g., insufficient memory)."
 ::= { vtpEditControlEntry 2 }

```

1.3.6.1.4.1.9.9.68.1.2.2.1.2 (CISCO-VLAN-MEMBERSHIP-MIB)

```

vmVlan OBJECT-TYPE
    SYNTAX      INTEGER(0..4095)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The VLAN id of the VLAN the port is assigned to
        when vmVlanType is set to static or dynamic.
        This object is not instantiated if not applicable.

        The value may be 0 if the port is not assigned
        to a VLAN.

        If vmVlanType is static, the port is always
        assigned to a VLAN and the object may not be
        set to 0.

        If vmVlanType is dynamic the object's value is
        0 if the port is currently not assigned to a VLAN.
        In addition, the object may be set to 0 only."
    ::= { vmMembershipEntry 2 }

```

[Adicionar uma VLAN a um switch Cisco Catalyst com SNMP](#)

[Step-by-Step Instructions](#)

No exemplo mostrado abaixo, a VLAN 11 é adicionada ao switch:

1. Para verificar quais VLANs estão configuradas atualmente no switch, emita um **snmpwalk** no OID **vtpVlanState**: **Observação:** o último número no OID é o número da VLAN.

```

snmpwalk -c public crumpy vtpVlanState
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1

```

```
.1 : INTEGER: operational
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1
.48 : INTEGER: operational
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1
.1002 : INTEGER: operational
```

2. Verifique se a edição está sendo usada por outra estação ou dispositivo NMS. A edição não está em uso se você vir esta mensagem: nenhum objeto MIB contido na subárvore:

```
snmpwalk -c public crumpy vtpVlanEditTable
no MIB objects contained under subtree.
```

3. A edição não está em uso, portanto é seguro começar a editar. Defina **vtpVlanEditOperation** para o estado de cópia (inteiro 2). Isso permite criar a VLAN.

```
snmpset -c private crumpy vtpVlanEditOperation.1 integer 2
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpEditControlTable.vtpEditControlEntry.vtpVlanEditOperation.1 : INTEGER: copy
```

4. Para tornar visível o proprietário atual da permissão de edição, você pode definir o proprietário ao emitir o comando **vtpVlanEditBufferOwner**.

```
snmpset -c private crumpy vtpVlanEditBufferOwner.1 octetstring "Gerald"
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpEditControlTable.vtpEditControlEntry.vtpVlanEditBufferOwner.1 : OCTET STRING- (ascii): Gerald
```

5. Este exemplo mostra como verificar se a tabela existe:

```
snmpwalk -c public crumpy vtpVlanEditTable
vtpVlanEditState.1.1 : INTEGER: operational
vtpVlanEditState.1.2 : INTEGER: operational
vtpVlanEditState.1.3 : INTEGER: operational
..
```

6. Este exemplo é a VLAN 11 e mostra como criar uma linha e definir o tipo e o nome:

```
snmpset -c private crumpy vtpVlanEditRowStatus.1.11 integer 4
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpVlanEditTable.vtpVlanEditEntry.vtpVlanEditRowStatus.1.11 : INTEGER: createAndGo
```

```
snmpset -c private crumpy vtpVlanEditType.1.11 integer 1
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpVlanEditTable.vtpVlanEditEntry.vtpVlanEditType.1.11 : INTEGER: ethernet
```

```
snmpset -c private crumpy vtpVlanEditName.1.11 octetstring "test_11_gerald"
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpVlanEditTable.vtpVlanEditEntry.vtpVlanEditName.1.11 : DISPLAY STRING- (ascii): test_11_gerald
```

7. Defina **vtpVlanEditDot10Said**. Esse é o número da VLAN + 100000 traduzido para hexadecimal. Este exemplo cria a VLAN 11, então o **vtpVlanEditDot10Said** deve ser: $11 + 100000 = 100011 \rightarrow$ Hexa: 000186AB

```
snmpset -c private crumpy vtpVlanEditDot10Said.1.11 octetstringhex 000186AB
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpVlanEditTable.vtpVlanEditEntry.vtpVlanEditDot10Said.1.11 : OCTET STRING- (hex): length = 4
0: 00 01 86 ab -- -- -- -- -- -- -- -- -- -- .....
```

8. Depois de criar a VLAN 11, você deve aplicar as modificações. Use o OID **vtpVlanEditOperation** novamente. Desta vez, use o comando **Apply** para confirmar as configurações:

```
snmpset -c private crumpy vtpVlanEditOperation.1 integer 3
```

```
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpEditControlTable.vtpEditControlEntry.vtpVlanEditOperation.1 : INTEGER: apply
```

9. Verifique se a VLAN foi criada com êxito. Use o OID **vtpVlanApplyStatus**. Verifique o processo até que o status seja: êxito:

```
snmpget -c public crumpy vtpVlanApplyStatus.1
vtpVlanApplyStatus.1 : INTEGER: inProgress
snmpget -c public crumpy vtpVlanApplyStatus.1
vtpVlanApplyStatus.1 : INTEGER: inProgress
snmpget -c public crumpy vtpVlanApplyStatus.1
vtpVlanApplyStatus.1 : INTEGER: succeeded
```

10. A última ação é confirmar as modificações e liberar as permissões para que outros usuários possam adicionar, modificar ou excluir VLANs de seu NMS.

```
snmpset -c private crumpy vtpVlanEditOperation.1 integer 4
vtpVlanEditOperation.1 : INTEGER: release
```

11. Verifique se o buffer está vazio:

```
snmpwalk -c public crumpy vtpVlanEditTable
no MIB objects contained under subtree.
```

12. Verifique se a VLAN 11 foi criada no switch com o comando CLI **show vlan** ou com um **snmpwalk**:

```
snmpwalk -c public crumpy vtpVlanState
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1.1 : INTEGER: operational
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1.11 : INTEGER: operational
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1.48 : INTEGER: operational
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1.1002 : INTEGER: operational
...
```

[Adicionar uma VLAN a um switch Cisco Catalyst com SNMP](#)

[Instruções de uma etapa](#)

O processo de uma etapa usa os números OID em vez dos nomes OID como o processo passo a passo anterior. Veja os [detalhes da MIB](#) para tradução. Este exemplo cria a VLAN 6:

```
snmpset -c private crumpy 1.3.6.1.4.1.9.9.46.1.4.1.1.1.1 integer 2
1.3.6.1.4.1.9.9.46.1.4.1.1.3.1 octetstring "gcober"
```

```
snmpset -c private gooroo 1.3.6.1.4.1.9.9.46.1.4.2.1.11.1.6 integer 4
1.3.6.1.4.1.9.9.46.1.4.2.1.3.1.6 integer 1 1.3.6.1.4.1.9.9.46.1.4.2.1.4.1.6 octetstring "vlan6"
1.3.6.1.4.1.9.9.46.1.4.2.1.6.1.6 octetstringhex 000186A6 1.3.6.1.4.1.9.9.46.1.4.1.1.1.1 integer 3
```

```
snmpset -c private gooroo 1.3.6.1.4.1.9.9.46.1.4.1.1.1.1 integer 4
```

```
snmpwalk -c public crumpy 1.3.6.1.4.1.9.9.46.1.3.1.1.2
```

```
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1.1 :  
INTEGER: operational  
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1.6 :  
INTEGER: operational  
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1.11 :  
INTEGER: operational
```

Observação: determinadas versões SNMP exigem que você use um (.) antes do OID nos comandos SNMP SET.

[Excluir uma VLAN de um switch Cisco Catalyst com SNMP](#)

[Step-by-Step Instructions](#)

Neste exemplo, a VLAN 48 é excluída do switch. Consulte [Adicionar uma VLAN a um Cisco Catalyst com SNMP](#) para obter mais informações. A diferença entre esta seção em que você exclui uma VLAN e aquela em que você adiciona uma VLAN é que você usa o **destruidor** em vez do comando **CreateAndGo** para o **vtpVlanEditRowStatus**:

1. Emita o comando para excluir a VLAN 48:

```
snmpset -c private crumpy vtpVlanEditOperation.1 integer 2  
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpEditControlTable.vtpEditControlEntry.  
vtpVlanEditOperation.1 : INTEGER: copy  
snmpset -c private crumpy vtpVlanEditRowStatus.1.48 integer 6  
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpVlanEditTable.vtpVlanEditEntry.vtpVla  
nEditRowStatus.1.48 : INTEGER: destroy
```

2. Para verificar se a VLAN 48 foi excluída, use **vtpVlanState** ou **show vlan** na CLI:

```
snmpwalk -c public crumpy vtpVlanState  
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1  
.1 : INTEGER: operational  
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanInfo.vtpVlanTable.vtpVlanEntry.vtpVlanState.1  
.1002 : INTEGER: operational  
...
```

[Adicionar uma porta a uma VLAN em um switch Cisco Catalyst com SNMP](#)

Este exemplo mostra como adicionar uma porta Fast Ethernet 0/5 à VLAN 48.

1. Para verificar qual ifIndex Fast Eth 0/5 possui, emita um **snmpwalk** de **ifDescr**:

```
snmpwalk -c public crumpy ifDescr  
...  
interfaces.ifTable.ifEntry.ifDescr.6 : DISPLAY STRING- (ascii): FastEthernet0/5  
...
```

2. Como você sabe que a porta Fast Eth 0/5 tem um ifIndex de 6, adicione a porta à VLAN 48:

```
snmpset -c private crumpy vmVlan.6 integer 48  
cisco.ciscoMgmt.ciscoVlanMembershipMIB.ciscoVlanMembershipMIBObjects.vmMembership.vmMembers  
hipTable.vmMembershipEntry.vmVlan.6 : INTEGER: 48
```



```
48 VLAN0048 active Gi0/2
Fa0/3
```

Após a alteração:

```
crumpy#sh vlan
```

VLAN Name	Status	Ports
1 default	active	Fa0/1, Fa0/2, Fa0/3, Fa0/4, Fa0/5, Fa0/6, Fa0/7, Fa0/8, Fa0/9, Fa0/10, Fa0/11, Fa0/12, Fa0/13, Fa0/14, Fa0/15, Fa0/16, Fa0/17, Fa0/18, Fa0/19, Fa0/20, Fa0/21, Fa0/22, Fa0/23, Fa0/24, Gi0/1, Gi0/2
48 VLAN0048	active	

Observação: você pode fazer outras alterações, como o nome da VLAN, o proprietário e muito mais. Consulte a MIB inteira para obter mais detalhes sobre OID.

[Informações Relacionadas](#)

- [Suporte Técnico - Cisco Systems](#)