

排除Catalyst交换机上的STP故障

目录

[简介](#)

[先决条件](#)

[要求](#)

[使用的组件](#)

[规则](#)

[背景信息](#)

[STP故障的原因](#)

[排除转发环路故障](#)

- [1. 确定环路](#)
- [2. 发现环路的拓扑\(范围\)](#)
- [3. 中断循环](#)
- [4. 查找并修复环路的原因](#)
- [5. 恢复冗余](#)

[调查拓扑更改](#)

[查找泛洪的原因](#)

[查找TC](#)

[采取步骤来防止 TC 过多](#)

[排除收敛时间相关问题](#)

[使用STP调试命令](#)

[保护网络免受转发环路影响](#)

- [1. 在所有交换机到交换机链路上启用单向链路检测\(UDLD\)](#)
- [2. 在所有交换机上启用环路防护](#)
- [3. 在所有终端站端口上启用Portfast](#)
- [4. 在两侧\(受支持时\)和Non-SilentOption上将EtherChannel设置为DesirableMode](#)
- [5. 请勿禁用交换机到交换机链路上的自动协商\(如果支持\)](#)
- [6. 调整STP计时器时务必谨慎](#)
- [7. 如果可能发生拒绝服务攻击,请使用根防护来保护网络STP周界](#)
- [8. 在启用Portfast的端口上启用BPDU防护,以防止STP受到连接到端口的未经授权网络设备\(如集线器、交换机和桥接路由器\)的影响](#)
- [9. 避免管理VLAN上的用户流量](#)
- [10. 可预测\(硬编码\)的STP根和备份STP根位置](#)

[相关信息](#)

简介

本文档介绍如何使用Cisco IOS®软件来排查生成树协议(STP)的问题。

先决条件

要求

Cisco 建议您了解以下主题：

- 各种生成树类型以及如何配置这些类型。有关详细信息，请参阅[配置STP和IEEE 802.1s MST](#)。
- 各种生成树功能以及如何配置这些功能。有关详细信息，请参阅[配置STP功能](#)。

使用的组件

本文档中的信息基于以下软件和硬件版本：

- 带有 Supervisor 2 引擎的 Catalyst 6500
- Cisco IOS 软件版本 12.1(13)E

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

规则

有关文档规则的详细信息，请参阅思科技术提示规则。

背景信息

有些特定命令仅适用于Catalyst 6500/6000；但是，您可以将大多数原则应用于运行Cisco IOS软件的任何思科Catalyst交换机。

大多数STP的问题有以下三个问题：

- 转发环路。
- 由于STP拓扑更改(TC)率较高而导致的过度泛洪。
- 与收敛时间有关的问题。

因为网桥没有跟踪特定数据包是多次转发（例如，IP生存时间[TTL]）还是用于在网络中丢弃循环时间过长的流量的机制。同一第2层(L2)域中的两台设备之间只能存在一条路径。

STP的目的是根据STP算法阻塞冗余端口，并将冗余物理拓扑解析为树状拓扑。如果未阻塞冗余拓扑中的任何端口，并且在循环中无限地转发流量，则会出现转发环路（如 STP 环路）。

一旦转发环路开始，它就会拥塞其路径上带宽最低的链路。如果所有链路的带宽都相同，则所有链路都会发生拥塞。此拥塞会导致数据包丢失并在受影响的L2域中导致网络中断。

过度泛洪时，症状不明显。慢速链路可能会因泛洪流量而拥塞，这些拥塞链路背后的设备或用户可能会遇到速度缓慢或连接完全丢失的情况。

STP故障的原因

STP 对其运行环境进行了某些假定。以下假定与本文档的相关程度最高：

- 两个网桥之间的每条链路都是双向的。这意味着，如果A直接连接到B，则A接收B已发送的内容，B接收A已发送的内容，只要它们之间的链路处于接通状态。
- 每个运行STP的网桥都能定期接收、处理和传输STP网桥协议数据单元(BPDU)，也称为STP数据包。

虽然这些假设似乎是合乎逻辑且显而易见的，但有些情况下这些假设并未得到满足。其中大多数情况都涉及某种硬件问题；但是，软件缺陷也可能导致STP故障。各种硬件故障、配置错误、连接问题导致大多数STP故障，而软件故障占少数。由于交换机之间存在多余的额外连接，也可能出现STP故障。VLAN 将因这些额外的连接而进入关闭状态。要解决此问题，请删除交换机之间所有不需要的连接。

如果不满足其中一种假设，一个或多个网桥将无法接收或处理BPDU。这意味着网桥（或多个网桥）未发现网络拓扑。如果不知道正确的拓扑，交换机将无法阻塞环路。因此，泛洪流量在环路拓扑上循环，消耗所有带宽，并中断网络。

交换机无法接收BPDU的原因示例包括收发器或千兆接口转换器(GBIC)故障、电缆问题或端口、板卡或Supervisor引擎上的硬件故障。导致STP故障的一个常见原因是网桥之间存在一条单向链路。在这种情况下，一个网桥发送BPDU，但下游网桥从未收到它们。STP处理也可能因CPU过载（99%或更多）而中断，因为交换机无法处理收到的BPDU。BPDU可能在从一个网桥到另一个网桥的路径中损坏，这样也会阻止适当的STP行为。

除了转发环路，当没有阻塞任何端口时，还会出现只有某些数据包通过阻塞流量的端口被错误转发的情况。在大多数情况下，这是由软件问题造成的。此类行为可能导致慢环路。这意味着一些数据包是环路的，但大部分流量仍然流经网络，因为链路没有拥塞。

排除转发环路故障

转发环路在其源（原因）和影响方面差别很大。由于可能影响STP的问题有多种，因此本文档只能提供有关如何排除转发环路故障的一般指南。

在开始进行故障排除之前，您需要以下信息：

- 详述所有交换机和网桥的实际拓扑图。
- 它们对应的端口号（互连）。
- STP配置详细信息，例如哪台交换机是根桥和备用根桥，哪些链路具有非默认开销或优先级，以及阻塞流量的端口的位置。

1. 确定环路

网络中形成转发环路时，通常会出现以下症状：

- 来自、通往或经过受影响网络区域的连接丢失。
- 连接到受影响网段或VLAN的路由器上的CPU使用率较高，这可能导致各种症状，例如路由协

议邻居抖动或热备用路由器协议(HSRP)活动路由器抖动。

- 链路利用率极高 (往往达到 100%)。
- 交换机背板利用率极高 (与基线利用率相比)。
- 系统日志消息，表示网络中的数据包循环 (例如HSRP重复IP地址消息)。
- 指示持续重新学习地址的 Syslog 消息或 MAC 地址抖动消息。
- 许多接口上的输出丢弃数量增加。

其中任何一种原因都可能表示不同的问题 (或根本不存在问题)。然而，如果同时观察到其中多种原因，则表明网络中很可能形成了转发环路。验证这一点的最快方式是检查交换机背板流量利用率：

```
<#root>
cat#
show catalyst6000 traffic-meter

traffic meter = 13%
Never cleared

peak = 14%
reached at 12:08:57 CET Fri Oct 4 2002
```

 注意：使用Cisco IOS软件的Catalyst 4000当前不支持此命令。

如果当前流量级别过大或基线级别未知，请检查最近是否达到了峰值级别，以及该级别是否接近当前流量级别。例如，如果峰值流量级别为15%，并且仅在两分钟前达到该值，而当前流量级别为14%，则意味着交换机具有异常高的负载。如果流量负载处于正常级别，则可能表明不存在环路或环路中未涉及此设备。然而，此设备仍可能包含在一个慢速环路中。

2. 发现环路的拓扑 (范围)

一旦确定网络中断的原因是转发环路，首先采取的操作就是停止该环路并恢复网络运行。

要停止环路，您必须知道哪些端口参与了环路：查看链路使用率最高的端口 (每秒数据包数)。show interface Cisco IOS软件命令可以显示每个接口的利用率。

要仅显示利用率信息和接口名称 (用于快速分析)，请使用Cisco IOS软件过滤正则表达式输出。发出show interface | include line|\vseccommand仅显示每秒数据包统计信息和接口名称：

```
<#root>
```

```
cat#
```


```
show interface | include line|\ /sec
```


```
GigabitEthernet2/1 is up, line protocol is down
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
GigabitEthernet2/2 is up, line protocol is down
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
GigabitEthernet2/3 is up, line protocol is up
  5 minute input rate 99765230 bits/sec, 24912 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
GigabitEthernet2/4 is up, line protocol is up
  5 minute input rate 1000 bits/sec, 27 packets/sec
  5 minute output rate 101002134 bits/sec, 25043 packets/sec
GigabitEthernet2/5 is administratively down, line protocol is down
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
GigabitEthernet2/6 is administratively down, line protocol is down
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
GigabitEthernet2/7 is up, line protocol is down
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
GigabitEthernet2/8 is up, line protocol is up
  5 minute input rate 2000 bits/sec, 41 packets/sec
  5 minute output rate 99552940 bits/sec, 24892 packets/sec
```


注意链路利用率最高的接口。在本例中，这些端口是接口g2/3、g2/4和g2/8；它们是参与环路的端口。


3. 中断循环

要断开环路，您必须先关闭或断开所涉及的端口。不仅要停止环路，还要查找并修复环路的根本原因，这一点尤其重要。中断环路相对容易

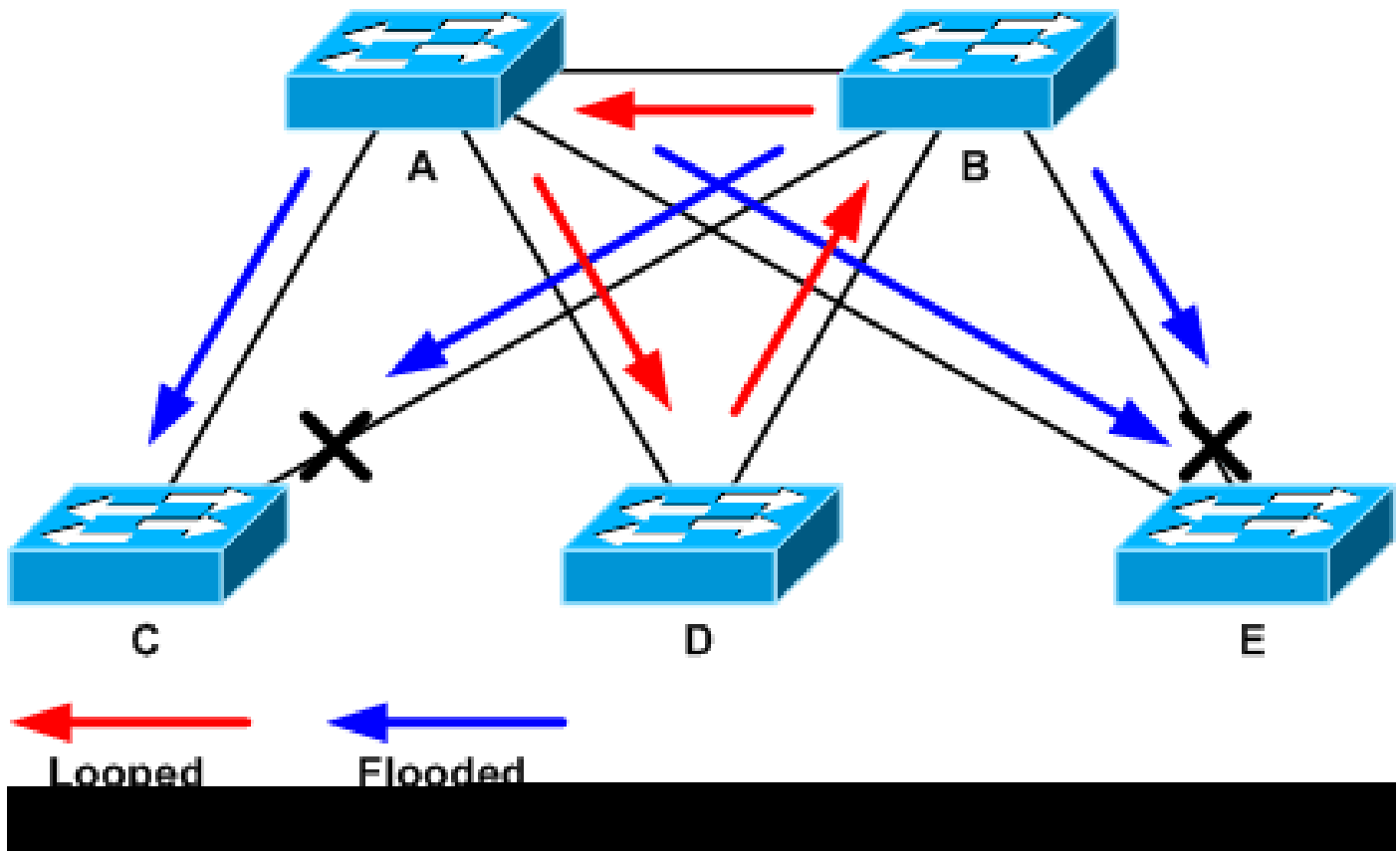
 **注意：**不必同时关闭或断开所有端口。您可以一次关闭一个设备。最好关闭受环路影响的聚合点（例如分布层交换机或核心层交换机）的端口。如果一次关闭所有端口并逐一启用或重新连接，则不会运行；环路会停止，并且在重新连接有故障的端口后无法立即启动。因此，很难将故障与任何特定端口相关联。

 **注意：**为了打破环路，建议在重新启动交换机之前收集信息。否则，后续根本原因分析将非常

 困难。禁用或断开每个端口后，必须检查交换机背板利用率是否恢复到正常级别。

 注意：请记住，端口不维持环路，而是泛洪随环路到达的流量。关闭此类泛洪端口时，只会略微降低背板利用率，而不会使环路停止。

在下面的示例拓扑中，环路位于交换机 A、B 和 D 之间。因此，链路 AB、AD 和 BD 是持续的。如果关闭其中任何链路，则会停止环路。链路 AC、AE、BC 和 BE 仅泛洪随环路到达的流量。



环路和泛洪流量

在关闭支持端口后，背板利用率降至正常值。您需要知道哪个端口的关闭使背板利用率（和其他端口的利用率）达到正常水平。此时，环路会停止，网络操作也会改善；但是，由于环路的原始原因尚未修复，因此还存在其他问题。

4. 查找并修复环路的原因

一旦环路停止，您需要确定环路开始的原因。这是此过程的难点，因为原因可能不同。制定在任何情况下都适用的确切步骤也很困难。

指南：

- 检查拓扑图以找到冗余路径。这包括上一步中找到的返回到同一交换机的支持端口（在环路期间交换的路径数据包）。在上一个示例拓扑中，此路径是 AD-DB-BA。
- 对于冗余路径上的每台交换机，检查该交换机是否知道正确的STP根。

L2网络中的所有交换机必须商定一个通用的STP根。如果网桥对特定 VLAN 或 STP 实例中的 STP 根一致显示不同 ID，则明确表明这是一个问题症状。请发出show spanning-tree vlan vlan-id 命令，以显示给定VLAN的根网桥ID：

```
<#root>
```

```
cat#
```

```
show spanning-tree vlan 333
```

```
MST03
```

```
Spanning tree enabled protocol mstp
```

```
Root ID      Priority      32771
             Address      0050.14bb.6000
             Cost          20000
             Port          136 (GigabitEthernet3/8)
             Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec

Bridge ID     Priority      32771 (priority 32768 sys-id-ext 3)
             Address      00d0.003f.8800
             Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec
```

```
Interface      Role Sts Cost      Prio.Nbr Status
-----
Gi3/8          Root FWD 20000     128.136 P2p
Po1            Desg FWD 20000     128.833 P2p
```

由于在前面步骤中确定了环路中涉及的端口，因此可以从端口中找到 VLAN 编号。如果相关端口是中继端口，则通常会涉及中继上的所有 VLAN。如果实际情况不是这样（例如，单个VLAN上似乎出现了环路），则可以尝试发出show interfaces | include L2|line|broadcastcommand（仅在Catalyst 6500/6000系列交换机上的Supervisor 2和更高版本引擎上，因为Supervisor 1不提供每个VLAN的交换统计信息）。请仅查看 VLAN 接口。交换数据包数量最多的VLAN通常是出现环路的VLAN：

```
<#root>
```

```
cat#
```

```
show interface | include L2|line|broadcast
```

```
Vlan1 is up, line protocol is up
```

```
  L2 Switched: ucast: 653704527 pkt, 124614363025 bytes - mcast:
    23036247 pkt, 1748707536 bytes
    Received 23201637 broadcasts, 0 runts, 0 giants, 0 throttles
```

```
Vlan10 is up, line protocol is up
```

```
  L2 Switched: ucast: 2510912 pkt, 137067402 bytes - mcast:
    41608705 pkt, 1931758378 bytes
    Received 1321246 broadcasts, 0 runts, 0 giants, 0 throttles
```

```
Vlan11 is up, line protocol is up
```

```
  L2 Switched: ucast: 73125 pkt, 2242976 bytes - mcast:
```

```

    3191097 pkt, 173652249 bytes
    Received 1440503 broadcasts, 0 runts, 0 giants, 0 throttles
Vlan100 is up, line protocol is up
    L2 Switched: ucast: 458110 pkt, 21858256 bytes - mcast:
        64534391 pkt, 2977052824 bytes
    Received 1176671 broadcasts, 0 runts, 0 giants, 0 throttles
Vlan101 is up, line protocol is up
    L2 Switched: ucast: 70649 pkt, 2124024 bytes - mcast:
        2175964 pkt, 108413700 bytes
    Received 1104890 broadcasts, 0 runts, 0 giants, 0 throttles

```

在本示例中，VLAN 1 占有的广播和 L2 交换流量的数量最多。确保根端口已正确识别。

根端口到根网桥的成本必须最低（有时一条路径的跳数更短，但成本更长，因为低速端口的成本更高）。要确定哪个端口被视为给定VLAN的根端口，请发出show spanning-tree vlan 命令：

```
<#root>
```

```
cat#
```

```
show spanning-tree vlan 333
```

```
MST03
```

```
Spanning tree enabled protocol mstp
Root ID    Priority    32771
           Address    0050.14bb.6000
           Cost      20000
```

```
Port      136 (GigabitEthernet3/8)
```

```
    Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec
```

```
Bridge ID Priority    32771 (priority 32768 sys-id-ext 3)
Address    00d0.003f.8800
Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec
```

Interface	Role	Sts	Cost	Prio.Nbr	Status
Gi3/8	Root	FWD	20000	128.136	P2p
Po1	Desg	FWD	20000	128.833	P2p

确保在根端口和应阻塞的端口上定期接收BPDU。

BPDU由根网桥以everyhellointerval（默认情况下为两秒）发送。非根网桥将接收、处理、修改和传播从根网桥接收的BPDU。请发出show spanning-tree interface interface detail命令，以查看是否收到了BPDU：

```
<#root>
```

```
cat#
```



```
show spanning-tree interface g3/2 detail
```

```
Port 130 (GigabitEthernet3/2) of MST00 is backup blocking
  Port path cost 20000, Port priority 128, Port Identifier 128.130.
  Designated root has priority 0, address 0007.4f1c.e847
  Designated bridge has priority 32768, address 00d0.003f.8800
  Designated port id is 128.129, designated path cost 2000019
  Timers: message age 4, forward delay 0, hold 0
```

```
Number of transitions to forwarding state: 0
```

```
  Link type is point-to-point by default, Internal
  Loop guard is enabled by default on the port
  BPDU: sent 3,
```

```
received 53
```

```
cat#
```

```
show spanning-tree interface g3/2 detail
```

```
Port 130 (GigabitEthernet3/2) of MST00 is backup blocking
  Port path cost 20000, Port priority 128, Port Identifier 128.130.
  Designated root has priority 0, address 0007.4f1c.e847
  Designated bridge has priority 32768, address 00d0.003f.8800
  Designated port id is 128.129, designated path cost 2000019
  Timers: message age 5, forward delay 0, hold 0
  Number of transitions to forwarding state: 0
  Link type is point-to-point by default, Internal
  Loop guard is enabled by default on the port
  BPDU: sent 3,
```

```
received 54
```



注意：在命令的两个输出之间收到一个BPDU（计数器从53变为54）。

显示的计数器实际上是 STP 进程本身维护的计数器。这意味着，如果接收计数器增加，不仅物理端口接收了BPDU，STP进程也接收了BPDU。如果received BPDU计数器在应该是根备用或备份端口的端口上没有增加，则检查该端口是否完全接收多播（BPDU作为多播发送）。发出show interface interface counterscommand：

```
<#root>
```

```
cat#
```

```
show interface g3/2 counters
```

Port	InOctets	InUcastPkts
Gi3/2	14873036	2

89387

0

Port	OutOctets	OutUcastPkts	OutMcastPkts	OutBcastPkts
Gi3/2	114365997	83776	732086	19

cat#

show interface g3/2 counters

Port	InOctets	InUcastPkts	InMcastPkts
------	----------	-------------	-------------

Port	InOctets	InUcastPkts	InBcastPkts
Gi3/2	14873677	2	

89391

0

Port	OutOctets	OutUcastPkts	OutMcastPkts	OutBcastPkts
Gi3/2	114366106	83776	732087	19

STP端口角色的概要可在[使用环路防护和BPDU迟滞检测功能的生成树协议增强功能的使用环路防护和BPDU迟滞检测增强STP](#)部分中找到。如果未收到任何BPDU，请检查端口是否对错误进行计数。请发出show interface interface counters errorscommand：

<#root>

cat#

show interface g4/3 counters errors

Port	Align-Err	FCS-Err	Xmit-Err	Rcv-Err	UnderSize	OutDiscards
Gi4/3	0	0	0	0	0	0

Port	Single-Col	Multi-Col	Late-Col	Excess-Col	Carri-Sen	Runts	Giants
Gi4/3	0	0	0	0	0	0	0

有可能物理端口收到了BPDU，但这些BPDU仍未到达STP进程。如果上两个示例中使用的命令显示收到一些多播，并且不计算错误，则检查是否在STP进程级别丢弃了BPDU。在Catalyst 6500上发出remote command switch test spanning-tree process-statscommand：

<#root>

cat#

remote command switch test spanning-tree process-stats

```
-----TX STATS-----  
transmission rate/sec      = 2
```

```

paks transmitted          = 5011226
paks transmitted (opt)    = 0
opt chunk alloc failures = 0
max opt chunk allocated   = 0
-----RX STATS-----

receive rate/sec          = 1

paks received at stp isr  = 3947627
paks queued at stp isr    = 3947627

paks dropped at stp isr   = 0
drop rate/sec             = 0

paks dequeued at stp proc = 3947627
paks waiting in queue     = 0
queue depth               = 7(max) 12288(total)
-----PROCESSING STATS-----
queue wait time (in ms)   = 0(avg) 540(max)
processing time (in ms)   = 0(avg) 4(max)
proc switch count         = 100
add vlan ports            = 20
time since last clearing   = 2087269 sec

```

本示例中使用的命令显示STP进程的统计信息。检验丢弃计数器是否不会增加以及收到的数据包是否增加非常重要。如果收到的数据包没有增加，但物理端口确实接收了多播，请验证交换机带内接口（CPU接口）是否接收了数据包。发出remote command switch show ibc | i rx_inputcommand（在Catalyst 6500/6000上）：

```

<#root>

cat#
remote command switch show ibc | i rx_input


rx_inputs=
5626468
, rx_cumbytes=859971138

cat#
remote command switch show ibc | i rx_input

rx_inputs=
5626471
, rx_cumbytes=859971539

```

本示例显示带内端口在两次输出之间收到了 23 个数据包。

 注意：这23个数据包不仅是BPDU数据包；它是带内端口收到的所有数据包的全局计数器。

如果没有指示BPDU在本地交换机或端口上被丢弃，您必须移到链路另一端的交换机并验证该交换机是否发送了BPDU。检查BPDU是否定期在非根指定端口上发送。如果端口角色同意，端口将发送BPDU，但邻居不会收到它们。检查是否发送了BPDU。请发出show spanning-tree interface interface detail命令：

```
<#root>
```

```
cat#
```

```
show spanning-tree interface g3/1 detail
```

```
Port 129 (GigabitEthernet3/1) of MST00 is
```

```
designated
```

```
forwarding
```

```
Port path cost 20000, Port priority 128, Port Identifier 128.129.  
Designated root has priority 0, address 0007.4f1c.e847  
Designated bridge has priority 32768, address 00d0.003f.8800  
Designated port id is 128.129, designated path cost 2000019  
Timers: message age 0, forward delay 0, hold 0  
Number of transitions to forwarding state: 0  
Link type is point-to-point by default, Internal  
Loop guard is enabled by default on the port
```

```
BPDU: sent 1774
```

```
, received 1
```

```
cat#
```

```
show spanning-tree interface g3/1 detail
```

```
Port 129 (GigabitEthernet3/1) of MST00 is
```

```
designated
```


```
forwarding
```

```
Port path cost 20000, Port priority 128, Port Identifier 128.129.  
Designated root has priority 0, address 0007.4f1c.e847  
Designated bridge has priority 32768, address 00d0.003f.8800  
Designated port id is 128.129, designated path cost 2000019  
Timers: message age 0, forward delay 0, hold 0  
Number of transitions to forwarding state: 0  
Link type is point-to-point by default, Internal  
Loop guard is enabled by default on the port
```

```
BPDU: sent 1776
```

```
, received 1
```

在本示例中，在输出之间发送两个BPDU。

 注意：STP进程维护BPDU：sentcounter。这意味着计数器表示BPDU已发送到物理端口并发送出去。检查传输的多播数据包的端口计数器是否增加。发出show interface interface counterscommand。这有助于确定BPDU流量。

<#root>

cat#

```
show interface g3/1 counters
```

Port	InOctets	InUcastPkts	InMcastPkts	InBcastPkts
Gi3/1	127985312	83776	812319	19

Port	OutOctets	OutUcastPkts
------	-----------	--------------

OutMcastPkts

	OutBcastPkts	
Gi3/1	131825915	3442

872342

386

cat#

```
show interface g3/1 counters
```

Port	InOctets	InUcastPkts	InMcastPkts	InBcastPkts
Gi3/1	127985312	83776	812319	19

Port	OutOctets	OutUcastPkts
------	-----------	--------------

OutMcastPkts

	OutBcastPkts	
Gi3/1	131826447	3442

872346

386

对于所有这些步骤，其思路都是查找没有接收、发送或处理 BPDU 的交换机或链路。STP可能计算出了端口的正确状态，但由于控制平面问题，无法在转发硬件上设置此状态。如果端口在硬件级别未被阻塞，则可能会形成环路。如果您认为这是您网络中的问题，[请与Cisco技术支持联系](#)以获取进一步帮助。

5. 恢复冗余

找到导致环路的设备或链路后，必须将该设备与网络隔离，否则必须解决问题。（例如更换光纤或GBIC）。必须恢复在步骤3中断开的冗余链路。


重要的是不要操作导致环路的设备或链路，因为导致环路的许多情况都是瞬时、间歇性和不稳定的。这意味着，如果在调查中或调查后清除该情况，则该情况暂时不会出现，或者根本不会出现。必

须记录该情况，以便[Cisco技术支持](#)可以进一步调查。务必在重置交换机之前收集有关该情况的信息。如果情况消失，则无法确定形成环路的根本原因。如果收集信息，请确保此问题不会再次导致环路。有关详细信息，请参阅[防止网络发生转发环路](#)。

调查拓扑更改

拓扑更改(TC)机制的作用是在拓扑更改后更正L2转发表。这是避免连接中断所必需的，因为之前可以通过特定端口访问的MAC地址可能会更改并通过不同端口访问。TC缩短了发生TC的VLAN中所有交换机的转发表时间。因此，如果不重新学习地址，地址会老化，并且发生泛洪以确保数据包到达目的MAC地址。

端口的STP状态与STP forwarding state之间的转换会触发TC。在TC之后，即使特定目标MAC地址已老化，泛洪也不会持续很长时间。来自主机（其MAC地址已过期）的第一个数据包将重新获取该地址。当TC在较短的间隔内重复发生时，可能会出现此问题。交换机不断快速老化其转发表，因此泛洪几乎可以保持恒定。

 **注意：**使用快速STP或多STP（IEEE 802.1w和IEEE 802.1s），TC由端口状态从forwarding变化以及角色从designated to root变化触发。对于快速STP，L2转发表会立即刷新（与802.1d相对），从而缩短老化时间。立即刷新转发表可更快地恢复连接，但可能会导致更多泛洪

在配置良好的网络中，TC是一种罕见的事件。当交换机端口上的某条链路接通或断开时，一旦该端口的STP状态在forwarding间切换，便会最终发生TC。当端口抖动时，将导致重复的TC和泛洪。

启用了STP portfast功能的端口在转为/转出forwarding状态时，无法导致TC。在所有终端设备端口（如打印机、PC和服务器）上配置portfast可以将TC限制在低数量范围内，因此强烈建议进行配置。

如果网络上有重复TC，则您必须标识这些TC的来源并采取相应操作来减少它们，从而将泛洪降到最低。

对于802.1d，有关TC事件的STP信息在网桥中通过TC通知(TCN)传播，TCN是一种特殊类型的BPDU。如果跟踪接收TCN BPDU的端口，则可以找到产生TC的设备。

查找泛洪的原因

您可以确定存在因性能低下而导致的泛洪、不应该拥塞的链路上的数据包丢弃，以及数据包分析器显示发送到不在本地网段上的同一目的地的多个单播数据包。有关单播泛洪的详细信息，请参阅[交换式园区网络中的单播泛洪](#)。

在运行Cisco IOS软件的Catalyst 6500/6000上，您可以检查转发引擎计数器（仅针对Supervisor 2引擎）以估计泛洪数量。请发出remote command switch show earl statistics | i MISS_DA|ST_FR命令：

```
<#root>
```

```
cat#
```

```
remote command switch show earl statistics | i MISS_DA|ST_FR

      ST_MISS_DA      =      18      530308834
      ST_FRMS         =      97      969084354
```

cat#

```
remote command switch show earl statistics | i MISS_DA|ST_FR

      ST_MISS_DA      =      4      530308838
      ST_FRMS         =     23      969084377
```

在本示例中，第一列显示自上次执行此命令以来的更改，第二列显示自上次重新启动以来的累计值。第一行显示泛洪帧的数量，第二行显示已处理的帧的数量。如果这两个值相近，或者第一个值以较高的速率增加，则可能交换机正在泛洪流量。不过，由于计数器并不精细，因此这只能与验证泛洪的其他方式一起使用。每台交换机（非每个端口或 VLAN）有一个计数器。看到一些泛洪数据包是很正常的，因为如果目标MAC地址不在转发表中，交换机总是可以泛洪。当交换机接收到目的地址尚未获知的数据包时，可能出现这种情况。

查找TC

如果已知发生过度泛洪的VLAN的VLAN编号，请检查STP计数器以查看TC的数量是高还是有规律地增加。发出show spanning-tree vlan vlan-id detail command（在本示例中，使用VLAN 1）：

```
<#root>
```

cat#

```
show spanning-tree vlan 1 detail
```

```
VLAN0001 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, sysid 1, address 0007.0e8f.04c0
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 0, address 0007.4f1c.e847
  Root port is 65 (GigabitEthernet2/1), cost of root path is 119
  Topology change flag not set, detected flag not set
```

```
Number of topology changes 1 last change occurred 00:00:35 ago
  from GigabitEthernet1/1
```


```
Times: hold 1, topology change 35, notification 2
       hello 2, max age 20, forward delay 15
Timers: hello 0, topology change 0, notification 0, aging 300
```

如果不知道 VLAN 编号，您可以使用数据包分析程序或检查所有 VLAN 的 TC 计数器。

采取步骤来防止 TC 过多

您可以监控拓扑更改计数器的数量以查看它是否定期增加。然后，移到连接到所示端口的网桥，接收

最后一个 TC (在前一个示例中为端口 GigabitEthernet1/1) 并查看 TC 从该网桥的何处发出。必须重复此过程，直到找到未启用 STP portfast 的终端站端口，或者直到发现需要修复的抖动链路。如果 TC 来自其他来源，则需要重复整个过程。如果链路属于终端主机，则可以配置 portfast 功能以防止生成 TC。

 注意：在 Cisco IOS 软件 STP 实施中，TC 的计数器只能在 VLAN 中的端口收到 TCN BPDU 时递增。如果接收到具有设置 TC 标志的正常配置 BPDU，则 TC 计数器不会递增。这意味着，如果您怀疑 TC 是泛洪的原因，则开始从该 VLAN 中的 STP 根网桥跟踪 TC 的源。它可以获得有关 TC 的数量和来源的最准确信息。

排除收敛时间相关问题

有时会出现 STP 的实际操作与预期行为不匹配的情况。以下是两个最常见的问题：

- STP 收敛或再收敛比预期所需时间长。
- 拓扑结果与预期结果不同。

通常，此行为的原因如下：


- 实际拓扑和记录的拓扑之间不匹配。
- 配置错误，例如 STP 计时器配置不一致、STP 直径增加或 portfast 配置错误。
- 收敛或再收敛期间交换机 CPU 过载。
- 软件缺陷。

如前文所述，由于可能影响 STP 的问题有多种，因此本文档只能提供用于故障排除的一般指南。

要了解收敛时间比预期长的原因，请查看 STP 事件的顺序，以了解发生的情况和顺序。由于 Cisco IOS 软件中的 STP 实施不会记录结果（端口不一致等特定事件除外），因此您可以使用 Cisco IOS 软件调试 STP 以便更清楚地查看。对于 STP，使用运行 Cisco IOS 软件的 Catalyst 6500/6000，处理在交换机处理器 (SP) (或 Supervisor) 上完成，因此需要在 SP 上启用调试。对于 Cisco IOS 软件网桥组，处理在路由处理器 (RP) 上完成，因此需要在 RP (MSFC) 上启用调试。

使用 STP 调试命令

许多 STP debug commands 适用于开发工程。对于没有详细了解 Cisco IOS 软件中的 STP 实施的某些用户，这些命令不会提供任何有意义的输出。部分调试命令可提供立即可读的输出，例如端口状态更改、角色更改、事件（例如 TC）以及已接收和传输的 BPDU 的转储。本部分没有完整介绍所有调试命令，而是简单介绍了最常使用的调试命令。

 注意：使用 debug commands 时，请启用最低限度的必要调试操作。如果不需要实时调试，请将输出记录到日志中而不是将其打印到控制台。过多的调试可能会使 CPU 过载并中断交换机操作。

要将调试输出定向到日志而非控制台或Telnet会话，请在全局配置模式下发出logging console informational and no logging monitor commands。要查看一般事件日志，请对每VLAN生成树(PVST)和快速PVST发出debug spanning-tree event command。这是提供有关STP所发生情况的第一个调试。在多生成树(MST)模式下，发出debug spanning-tree event command不起作用。因此，请发出debug spanning-tree mstp roles command以查看端口角色更改。要查看端口STP状态更改，请将debug spanning-tree switch state command与debug pm vp command一起发出：

```
<#root>
```

```
cat-sp#
```

```
debug spanning-tree switch state
```

```
Spanning Tree Port state changes debugging is on
```

```
cat-sp#
```

```
debug pm vp
```

```
Virtual port events debugging is on
```

```
Nov 19 14:03:37: SP: pm_vp 3/1(333): during state forwarding, got event 4(remove)
```

```
Nov 19 14:03:37: SP:
```

```
@@@
```

```
pm_vp 3/1(333):
```

```
forwarding -> notforwarding
```

```
port 3/1 (was forwarding) goes down in vlan 333
```

```
Nov 19 14:03:37: SP: *** vp_fwdchange: single: notfwd: 3/1(333)
```

```
Nov 19 14:03:37: SP: @@@ pm_vp 3/1(333): notforwarding -> present
```

```
Nov 19 14:03:37: SP: *** vp_linkchange: single: down: 3/1(333)
```

```
Nov 19 14:03:37: SP: @@@ pm_vp 3/1(333): present -> not_present
```

```
Nov 19 14:03:37: SP: *** vp_statechange: single: remove: 3/1(333)
```

```
Nov 19 14:03:37: SP: pm_vp 3/2(333): during state notforwarding,  
got event 4(remove)
```

```
Nov 19 14:03:37: SP:
```

```
@@@
```

```
pm_vp 3/2(333): notforwarding -> present
```

```
Nov 19 14:03:37: SP: *** vp_linkchange: single: down: 3/2(333)
```

```
Port 3/2 (was not forwarding) in vlan 333 goes down
```

```
Nov 19 14:03:37: SP: @@@ pm_vp 3/2(333): present -> not_present
```

```
Nov 19 14:03:37: SP: *** vp_statechange: single: remove: 3/2(333)
```

```
Nov 19 14:03:53: SP: pm_vp 3/1(333): during state not_present,  
got event 0(add)
```

```
Nov 19 14:03:53: SP: @@@ pm_vp 3/1(333): not_present -> present
```

```
Nov 19 14:03:53: SP: *** vp_statechange: single: added: 3/1(333)
```

```
Nov 19 14:03:53: SP: pm_vp 3/1(333): during state present,
```

```

got event 8(linkup)
Nov 19 14:03:53: SP:

@@@

pm_vp 3/1(333): present ->
notforwarding
Nov 19 14:03:53: SP: STP SW: Gi3/1 new blocking req for 0 vlans
Nov 19 14:03:53: SP: *** vp_linkchange: single: up: 3/1(333)

Port 3/1 link goes up and blocking in vlan 333

Nov 19 14:03:53: SP: pm_vp 3/2(333): during state not_present,
got event 0(add)
Nov 19 14:03:53: SP: @@@ pm_vp 3/2(333): not_present -> present
Nov 19 14:03:53: SP: *** vp_statechange: single: added: 3/2(333)

Nov 19 14:03:53: SP: pm_vp 3/2(333): during state present,
got event 8(linkup)
Nov 19 14:03:53: SP:

@@@

pm_vp 3/2(333): present ->
notforwarding
Nov 19 14:03:53: SP: STP SW: Gi3/2 new blocking req for 0 vlans
Nov 19 14:03:53: SP: *** vp_linkchange: single: up: 3/2(333)

Port 3/2 goes up and blocking in vlan 333

Nov 19 14:04:08: SP: STP SW: Gi3/1 new learning req for 1 vlans
Nov 19 14:04:23: SP: STP SW: Gi3/1 new forwarding req for 0 vlans
Nov 19 14:04:23: SP: STP SW: Gi3/1 new forwarding req for 1 vlans
Nov 19 14:04:23: SP: pm_vp 3/1(333): during state notforwarding,
got event 14(forward_notnotify)
Nov 19 14:04:23: SP:

@@@ pm_vp 3/1(333): notforwarding ->
forwarding
Nov 19 14:04:23: SP: *** vp_list_fwdchange: forward: 3/1(333)

Port 3/1 goes via learning to forwarding in vlan 333

```

要了解 STP 以某种方式运行的原因，查看通过交换机接收和发送的 BPDU 通常很有用：

```
<#root>
```

```
cat-sp#
```

```
debug spanning-tree bpdu receive
```

```
Spanning Tree BPDU Received debugging is on
```

```
Nov 6 11:44:27: SP: STP: VLAN1 rx BPDU: config protocol = ieee,
packet from GigabitEthernet2/1 , linktype IEEE_SPANNING ,
enctype 2, encsize 17
```

```
Nov 6 11:44:27: SP: STP: enc 01 80 C2 00 00 00 00 06 52 5F 0E 50 00 26 42 42 03
```

```
Nov 6 11:44:27: SP: STP: Data 000000000000000074F1CE8470000001380480006525F0E4
```

```
080100100140002000F00
Nov 6 11:44:27: SP: STP: VLAN1 Gi2/1:0000 00 00 00 000000074F1CE847 00000013
80480006525F0E40 8010 0100 1400 0200 0F00
```

此调试适用于PVST、快速PVST和MST模式；但它不会解码BPDU的内容。然而，您可以使用它来确保收到BPDU。要查看BPDU的内容，请对PVST和快速PVST将debug spanning-tree switch rx decodecommand与debug spanning-tree switch rx processcommand一起发出。发出debug spanning-tree mstp bpdu-rxcommand可以查看MST的BPDU的内容：

```
<#root>
```

```
cat-sp#
```

```
debug spanning-tree switch rx decode
```

```
Spanning Tree Switch Shim decode received packets debugging is on
```

```
cat-sp#
```

```
debug spanning-tree switch rx process
```

```
Spanning Tree Switch Shim process receive bpdu debugging is on
```

```
Nov 6 12:23:20: SP: STP SW: PROC RX: 0180.c200.0000<-0006.525f.0e50 type/len 0026
Nov 6 12:23:20: SP:      encap SAP linktype ieee-st vlan 1 len 52 on v1 Gi2/1
Nov 6 12:23:20: SP:      42 42 03 SPAN
Nov 6 12:23:20: SP:      CFG P:0000 V:00 T:00 F:00 R:0000 0007.4f1c.e847 00000013
Nov 6 12:23:20: SP:      B:8048 0006.525f.0e40 80.10 A:0100 M:1400 H:0200 F:0F00

Nov 6 12:23:22: SP: STP SW: PROC RX: 0180.c200.0000<-0006.525f.0e50 type/len 0026
Nov 6 12:23:22: SP:      encap SAP linktype ieee-st vlan 1 len 52 on v1 Gi2/1
Nov 6 12:23:22: SP:      42 42 03 SPAN
Nov 6 12:23:22: SP:      CFG P:0000 V:00 T:00 F:00 R:0000 0007.4f1c.e847 00000013
Nov 6 12:23:22: SP:      B:8048 0006.525f.0e40 80.10 A:0100 M:1400 H:0200 F:0F00
```

对于MST模式，您可以使用thisdebugcommand：

```
<#root>
```

```
cat-sp#
```

```
debug spanning-tree mstp bpdu-rx
```

```
Multiple Spanning Tree Received BPDUs debugging is on
```


```
Nov 19 14:37:43: SP: MST:BPDU DUMP [
```

```
rcvd_bpdu Gi3/2
```

```
Repeated]
```

```
Nov 19 14:37:43: SP: MST:   Proto:0 Version:3 Type:2 Role: DesgFlags[   F   ]
Nov 19 14:37:43: SP: MST:   Port_id:32897 cost:2000019
Nov 19 14:37:43: SP: MST:   root_id   :0007.4f1c.e847 Prio:0
Nov 19 14:37:43: SP: MST:   br_id    :00d0.003f.8800 Prio:32768
```

```
Nov 19 14:37:43: SP: MST: age:2 max_age:20 hello:2 fwdelay:15
Nov 19 14:37:43: SP: MST: V3_len:90 PathCost:30000 region:STATIC rev:1
Nov 19 14:37:43: SP: MST: ist_m_id :0005.74
Nov 19 14:37:43: SP: MST:BPDU DUMP [
rcvd_bpdu Gi3/2
Repeated]
Nov 19 14:37:43: SP: MST: Proto:0 Version:3 Type:2 Role: DesgFlags[ F ]
Nov 19 14:37:43: SP: MST: Port_id:32897 cost:2000019
Nov 19 14:37:43: SP: MST: root_id :0007.4f1c.e847 Prio:0
Nov 19 14:37:43: SP: MST: br_id :00d0.003f.8800 Prio:32768
Nov 19 14:37:43: SP: MST: age:2 max_age:20 hello:2 fwdelay:15
Nov 19 14:37:43: SP: MST: V3_len:90 PathCost:30000 region:STATIC rev:1
Nov 19 14:37:43: SP: MST: ist_m_id :0005.7428.1440 Prio:32768 Hops:18
Num Mrec: 1
Nov 19 14:37:43: SP: MST: stci=3 Flags[ F ] Hop:19 Role:Desg [Repeated]
Nov 19 14:37:43: SP: MST: br_id:00d0.003f.8800 Prio:32771 Port_id:32897
Cost:2000028.1440 Prio:32768 Hops:18 Num Mrec: 1
Nov 19 14:37:43: SP: MST: stci=3 Flags[ F ] Hop:19 Role:Desg [Repeated]
Nov 19 14:37:43: SP: MST: br_id:00d0.003f.8800 Prio:32771 Port_id:32897
Cost:20000
```

 **注意：**对于Cisco IOS软件版本12.1.13E及更高版本，支持STP的条件调试。这意味着，您可以基于每个端口或每个VLAN调试所接收或传输的BPDU。

发出debug condition vlan vlan_num 或debug condition interface interface 命令，可将调试输出的范围限制为每个接口或每个VLAN。

保护网络免受转发环路影响

Cisco已经开发了许多功能和增强功能，用于在STP无法管理某些故障时保护网络免受转发环路的影响。

当您排除STP故障时，它有助于隔离特定故障并可能找到其原因，而实施这些增强功能是保护网络免受转发环路影响的唯一方法。

以下方法可防止网络出现转发环路：


1. 在所有交换机到交换机链路上启用单向链路检测(UDLD)

有关UDLD的详细信息，请参阅[了解和配置单向链路检测协议功能](#)。


2. 在所有交换机上启用环路防护

有关环路防护的详细信息，请参阅[使用环路防护和BPDU迟滞检测功能的生成树协议增强功能](#)。

启用后，UDLD和环路防护可消除导致转发环路的大部分原因。有缺陷的链路（或依赖于有缺陷的硬件的所有链路）关闭或被阻塞，而不是造成转发环路。


 注意：虽然这两个功能看似多余，但每个功能都有其独特的功能。因此，同时使用这两个功能可提供最高级别的保护。有关UDLD和环路防护的详细比较，请参阅[环路防护与单向链路检测](#)。

关于是否必须使用主动 UDLD 还是正常 UDLD 有不同的观点。与普通模式UDLD相比，主动UDLD无法提供更多环路保护。主动 UDLD 可检测端口卡住情形（如果链路接通，但没有任何关联的流量黑洞）。增加的这一功能的不利方面是，主动 UDLD 可能会在不存在一致故障时禁用链路。通常，用户会混淆UDLDhellointerval的修改与主动UDLD功能。这是不正确的。可以在这两种 UDLD 模式下修改计时器。

 注意：在极少数情况下，主动UDLD可以关闭所有上行链路端口，这基本上会将该交换机与网络的其余部分隔离。例如，当两台上游交换机的CPU使用率极高且使用主动模式UDLD时，可能会发生这种情况。因此，建议您配置在交换机没有带外管理的情况下不能破坏的超时。

3. 在所有终端站端口上启用Portfast

您必须启用 portfast 来限制 TC 和随后的泛洪的数量，这可能会影响网络的性能。请将此命令仅用于连接到终端站的端口。否则，意外出现的拓扑环路可能会导致数据包环路并中断交换机和网络运行。

 注意：在使用no spanning-tree portfast命令时务必谨慎。此命令只会删除任何端口特定的 portfast 命令。如果您在全局配置模式下定义 spanning-tree portfast default 命令，并且端口不是中继端口，此命令会隐式启用 portfast。如果不全局配置 portfast，no spanning-tree portfast 命令与 spanning-tree portfast disable 命令等效。

4. 在两端（受支持）和Non-silent 选项上将EtherChannel设置为Desirable 模式

Desirablemode可以启用端口聚合协议(PAgP)，以确保信道对等体之间的运行时一致性。这样可针对环路提供额外的防护，尤其是在信道重新配置期间（例如，链路加入或离开信道以及链路故障检测）更是如此。有一个内置的信道配置错误防护，默认情况下启用，可防止由于信道配置错误或其他情况而导致的转发环路。有关此功能的详细信息，请参阅[了解EtherChannel不一致检测](#)。

5. 请勿禁用交换机到交换机链路上的自动协商（如果支持）

自动协商机制可以传达远程故障信息，这是在远程端检测故障的最快方法。如果在远程端检测到故障，则即使链路收到脉冲，本地端也会关闭链路。与UDLD等高级检测机制相比，自动协商速度极快（在微秒内），但缺少UDLD的端到端覆盖（例如整个数据路径：CPU -转发逻辑-端口1 -端口2 -转发逻辑-CPU与端口1 -端口2）。在故障检测方面，主动 UDLD 模式可提供与自动协商类似的功能。当协商在链路的两端受支持时，不需要启用主动模式 UDLD。

6. 调整STP计时器时务必谨慎

STP 计时器彼此之间以及在网络拓扑上相互依赖。STP无法正确处理对计时器进行的任意修改。有关STP计时器的详细信息，请参阅[了解和调整生成树协议计时器](#)。

7. 如果可能发生拒绝服务攻击，请使用根防护来保护网络STP周界

通过根防护和 BPDU 防护，可以防止 STP 受到外界影响。如果可能存在此类攻击，则必须使用根防护和 BPDU 防护来保护网络。有关根防护和 BPDU 防护的详细信息，请参阅以下文档：

- [生成树协议根防护增强](#)
- [生成树 PortFast BPDU 防护增强功能](#)

8. 在启用Portfast的端口上启用BPDU防护，以防止STP受到连接到端口的未授权网络设备（如集线器、交换机和桥接路由器）的影响

如果根防护配置正确，它将防止STP受到外界的影响。如果启用了BPDU防护，它会关闭接收任何BPDU的端口。这对调查事件很有用，因为BPDU防护会生成系统日志消息并关闭端口。如果根或BPDU防护不能防止短周期环路，则两个支持快速功能的端口直接连接或通过集线器连接。

9. 避免管理VLAN上的用户流量

管理 VLAN 包含在构造块（而非整个网络）中。

交换机管理接口在管理 VLAN 上接收广播数据包。如果出现过多的广播（例如广播风暴或应用程序故障），交换机CPU可能会过载，从而可能使STP运行失真。

10. 可预测（硬编码）的STP根和备份STP根位置

必须配置 STP 根和备份 STP 根，以便在发生故障时以可预测的方式进行收敛，并在每种情形下构建最佳拓扑。请不要将 STP 优先级置于默认值，以防止执行无法预测的根交换机选择。

相关信息

- [LAN 产品支持](#)
- [LAN 交换技术支持](#)

关于此翻译

思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。