

# Come aggiungere, modificare e rimuovere le VLAN su un Catalyst utilizzando SNMP

## Sommario

[Introduzione](#)

[Prerequisiti](#)

[Requisiti](#)

[Componenti](#)

[Convenzioni](#)

[Sfondo](#)

[Dettagli delle variabili MIB, inclusi gli OID \(Object Identifier\)](#)

[Aggiunta di una VLAN a uno switch Cisco Catalyst con SNMP](#)

[Istruzioni dettagliate](#)

[Aggiunta di una VLAN a uno switch Cisco Catalyst con SNMP](#)

[Istruzioni in un unico passaggio](#)

[Eliminazione di una VLAN da uno switch Cisco Catalyst con SNMP](#)

[Istruzioni dettagliate](#)

[Aggiunta di una porta a una VLAN su uno switch Cisco Catalyst con SNMP](#)

[Come modificare una porta da una VLAN a un'altra VLAN](#)

[Informazioni correlate](#)

## [Introduzione](#)

In questo documento viene descritto come creare ed eliminare le VLAN su uno switch Cisco Catalyst che utilizza il protocollo SNMP (Simple Network Management Protocol). Descrive anche come aggiungere porte a una VLAN con SNMP.

## [Prerequisiti](#)


### [Requisiti](#)

Prima di usare le informazioni riportate in questo documento, accertarsi di aver compreso:

- Funzionamento di ifTable e ifIndexes
- Funzionamento delle VLAN sugli switch Cisco Catalyst
- Come visualizzare le informazioni sulla VLAN sugli switch Cisco Catalyst
- Uso generale dei comandi **get**, **set** e **walk** del protocollo SNMP

## [Componenti](#)

Questo documento è destinato agli switch Catalyst che eseguono regolarmente il sistema operativo Catalyst o Catalyst IOS che supporta IF-MIB, CISCO-VTP-MIB e CISCO-VLAN-MEMBERSHIP-MIB. Le informazioni fornite in questo documento si basano sulle seguenti versioni software e hardware:

- Catalyst 3524XL con CatIOS 12.0(5)WC5a
- NET-SNMP versione 5.0.6 disponibile all'indirizzo <http://www.net-snmp.org/> 

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, prima di usare un comando accertarsi di aver ben compreso l'impatto potenziale di ciascun comando.

## Convenzioni

Per ulteriori informazioni sulle convenzioni usate, consultare il documento [Cisco sulle convenzioni nei suggerimenti tecnici](#).

## Sfondo

### Dettagli delle variabili MIB, inclusi gli OID (Object Identifier)

#### 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 instantiate 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 }

```

## [Aggiunta di una VLAN a uno switch Cisco Catalyst con SNMP](#)

### [Istruzioni dettagliate](#)

Nell'esempio seguente, la VLAN 11 viene aggiunta allo switch:

1. Per verificare quali VLAN sono attualmente configurate sullo switch, usare uno **snmpwalk** sull'OID **vtpVlanState**: **Nota**: l'ultimo numero nell'OID è il numero 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. Verificare se l'edizione è utilizzata da un'altra stazione o dispositivo NMS. Se viene visualizzato questo messaggio, l'edizione non è in uso: nessun oggetto MIB contenuto nella sottostruttura:

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

3. L'edizione non è in uso, pertanto è consigliabile iniziare a modificarla. Impostare **vtpVlanEditOperation** sullo stato di copia (integer 2). Ciò consente di creare la VLAN.

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

4. Per rendere visibile il proprietario corrente dell'autorizzazione alla modifica, è possibile impostarlo quando si esegue il comando **vtpVlanEditBufferOwner**.

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

5. In questo esempio viene illustrato come verificare l'esistenza della tabella:

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

6. L'esempio è la VLAN 11 e mostra come creare una riga e impostare il tipo e il nome:

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

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

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

7. Impostare **vtpVlanEditDot10Said**. Il numero VLAN + 100000 è convertito in esadecimale. Nell'esempio viene creata la VLAN 11, quindi il valore di **vtpVlanEditDot10Said** deve essere:

11 + 100000 = 100011 -> Hex: 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. Dopo aver creato la VLAN 11, occorre applicare le modifiche. Usare nuovamente l'OID **vtpVlanEditOperation**. Questa volta, usare il comando **Apply** per confermare le impostazioni:

```
snmpset -c private crumpy vtpVlanEditOperation.1 integer 3
cisco.ciscoMgmt.ciscoVtpMIB.vtpMIBObjects.vlanEdit.vtpEditControlTable.vtpEditControlEntry.vtpVlanEditOperation.1 : INTEGER: apply
```

9. Verificare che la VLAN sia stata creata correttamente. Usare OID **vtpVlanApplyStatus**.

Controllare il processo fino a quando lo stato non è: operazione completata:

```
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. L'ultima azione consiste nel confermare le modifiche e rilasciare le autorizzazioni in modo che altri utenti possano aggiungere, modificare o eliminare le VLAN dal relativo NMS.

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

11. Verificare che il buffer sia vuoto:

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

12. Verificare che la VLAN 11 sia stata creata sullo switch con il comando **show vlan** della CLI o con una **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
...
```

## [Aggiunta di una VLAN a uno switch Cisco Catalyst con SNMP](#)

### [Istruzioni in un unico passaggio](#)

Il processo in un unico passaggio utilizza i numeri OID anziché i nomi OID, come nel processo precedente. Vedere i [dettagli MIB](#) per la traduzione. In questo esempio viene creata la 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
```

**Nota:** alcune versioni di SNMP richiedono l'uso di un punto (.) prima dell'OID nei comandi SET di SNMP.

## [Eliminazione di una VLAN da uno switch Cisco Catalyst con SNMP](#)

### [Istruzioni dettagliate](#)

Nell'esempio, la VLAN 48 viene eliminata dallo switch. Per ulteriori informazioni, consultare il documento sull'[aggiunta di una VLAN a Cisco Catalyst con SNMP](#). La differenza tra questa sezione in cui si elimina una VLAN e quella in cui si aggiunge una VLAN è che si usa il comando **delete** anziché il comando **CreateAndGo** per il parametro **vtpVlanEditRowStatus**:

1. Per eliminare la VLAN 48, eseguire il comando:

```
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. Per verificare che la VLAN 48 sia stata eliminata, usare **vtpVlanState** o **show vlan** sulla 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  
...
```

## [Aggiunta di una porta a una VLAN su uno switch Cisco Catalyst con SNMP](#)

Nell'esempio viene mostrato come aggiungere una porta Fast Ethernet 0/5 alla VLAN 48.

1. Per verificare se dispone di ifIndex Fast Eth 0/5, eseguire uno **snmpwalk** di **ifDescr**:

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

2. Poiché si sa che la porta Fast Eth 0/5 ha un ifIndex pari a 6, aggiungere la porta alla VLAN 48:



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

3. Verificare che la porta sia stata aggiunta correttamente eseguendo nuovamente una query sullo stesso OID.

```
snmpget -c public crumpy vmVlan.6
cisco.ciscoMgmt.ciscoVlanMembershipMIB.ciscoVlanMembershipMIBObjects.vmMembership.vmMembers
hipTable.vmMembershipEntry.vmVlan.6 : INTEGER: 48
```

È possibile verificare questa condizione anche sullo switch:

```
crumpy#sh vlan
```

VLAN Name	Status	Ports	
1	default	active	Fa0/1, Fa0/2, Fa0/3, Fa0/4, 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	Fa0/5

## Come modificare una porta da una VLAN a un'altra VLAN

Nell'esempio viene mostrato come la porta Fast Ethernet 0/3 appartiene alla VLAN 48 e come spostarla sulla VLAN 1 (VLAN predefinita):

1. Per verificare se dispone di ifIndex Fast Eth 0/3, eseguire uno **snmpwalk** di **ifDescr**:

```
snmpwalk -c public crumpy ifDescr
...
interfaces.ifTable.ifEntry.ifDescr.4 : DISPLAY STRING- (ascii): FastEthernet0/3
...
```

2. Poiché si sa che la porta Fast Eth 0/3 ha un ifIndex pari a 4, è possibile verificare a quale VLAN la porta appartiene attualmente:

```
snmpget -c public crumpy vmVlan.4
cisco.ciscoMgmt.ciscoVlanMembershipMIB.ciscoVlanMembershipMIBObjects.vmMembership.vmMembers
hipTable.vmMembershipEntry.vmVlan.4 : INTEGER: 48
```

3. La porta appartiene alla VLAN 48.

```
snmpset -c private crumpy vmVlan.4 integer 1
cisco.ciscoMgmt.ciscoVlanMembershipMIB.ciscoVlanMembershipMIBObjects.vmMembership.vmMembers
hipTable.vmMembershipEntry.vmVlan.4 : INTEGER: 1
```

4. Per spostare la porta dalla VLAN 48 alla VLAN 1, usare un comando **snmpset** di **vmVlan**.

5. Per verificare se la porta è stata modificata sull'altra VLAN, eseguire di nuovo la query su **vmVlan**:

```
snmpget -c public crumpy vmVlan.4
cisco.ciscoMgmt.ciscoVlanMembershipMIB.ciscoVlanMembershipMIBObjects.vmMembership.vmMembers
hipTable.vmMembershipEntry.vmVlan.4 : INTEGER: 1
```

È possibile verificare questa condizione anche sullo switch stesso:Prima della modifica:

```
crumpy#sh vlan
```

VLAN Name	Status	Ports
-----------	--------	-------

```

-----
1    default                active    Fa0/1, Fa0/2, 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    Fa0/3

```

Dopo la modifica:

```
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

```

**Nota:** è possibile apportare altre modifiche, ad esempio il nome della VLAN, il proprietario e molto altro ancora. Fare riferimento all'intero MIB per ulteriori dettagli su OID.

## [Informazioni correlate](#)

- [Supporto tecnico – Cisco Systems](#)