了解Catalyst 9800 WLC上的CAPWAP Keepalive并对其进行故障排除

目录

简介

本文档介绍CAPWAP控制keepalive、data keepalive和retransmissions，

情景

CAPWAP提供思科接入点和Cisco Catalyst 9800平台上的无线局域网控制器之间的通信框架。它使用单独的控制通道和数据通道，以及定义的keepalive和retransmission机制来保持连接。

本文介绍CAPWAP控制keepalive、data keepalive和retransmissions，包括关联的计时器、故障行为和关键故障排除指示符。

两个独立的通道，两个不同的机制

CAPWAP在两个UDP信道上运行，每个信道都有自己的活动机制

通道	默认端口(UDP)	活性机制
Control	5246	回应请求/回应响应
数据	5247	数据信道保活

CAPWAP控制Keepalive（回声/心跳）

目的

CAPWAP控制keepalive用于确认接入点在控制信道上仍然可访问。

其目的非常简单：它验证AP和控制器之间的控制连接仍然处于活动状态且响应迅速。它不用于传送配置更新或客户端数据。相反，它充当通过UDP端口5246的CAPWAP控制路径的活跃机制。

谁发送Keepalive

接入点是主动启动控制keepalive的设备。

必要时，AP会向控制器发送CAPWAP Echo Request。然后，控制器使用相应的Echo Response进行响应。此响应确认控制器已收到请求，并且控制信道在两个方向上都正常工作。

响应与请求匹配，使AP能够确认它正在接收对发送的keepalive的有效应答。

此行为很重要，因为它表明控制keepalive主要不是控制器驱动的轮询机制。相反，AP负责检查控制信道的可达性，控制器做出相应响应。

回声间隔基于非活动状态，而不是固定的时间表

这是CAPWAP控制keepalive最常被误解的方面之一。

一个常见的假设是AP每30秒发送一次回应请求，就像定期发送固定心跳一样。但实际上，这并非它的运作方式。

控制keepalive基于控制信道非活动状态。这意味着AP仅在控制信道在配置的时间间隔内空闲时才发送独立回应请求。如果AP和控制器之间已在交换其他控制流量，则无需发送单独的回应数据包。

AP与控制器之间的任何有效控制通信都有效证明了控制信道是活动的。因此，正常CAPWAP控制

流量会重置keepalive计时器。

此类控制流量的示例包括：

- 配置交换
- 无线电资源管理更新
- 与WLAN相关的控制消息
- 客户端相关的控制事件
- 统计信息或状态同步消息
- 正常操作过程中交换的其他CAPWAP控制数据包

因此，仅当控制信道处于静默状态时，才会看到独立的回应请求。

简单示例

您可以使用show capwap client timer命令检查AP上的CAPWAP回应计时器。

<#root>

```
Training-AP#

show capwap client timer


CAPWAP TIMERS Running Current / Max (seconds)
PATHMTU_TIMER : 23 / 30
MSG_CLIENT_STAT : 124 / 180
DATA_CHANNEL_KEEP_ALIVE_TIMER : 24 / 30

ECHO_INTERVAL_TIMER : 23 / 30



PERIODIC_ECHO_TIMER : 233 / 300


PRIMARY_DISCOVERY_TIMER : 50 / 120
CAPWAP_WDG_UPDATE_TIMER : 4 / 5
FLASH_WRITE_INTERVAL_TIMER : 21 / 60
```

计时器

控制保持连接机制依赖于两个重要的定时值。

项目	默认值	描述
回声间隔	30 秒	AP发送独立回应请求之前控制信道处于非活动状态的数量
控制器控制dead计时器	90 秒	在声明控制会话死机之前，控制器在未接收有效控制流量的情况下允许的最长时间

当控制Keepalive失败时会发生什么情况

如果控制信道保持静默的时间超过允许的超时，控制器最终会声明控制会话丢失。

此时，控制器会将AP视为在控制信道上不再可访问，并开始会话断开。这通常包括关闭控制连接以及删除AP的活动控制会话状态。

然后，AP可以根据故障场景通过重新发现和重新加入控制器尝试重新建立连接。

故障排除含义

在故障排除过程中，了解此keepalive行为至关重要。

缺少回应数据包并不总是意味着有问题

如果数据包捕获未每30秒显示一次Echo Requests，则不自动指示故障。它可能只是意味着有足够多的其他CAPWAP控制流量在流动，因此不需要独立回声。

不规则的回声间隔通常是正常的

回波通常以非统一间隔出现，因为它们由非活动性触发。这是预料之中的现象。

关注整体控制渠道活动

对AP断开进行故障排除时，最好询问以下问题：

- 控制器是否仍然接收来自AP的任何CAPWAP控制流量？
- 控制信道是否保持静默足够长以触发Dead计时器？
- 是否存在仅影响控制路径的网络问题？
- 数据包丢失、防火墙行为、NAT问题或控制平面丢弃是否会中断UDP 5246流量？

真正的问题不是没有定期回声数据包。真正的问题是控制信道通信丢失的时间足够长，控制器会宣布AP不可达。

CAPWAP数据保持连接

目的

CAPWAP控制回显确认控制信道正常工作，但无法证明数据信道也是正常的。由于控制路径和数据路径可能会独立失败，CAPWAP在UDP端口5247上使用单独的数据保持连接。

数据keepalive的作用在于：

- 验证AP仍然可以到达数据信道上的控制器
- 维护AP的数据隧道状态
- 帮助检测仅影响数据路径的故障

发送者以及控制器如何响应

AP发送数据keepalive，控制器以数据Keepalive Response响应。

响应可以加密或不加密，具体取决于是否启用数据通道加密。

如果keepalive有效，则控制器使用它来确认AP的数据路径仍然可达。如果数据包无法与有效的AP会话匹配，或者无法发送响应，则将keepalive丢弃，数据路径最终将被视为失败。

控制器如何对其进行验证

在接受keepalive数据包之前，控制器会执行基本验证，以确保数据包属于正确的AP。

控制器会检查：

- keepalive包含有效的CAPWAP信息
- ap无线电MAC地址存在
- 数据包可以与AP会话匹配

控制器首先尝试使用源IP地址和源UDP端口识别会话。如果失败，它会回退到使用AP无线MAC地址。

这对于NAT或PAT后面的AP非常重要，其中数据信道可以从不同于控制信道的转换UDP端口到达。在这种情况下，控制器可以学习和更新AP的实际数据信道元组。

如果数据包似乎属于与该IP端口组合关联的会话不同的AP，则拒绝保持连接，以防止不正确的会话映射。

验证后发生的情况

一旦接受keepalive，控制器会根据AP的当前会话状态对其进行处理。

- 如果已建立数据会话，则keepalive只会刷新活动性。
- 如果这是第一个有效数据keepalive，则控制器可以使用它了解或更新AP的数据通道信息并完成数据隧道建立。

第一次保持连接特别重要，因为它有助于控制器正确编程数据隧道，包括AP位于NAT或PAT后面的情况。

在数据会话完全建立后，将来的keepalive数据包遵循正常的稳态路径，只用于保持活动状态。

重要行为

有效数据keepalive的作用不只限于确认数据路径的运行状况。它还会刷新控制器上的AP的整体会话活动性。

这意味着，在控制器上，数据通道活动性会影响AP会话的整体运行状况。因此，控制和数据活动机制是相关的，即使它们用于不同的目的。

AP端数据保持连接计时器

AP控制数据保持连接间隔，并决定何时必须考虑关闭数据隧道。

项目	默认值
数据保持连接间隔	30 秒
重新传输回退	3秒、6秒、12秒，然后是15秒
数据停顿间隔	180 秒

在正常情况下，AP每30秒发送一次数据keepalive。

如果AP未收到响应，它将使用回退模式重试。如果故障持续180秒，则AP会声明数据隧道关闭。

CAPWAP重新传输

核心原则：可靠性双向工作

CAPWAP控制消息使用请求和响应模型，即使它通过UDP运行。为了提供可靠性，发送请求的设备还负责重新传输请求，直到收到预期响应。

这意味着重新传输是对称的:

- 对于控制器到AP的请求（如配置更新），控制器会处理重传。
- 对于AP到控制器的请求，例如Discovery、Join、Configuration Status、映像相关消息和Echo Requests,AP处理重新传输。

这是一个重要的故障排除点。如果在数据包捕获中看到AP到控制器的重新传输，通常意味着AP只是重试自己的请求，因为它没有收到预期的响应。这在连接期间是正常的，在控制回声活动期间也可能发生。

控制器端重新传输行为

在控制器上，重新传输由专用传输可靠性状态机处理。

简单地说，控制器：

1. 接受需要确认的请求
2. 将其置于每个AP传输队列中
3. 发送请求
4. 等待匹配的响应
5. 当响应到达时，从队列中将其删除，或者在计时器到期时重新传输它

每个AP会话会单独跟踪此逻辑。控制器还保留传输窗口和未处理请求消息的计数。

控制器如何决定是否重新传输

每次重传计时器到期时，控制器都会检查排队的请求条目并作出以下决定之一：

1. 已收到的响应顺序混乱

如果已经收到对该请求的响应，即使该响应不按顺序到达，控制器也不会重新传输该消息。

2. Retry limit exceeded

如果重新传输请求已超过允许次数，控制器会将其视为故障并中止该AP会话的传输过程。

此时，AP会话终止。

3. 请求还不够旧

控制器不会在每个计时器事件中立即重新传输每个排队消息。请求在队列中至少必须保持重新传输间隔，然后才能再次发送。

默认情况下，此重传间隔为3秒。

4. 重新传输请求

如果请求仍处于待处理状态、已老化时间足够长，且未超过重试限制，则控制器会在CAPWAP控制信道上重新发送请求并递增重试计数器。

特殊情况：有线菊花链AP

对于使用有线菊花链AP路径的网状部署，控制器允许更大的重试预算。

在这种情况下，正常重试极限实际上是三倍。

如果默认重试计数5次，这意味着控制器最多可以重试15次，然后才宣告失败。

这种例外情况之所以存在，是因为这些拓扑可能需要更多延迟或消息丢失的容限。

AP端重新传输行为

AP也重新传输自己的CAPWAP请求，但它使用的模式与控制器不同。

当控制器使用固定的重传间隔时，AP使用指数回退。这意味着每次尝试失败后等待时间都会增加。

使用默认设置时，AP重试计时大约为：

- 6 秒
- 12 秒
- 24 秒
- 48 秒
- 96 秒

此行为适用于AP发起的CAPWAP请求，例如：

- 发现
- 加入
- 配置相关的交换
- 加密控制事务
- 回应请求

这些重试参数作为AP加入配置的一部分从控制器获取。

数据包捕获中的诊断指纹

在故障排除过程中，重新传输模式本身通常是一个有用的线索。

重新传输模式	可能来源
均匀间隔的重新传输，大约每3秒	控制器端重传
随着时间推移而分散的重新传输，如6秒、12秒、24秒、48秒、96秒	具有指数回退的AP端重传

这是确定哪一端正在重试的实用方法，尤其是在连接失败或与回应相关的问题期间。

在重试用完后会发生什么情况

如果重新传输继续而没有收到预期响应，两端最终都会放弃，但它们的恢复操作不同。

控制器端

如果控制器耗尽其重试预算，它会中止传输过程并终止AP会话。这会导致CAPWAP控制和数据会话被关闭。

AP端

如果AP用尽其自身的重试尝试，它会放弃该控制器并从开始重新开始，通常通过返回Discovery阶段并尝试完全重新加入。

配置旋钮和默认值

重新传输行为由AP加入配置文件中的两个主要设置控制。

命令	默认值	功能
capwap重传计数	5	最大重新传输尝试次数
capwap重传间隔	3 秒	基本重传间隔

这些值会影响加入可靠性和其他AP发起的CAPWAP控制重试，包括与回声相关的重试。

合并计时器参考

此表总结了本文讨论的主要CAPWAP计时器。

计时器	默认值	平面	所有者	描述
控制回声间隔	30 秒	Control	无线接入点	如果AP在30秒内未传输CAPWAP控制流量，则会发送回应请求。
控制心跳失效计时器	90 秒	Control	控制器	控制器期望此窗口内的有效控制流量。如果未收到任何信息，则控制会话将被视为关闭。
控制重传间隔	3 秒	Control	控制器	控制器以固定间隔重新传输其自己的未应答CAPWAP请求。
控制重传计数	5次尝试	Control	控制器	控制器发起的CAPWAP请求的最大重试计数。在有线菊花链方案中，尝试次数可增加到15次。
AP控制重传	6、12、24、	Control	无线接	AP使用指数回退重新传输自己的CAPWAP请求。

计时器	默认值	平面	所有者	描述
回退	48、96秒		入点	
数据保持连接间隔	30 秒	数据	无线接入点	在正常情况下，AP每30秒发送一次数据keepalive。
数据保持连接重试回退	3、6、12、15、15秒	数据	无线接入点	如果数据保持连接响应丢失，则AP使用回退重试，上限为15秒。
数据信道停顿间隔	180 秒	数据	无线接入点	如果AP在此时间段内未成功维护数据keepalive交换，则会声明数据隧道关闭。

并列差异

保活与重传

虽然它们有时会感到困惑，但keepalive和retransmission有不同的用途。

方面	Keepalive	重新传输
主要目的	验证对等体是否仍然可访问	未收到响应时重试特定请求
范围	每个会话或每个信道	每条消息
触发器	不活动或活动超时	请求仍未得到应答
跟踪方法	基于时间	基于重试计数
故障含义	通道或会话无法再访问	特定CAPWAP交换已反复失败
故障结果	可以声明会话已关闭	控制器可以终止会话，或者AP可以重新启动加入

RA跟踪日志字符串（按机制）

对于特定于AP的跟踪，收集AP的日志并搜索这些精确的字符串。

日志收集

使用:

- debug wireless mac <ap-radio-mac>
- show logging profile wireless filter mac <ap-radio-mac> to-file bootflash:<file>

控制keepalive/心跳

搜索：

- 处理回应请求
- 构建回应响应
- 为序列号为<N>的请求发送的回应响应。
- 心跳计时器已重新启动，其新值为<ms>
- 已调用心跳计时器触摸
- 心跳计时器到期

数据保持连接

搜索：

- 已收到数据keepalive
- 已建立数据会话
- 未建立数据DTLS
- 更新CAPWAP数据隧道条目
- 无法处理数据keepalive。理由：CAPWAP会话未处于运行或配置状态
- keepalive消息无效，无capwap块
- 保活消息无效，无WTP MAC
- 无法处理keepalive，找不到CAPWAP会话
- 除控制端口之外的其他端口上接收了数据保活：<port>
- 收到不同AP <mac>的数据保持连接
- 无法更新远程主机的WTP会话表中的数据元组信息：<ip>
- 无法更新CAPWAP数据隧道条目

数据平面保持连接处理

搜索：

- 从<src>到<dst>接收的keep alive数据
- 已发送<enc/plain>数据保活响应
- 正在从IP丢弃保持连接数据包：<ip>，未知会话
- 正在从IP丢弃保持连接数据包：<ip>,，无法获取数据DTLS状态
- 无法发送数据保活响应
- 负载均衡数据保持连接消息失败

重新传输

搜索：

- 重新传输计时器回调
- 正在重新传输seqnum <N>，因为它处于队列中且尚未获取ack
- 正在重新传输消息：<msg-name> retransmit attempt:<M>
- seq num条目：<N>在队列中<S>秒。不重新传输
- CAPWAP会话终止，最大重新传输过期时间：<msg> ...
- AP离开时的CAPWAP会话资源清理。

会话断开

搜索：

- 正在终止AP CAPWAP会话。
- 关闭CAPWAP DTLS会话。
- capwap-dtls控制会话关闭
- capwap-dtls数据会话关闭
- <S>秒前收到的最后一个控制数据包。
- <S>秒前收到的最后一个数据保持连接数据包。
- AP的CAPWAP DTLS会话已关闭，原因：<reason>

示例：

工作场景

AP调试验证

此AP debug命令可用于监控AP和WLC之间的CAPWAP控制和数据keepalive通信。

```
#debug capwap client event
```

这些调试日志展示了一个成功的CAPWAP通信序列。

在13:11:44,AP通过UDP端口5246向WLC发送CAPWAP回应请求。在同一间隔内，AP还通过UDP端口5247传输CAPWAP数据保持连接数据包。日志确认WLC已成功响应这两个请求。

时间戳表示正常的CAPWAP通信周期：

- 13:11:44.0917 — 已传输回应请求。
- 13:11:44.0918 — 数据保持连接已传输。
- 13:11:44.0926 -从WLC接收的数据Keepalive。
- 13:11:44.0940 -从WLC接收的回应响应。

这些时间戳确认CAPWAP控制通道和数据通道均按预期运行，往返延迟可忽略不计。

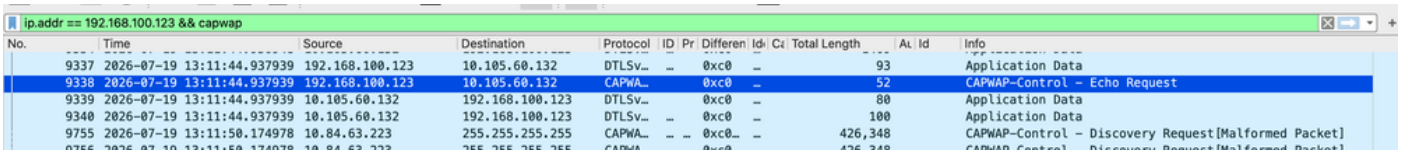
```
[*07/19/2026 13:11:14.0817] [RX]KEEPALIVE: RoundTripTime=0.001 sec
[*07/19/2026 13:11:44.0917] Echo Request: Send count 22
[*07/19/2026 13:11:44.0917] [TX]Echo Request: Sent to 10.105.60.132
[*07/19/2026 13:11:44.0918] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 13:11:44.0918] [TX]KEEPALIVE: Schedule for Retransmit in 3 sec sec_drop_count=0
[*07/19/2026 13:11:44.0918] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 13:11:44.0926] chatter: [RX]KEEPALIVE: Count 164
[*07/19/2026 13:11:44.0927] Sending KEEPALIVE to WTP SM
[*07/19/2026 13:11:44.0927] Capwap data keep-alive Msg.
[*07/19/2026 13:11:44.0927] [RX]KEEPALIVE: Session ID 3221108139, Next scheduled for TX in 30 sec
[*07/19/2026 13:11:44.0927] [RX]KEEPALIVE: RoundTripTime=0.001 sec
[*07/19/2026 13:11:44.0940] [RX]Echo Response from 10.105.60.132
[*07/19/2026 13:11:44.0004] [RX]Echo Response from 10.105.60.132 RttCount 1
[*07/19/2026 13:12:14.0004] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 13:12:14.0005] [TX]KEEPALIVE: Schedule for Retransmit in 3 sec sec_drop_count=0
[*07/19/2026 13:12:14.0005] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
```

CAPWAP控制数据包验证（UDP端口5246）

数据包捕获分析

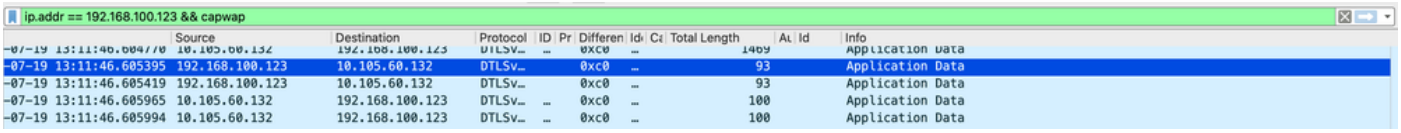
数据包捕获确认AP在13:11:44通过UDP端口5246生成CAPWAP回应请求。

WLC成功接收数据包，处理了请求，并立即生成相应的回应响应。由于CAPWAP控制信道是使用DTLS加密进行保护的，因此响应在数据包捕获中显示为加密的应用数据。



交换机数据包验证

交换机数据包捕获可确认已加密的CAPWAP控制数据包已成功从WLC接收并转发到AP。

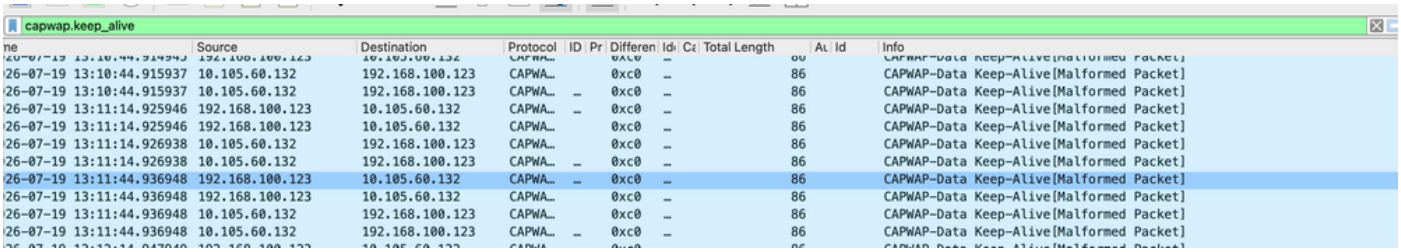


CAPWAP数据保持连接验证（UDP端口5247）

AP通过UDP端口5247定期传输CAPWAP数据保持连接数据包，以验证CAPWAP数据隧道的运行状况。

在13:11:44,AP向WLC发送了“数据保持连接”数据包。WLC成功接收了数据包，并立即以相应的keepalive响应做出响应。

此成功交换确认CAPWAP数据路径保持运行并且AP和WLC之间的双向通信运行正常



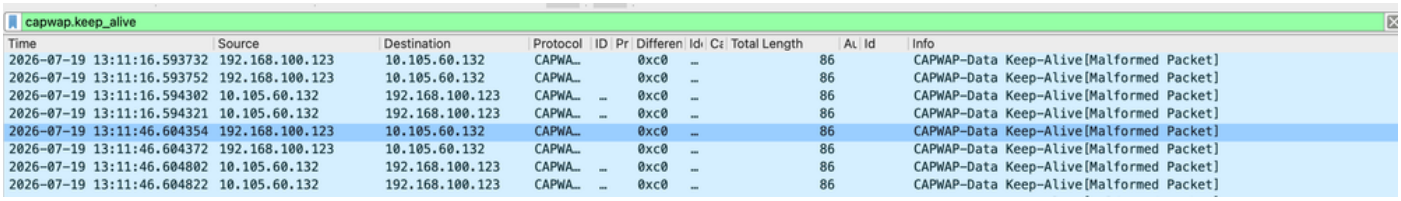
交换机数据包验证

交换机数据包捕获进一步验证CAPWAP Data Keepalive数据包是否在AP和WLC之间成功转发而没有中断。

观察到的数据包流确认：

- 数据保持连接数据包成功到达WLC。
- WLC生成了预期的keepalive响应。
- 交换机将响应转发回AP。

- 转发路径上未发生数据包丢失。


capwap.keep_alive

Time	Source	Destination	Protocol	ID	Pr	Differen	Id	Ca	Total Length	Al	Id	Info
2026-07-19 13:11:16.593732	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
2026-07-19 13:11:16.593752	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
2026-07-19 13:11:16.594302	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
2026-07-19 13:11:16.594321	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
2026-07-19 13:11:46.604354	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
2026-07-19 13:11:46.604372	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
2026-07-19 13:11:46.604802	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
2026-07-19 13:11:46.604822	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]

AP上行链路交换机上丢弃的控制数据包(UDP 5246)

此测试演示当CAPWAP控制端口(UDP 5246)在AP上行链路交换机上丢弃时AP的行为。

目标是在CAPWAP控制数据包无法到达AP时验证AP、WLC和网络行为，尽管WLC已成功处理和响应请求。

在这种情况下：

- AP继续向WLC发送CAPWAP回应请求。
- WLC成功接收和处理每个回应请求。
- WLC生成相应的回应响应。
- AP上行链路交换机收到响应，但不会将其转发回AP。
- 由于AP从未收到回应响应，因此它最终超过了最大重新传输计数，并重新启动CAPWAP状态机。

AP调试分析

在12:11:55.4568,AP通过UDP端口5246向WLC发送CAPWAP回应请求。

回应请求:发送计数0

与正常工作场景不同，未收到回应响应。因此，AP根据默认CAPWAP计时器发起重新传输。

重新传输发生在以下时间戳：

时间	Event
12:12:00.2587	重新传输计数= 1

时间	Event
12:12:03.2599	重新传输计数= 2
12:12:06.2610	重新传输计数= 3
12:12:09.2624	重新传输计数= 4
12:12:12.2637	重新传输计数= 5

在第五次重新传输失败后，AP声明无法访问CAPWAP控制会话。

在12:12:15.2647上，美联社报道：

已超过最大重新传输计数，返回到DISCOVER模式。

之后，AP立即重新启动CAPWAP状态机以启动新的发现过程。

```
[*07/17/2026 12:11:54.2577] [RX]KEEPALIVE: Session ID 4068929293, Next scheduled for TX in 30 sec
[*07/17/2026 12:11:54.2577] [RX]KEEPALIVE: RoundTripTime=0.001 sec
[*07/17/2026 12:11:55.4568] Echo Request: Send count 0
[*07/17/2026 12:11:55.4568] [TX]Echo Request: Sent 1 Lost 269
[*07/17/2026 12:12:00.2587] Re-Tx Count=1, Max Re-Tx Value=5, SendSeqNum=63, NumofPendingMsgs=3
[*07/17/2026 12:12:03.2599] Re-Tx Count=2, Max Re-Tx Value=5, SendSeqNum=63, NumofPendingMsgs=3
[*07/17/2026 12:12:06.2610] Re-Tx Count=3, Max Re-Tx Value=5, SendSeqNum=63, NumofPendingMsgs=3
[*07/17/2026 12:12:09.2624] Re-Tx Count=4, Max Re-Tx Value=5, SendSeqNum=63, NumofPendingMsgs=3
[*07/17/2026 12:12:12.2637] Re-Tx Count=5, Max Re-Tx Value=5, SendSeqNum=63, NumofPendingMsgs=3
[*07/17/2026 12:12:15.2647] Max retransmission count exceeded, going back to DISCOVER mode.
[*07/17/2026 12:12:15.2647] Failed to reach capwap down with retransmission 3 times
```

WLC RA跟踪确认控制器未遇到任何处理问题。

在12:11:58.731835712,WLC成功接收了AP发送的CAPWAP回应请求。这些日志表明WLC成功处理了请求并生成了相应的响应。稍后，在12:12:19.802183814,WLC从AP收到DTLS Close Notify。AP断开连接，因为它从未收到控制器发送的回应响应。因此，WLC终止了DTLS会话并记录了AP取消关联。

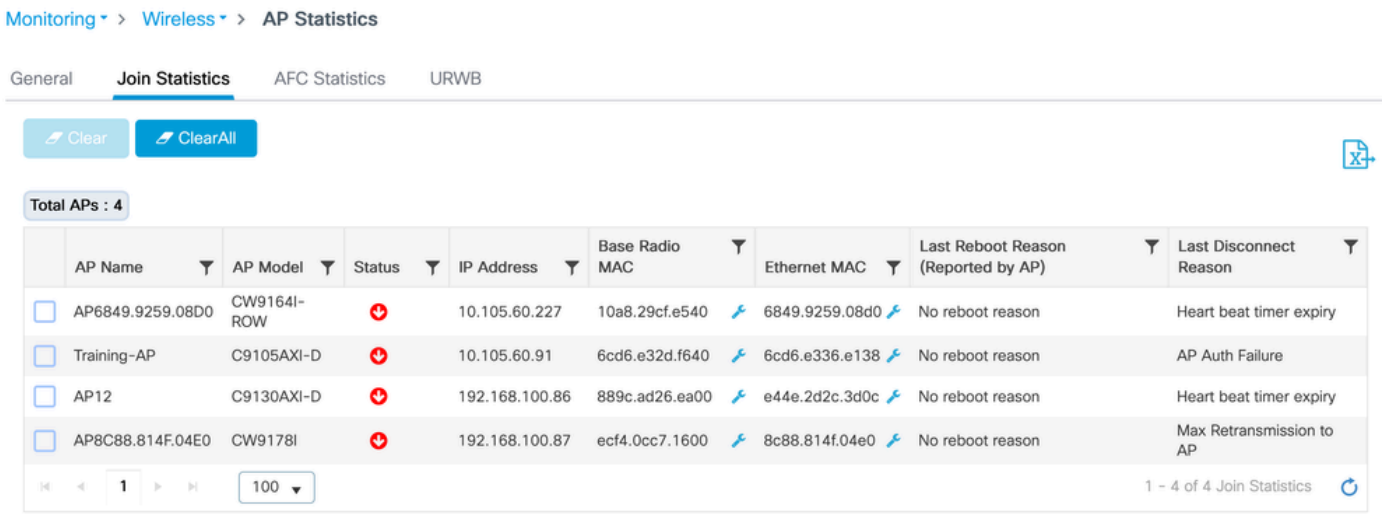
<#root>

```
2026/07/17 12:12:19.802274790 {wncd_x_R0-1}{2}: [ewlc-dtls-sess] [16522]: (info):
```

```
Remote Host: 192.168.100.120[5256] MAC: 889c.ad26.ea00  dtls session closed


2026/07/17 12:12:19.802279648 {wncd_x_R0-1}{2}: [ewlc-infra-capwap-dgram] [16522]: (debug): dgram handl
2026/07/17 12:12:19.802317470 {wncd_x_R0-1}{2}: [ewlc-capwapmsg-sess] [16522]: (debug): Encrypted DTLS
2026/07/17 12:12:19.802321202 {wncd_x_R0-1}{2}: [capwapac-smgr-srvr] [16522]: (debug): Mac: 889c.ad26.e
2026/07/17 12:12:19.802376588 {wncd_x_R0-1}{2}: [ap-join-info-db] [16522]: (note): MAC: 889c.ad26.ea00

AP disconnect initiated. Name : AP12, Ethernet mac : e44e.2d2c.3d0c, Reason: DTLS close alert from peer,
```

<#root>

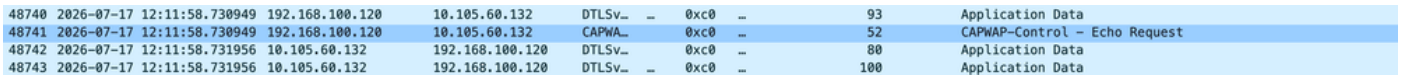
```
VK-WLC#show wireless stats ap history | i AP12


AP12                              889c.ad26.ea00  Joined      07/17/26 12:24:18  NA
AP12                              889c.ad26.ea00  Disjoined   07/17/26 12:12:19  NA
AP12                              889c.ad26.ea00  Joined      07/17/26 12:09:25  NA

AP12                           889c.ad26.ea00  Disjoined   07/17/26 12:08:33  NA                    Heart be
```

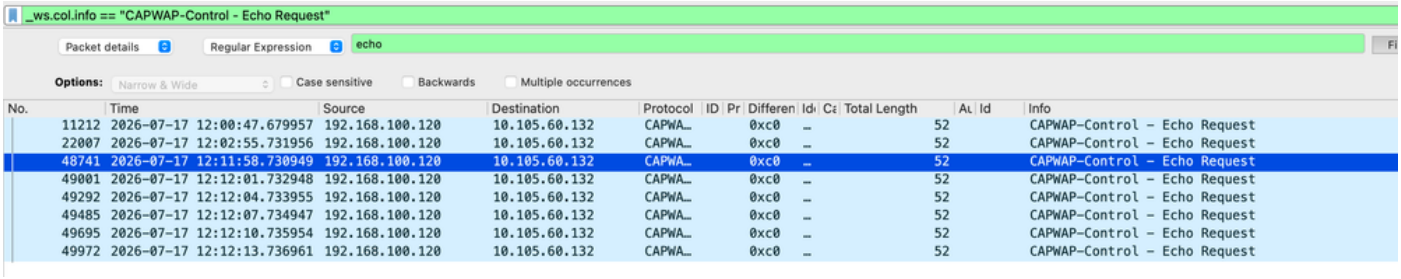


WLC EPC确认：

- 在相同的时间戳下，CAPWAP回应请求成功到达WLC。
- WLC生成了加密的CAPWAP回声响应。
- 由于CAPWAP控制流量受DTLS加密保护，因此响应显示为Application Data。

因此，EPC确认控制器成功传输了响应。这消除了WLC作为问题的来源。





在AP上行链路交换机上收集的数据包捕获提供了最终证据。

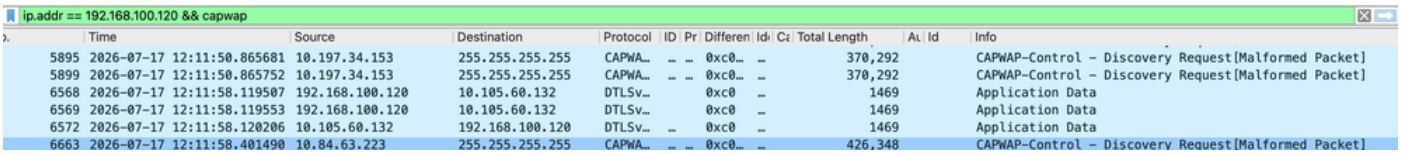
捕获表明：

- 交换机成功接收到WLC发送的加密回应响应。
- 响应显示为Application Data，这是因为DTLS加密而预期的。
- 响应未转发到AP。

这就解释了原因：

- AP从未收到CAPWAP回应响应。
- AP继续重新传输回应请求。
- 重新传输计数器最终超过了配置的阈值。
- AP重新启动了CAPWAP状态机。

数据包捕获清楚地将交换机标识为CAPWAP控制流量中断的点



AP上行链路交换机上丢弃的CAPWAP数据端口(UDP 5247)

当CAPWAP控制端口(UDP 5246)保持正常运行时，AP上行链路交换机上丢弃CAPWAP数据端口(UDP 5247)时，此测试将验证AP的行为。

与前一个场景不同，AP通过成功通过UDP端口5246交换CAPWAP回应消息来继续与WLC保持CAPWAP控制连接。但是，由于CAPWAP数据保持连接数据包无法完成往返过程，AP最终会宣布CAPWAP数据路径不可达，并启动CAPWAP重新启动。

AP调试日志确认CAPWAP控制信道在整个测试期间保持正常运行。

在捕获开始时，通过UDP端口5246传输的CAPWAP回应请求继续从WLC接收有效的回应响应，从而确认不间断的控制平面通信。

但是，在14:30:15,AP通过UDP端口5247传输了CAPWAP数据保持连接数据包。由于未收到对应的数据保持连接响应，因此AP启动了重试机制。可以在以下时间戳上观察重新传输的情况：

时间戳	Event
14:30:15	初始数据保持连接已传输
14:30:19	重试1
14:30:25	重试2
14:30:37	重试3
14:30:49	重试4
14:31:01	重试5
14:31:13	正在进行最终的重试

虽然CAPWAP回应响应在此期间继续接收，但AP未收到任何对数据保持连接数据包的响应。在重试尝试用完后，AP报告了未加密的数据保持连接超时并启动了CAPWAP重新启动。在大约14:31:16处，AP终止了现有的CAPWAP会话并返回到发现过程。

```
AP12#[*07/19/2026 14:30:15.9995] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 14:30:15.9995] [TX]KEEPALIVE: Schedule for Retransmit in 3 sec sec_drop_count=0
[*07/19/2026 14:30:15.9996] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 14:30:19.0002] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 14:30:19.0002] [TX]KEEPALIVE: Schedule for Retransmit in 6 sec sec_drop_count=0
[*07/19/2026 14:30:19.0003] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 14:30:25.0023] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 14:30:25.0024] [TX]KEEPALIVE: Schedule for Retransmit in 12 sec sec_drop_count=0
[*07/19/2026 14:30:25.0024] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 14:30:37.0066] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 14:30:37.0066] [TX]KEEPALIVE: Schedule for Retransmit in 12 sec sec_drop_count=0
[*07/19/2026 14:30:37.0067] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 14:30:49.0109] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 14:30:49.0109] [TX]KEEPALIVE: Schedule for Retransmit in 12 sec sec_drop_count=0
```

```
[*07/19/2026 14:30:49.0109] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 14:31:01.0151] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 14:31:01.0151] [TX]KEEPALIVE: Schedule for Retransmit in 12 sec sec_drop_count=0
[*07/19/2026 14:31:01.0152] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 14:31:13.0195] [TX]KEEPALIVE: Send to 10.105.60.132-5247
[*07/19/2026 14:31:13.0195] [TX]KEEPALIVE: Schedule for Retransmit in 12 sec sec_drop_count=0
[*07/19/2026 14:31:13.0196] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
[*07/19/2026 14:31:16.0207] Warning, unencrypted data keepalive failed
[*07/19/2026 14:31:16.0207] Going to restart CAPWAP (reason : data keepalive not received)...
```

对于CAPWAP数据通道，跟踪指示控制器或者未收到预期的数据保持连接数据包。最终，在AP声明数据保持连接失败后，WLC记录了DTLS会话终止和AP断开事件。在RA跟踪中观察到的序列确认控制器保持运行，直到AP因数据保持连接超时而自动断开连接。

RA跟踪与AP以太网MAC:

<#root>

```
2026/07/19 14:31:16.842212773 {fman_rp_R0-0}{2}: [source] [20448]: (debug): ipc(mqipc/wncd_2/wncd-fmrp)
2026/07/19 14:31:16.842250177 {wncd_x_R0-2}{2}: [errmsg] [16638]: (note): %CAPWAPAC_SMGR_TRACE_MESSAGE-
2026/07/19 14:31:16.842251233 {wncd_x_R0-2}{2}: [capwapac-smgr-sess-fsm] [16638]: (note): Mac: 889c.ad2
```

Last Data Keep Alive Packet received 90 seconds ago.

```
2026/07/19 14:31:16.843318439 {wncmgrd_R0-0}{2}: [loadbalance-algo] [16262]: (note): Algo counter decre
2026/07/19 14:31:16.843318599 {wncd_x_R0-2}{2}: [wsa-core] [16638]: (debug): WSA AP EVT Create Populate
2026/07/19 14:31:16.843320409 {wncd_x_R0-2}{2}: [wsa-core] [16638]: (debug): WSA AP EVT Create Populate
```

DISJOIN - AP Mac:889c.ad26.ea00 , new ap disconnect reason 'DTLS close alert from peer' (26)

无线电MAC的RA跟踪：

<#root>

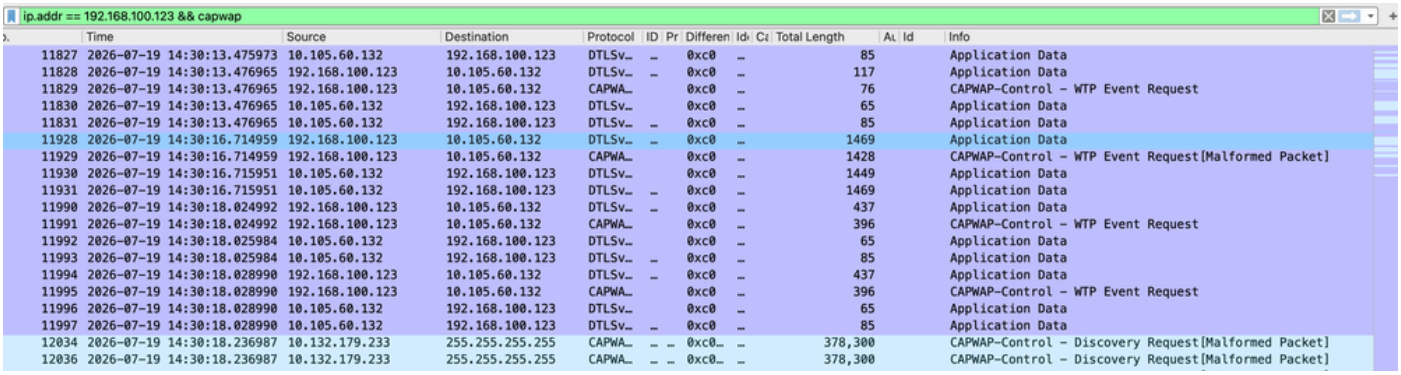
```
2026/07/19 14:31:16.737070835 {wncd_x_R0-2}{2}: [capwapac-smgr-sess] [16638]: (debug): Mac: 889c.ad26.e
2026/07/19 14:31:16.737072831 {wncd_x_R0-2}{2}: [capwapac-smgr-srvr] [16638]: (debug): Mac: 889c.ad26.e
2026/07/19 14:31:16.737076419 {wncd_x_R0-2}{2}: [msc-fsm] [16638]: (debug): @msc_event {"entity":"/capw
2026/07/19 14:31:16.737078137 {wncd_x_R0-2}{2}: [msc-fsm] [16638]: (debug): @msc_event {"entity":"/capw
2026/07/19 14:31:16.841870791 {wncd_x_R0-2}{2}: [ap-join-info-db] [16638]: (note): MAC: 889c.ad26.ea00
```

AP disconnect initiated. Name : AP12, Ethernet mac : e44e.2d2c.3d0c, Reason: DTLS close alert from peer,

```
2026/07/19 14:31:16.841890831 {wncd_x_R0-2}{2}: [capwapac-smgr-srvr] [16638]: (debug): MAC: 889c.ad26.e
```

在WLC上收集的EPC验证控制器端数据包处理。捕获确认CAPWAP控制数据包在整个测试期间继

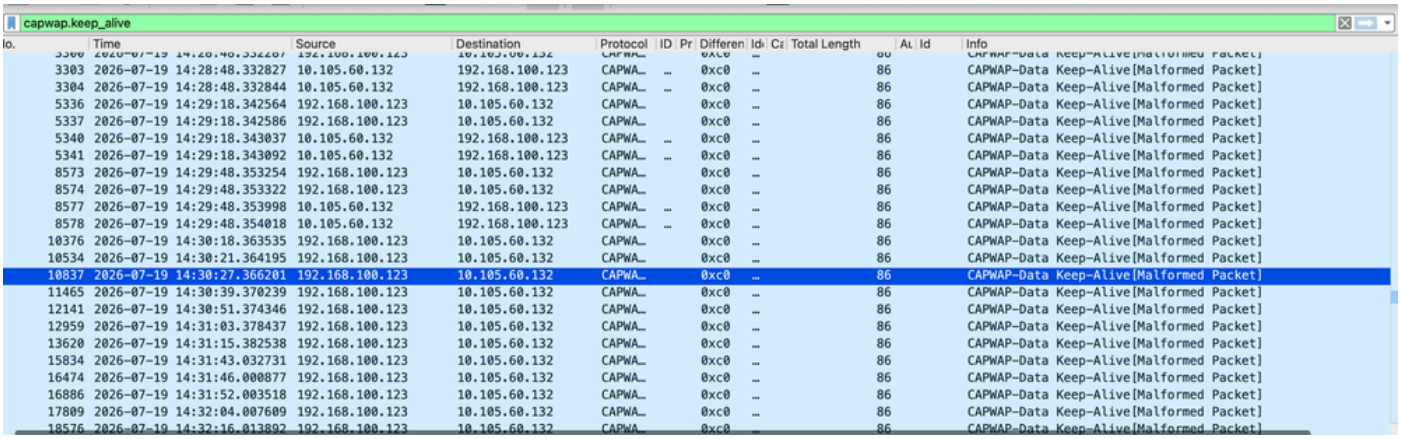
续成功交换，但wlc上未收到数据keepalive数据包。此观察与AP调试日志一致，并且表明Data Keepalive机制在控制通道保持活动状态的情况下仍失败。


ip.addr == 192.168.100.123 && capwap

	Time	Source	Destination	Protocol	ID	Pr	Differen	Id	Ca	Total Length	Al	Id	Info
11827	2026-07-19 14:30:13.475973	10.105.60.132	192.168.100.123	DTLSv…	…		0xc0	…		85			Application Data
11828	2026-07-19 14:30:13.476965	192.168.100.123	10.105.60.132	DTLSv…	…		0xc0	…		117			Application Data
11829	2026-07-19 14:30:13.476965	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		76			CAPWAP-Control - WTP Event Request
11830	2026-07-19 14:30:13.476965	10.105.60.132	192.168.100.123	DTLSv…			0xc0	…		65			Application Data
11831	2026-07-19 14:30:13.476965	10.105.60.132	192.168.100.123	DTLSv…	…		0xc0	…		85			Application Data
11928	2026-07-19 14:30:16.714959	192.168.100.123	10.105.60.132	DTLSv…	…		0xc0	…		1469			Application Data
11929	2026-07-19 14:30:16.714959	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		1428			CAPWAP-Control - WTP Event Request[Malformed Packet]
11930	2026-07-19 14:30:16.715951	10.105.60.132	192.168.100.123	DTLSv…			0xc0	…		1449			Application Data
11931	2026-07-19 14:30:16.715951	10.105.60.132	192.168.100.123	DTLSv…	…		0xc0	…		1469			Application Data
11990	2026-07-19 14:30:18.024992	192.168.100.123	10.105.60.132	DTLSv…	…		0xc0	…		437			Application Data
11991	2026-07-19 14:30:18.024992	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		396			CAPWAP-Control - WTP Event Request
11992	2026-07-19 14:30:18.025984	10.105.60.132	192.168.100.123	DTLSv…			0xc0	…		65			Application Data
11993	2026-07-19 14:30:18.025984	10.105.60.132	192.168.100.123	DTLSv…	…		0xc0	…		85			Application Data
11994	2026-07-19 14:30:18.028990	192.168.100.123	10.105.60.132	DTLSv…	…		0xc0	…		437			Application Data
11995	2026-07-19 14:30:18.028990	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		396			CAPWAP-Control - WTP Event Request
11996	2026-07-19 14:30:18.028990	10.105.60.132	192.168.100.123	DTLSv…			0xc0	…		65			Application Data
11997	2026-07-19 14:30:18.028990	10.105.60.132	192.168.100.123	DTLSv…	…		0xc0	…		85			Application Data
12034	2026-07-19 14:30:18.236987	10.132.179.233	255.255.255.255	CAPWA…	…	…	0xc0…	…		378,300			CAPWAP-Control - Discovery Request[Malformed Packet]
12036	2026-07-19 14:30:18.236987	10.132.179.233	255.255.255.255	CAPWA…	…	…	0xc0…	…		378,300			CAPWAP-Control - Discovery Request[Malformed Packet]

交换机：

在AP上行链路交换机上收集的数据包捕获显示，虽然交换机上存在数据包，但数据包未转发到wlc


capwap.keep_alive

lo.	Time	Source	Destination	Protocol	ID	Pr	Differen	Id	Ca	Total Length	Al	Id	Info
3303	2026-07-19 14:28:48.332827	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
3304	2026-07-19 14:28:48.332844	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
5336	2026-07-19 14:29:18.342564	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
5337	2026-07-19 14:29:18.342586	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
5340	2026-07-19 14:29:18.343037	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
5341	2026-07-19 14:29:18.343092	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
8573	2026-07-19 14:29:48.353254	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
8574	2026-07-19 14:29:48.353322	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
8577	2026-07-19 14:29:48.353998	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
8578	2026-07-19 14:29:48.354018	10.105.60.132	192.168.100.123	CAPWA…	…		0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
10376	2026-07-19 14:30:18.363535	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
10534	2026-07-19 14:30:21.364195	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
10837	2026-07-19 14:30:27.366201	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
11465	2026-07-19 14:30:39.370239	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
12141	2026-07-19 14:30:51.374346	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
12959	2026-07-19 14:31:03.378437	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
13620	2026-07-19 14:31:15.382538	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
15834	2026-07-19 14:31:43.032731	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
16474	2026-07-19 14:31:46.000877	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
16886	2026-07-19 14:31:52.003518	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]
17809	2026-07-19 14:32:04.007609	192.168.100.123	10.105.60.132	CAPWA…			0xc0	…		86			CAPWAP-Data Keep-Alive[Malformed Packet]

CAPWAP数据保持连接禁用

此功能是在Cisco Bug ID CSCvs66015下引入的

此功能对于排除这些场景故障非常有用。

1. 排除AP断开故障：在禁用capwap数据保持连接后，我们可以看到由于capwap控制保持连接而AP是否仍在断开连接，并确定AP保持连接断开的根本原因。
2. 避免将capwap data-keepalive作为解决方法而断开连接，直到确定capwap data keepalive丢弃的原因。
3. 通过低吞吐量广域网链路连接的分支机构灵活部署中嗅探器AP。如果想要收集OTA数据包捕获，并将其中一个AP配置为嗅探器模式，则所有捕获的数据包都将使用capwap数据端口通过WAN链路传输到WLC。如果这样做，就会淹没WAN链路并影响整个分支机构。

4. 如果在嗅探器模式下禁用AP的capwap数据保持连接，并在分支中创建一个ACL以丢弃来自该具体AP的capwap数据包，则AP保持连接，因为capwap控制正常，然后您可以从AP交换机端口收集OTA，而不会用数据包捕获导致WAN链路拥塞。

从9800 WLC触发COS-AP数据保持连接启用/禁用。默认情况下，在WLC和AP中均启用数据保持连接。

WLC命令：

1)显示带有字符串“Unencrypted Data Keep Alive”的状态

```
show ap config general
```

2)使用AP名称启用/禁用数据保持连接

```
ap name AP-NAME keepalive
ap name AP-NAME no keepalive
```

从AP自身触发COS-AP数据保持连接启用/禁用。默认情况下，AP中的数据保持连接处于启用状态。

无线接入点命令：

1)使用字符串“Unencrypted Data Keep Alive”显示状态

```
show capwap client config
```

2)在AP中启用/禁用数据保持连接

```
capwap ap unencrypted_data_keepalive enable
capwap ap unencrypted_data_keepalive disable
```

例如:

已禁用AP级别上的Keepalive检查：

```
AP12#capwap ap unencrypted_data_keepalive disable
```

<#root>

```
AP12#show capwap client configuration
```

```
AdminState                          : ADMIN_ENABLED(1)
Name                                : AP12
Location                            : default location
Primary controller name             : VK-WLC
Primary controller IP               : 10.105.60.132
Secondary controller name           : 9800-demo
Secondary controller IP             : 10.106.39.156
Tertiary controller name            :
ssh status                          : Enabled
ApMode                              : Local
ApSubMode                           : Not Configured
Link-Encryption                     : Disabled

Unencrypted Data Keep Alive         : Disabled

OfficeExtend AP                     : Disabled
Discovery Timer                     : 10
```

无线接入点调试：

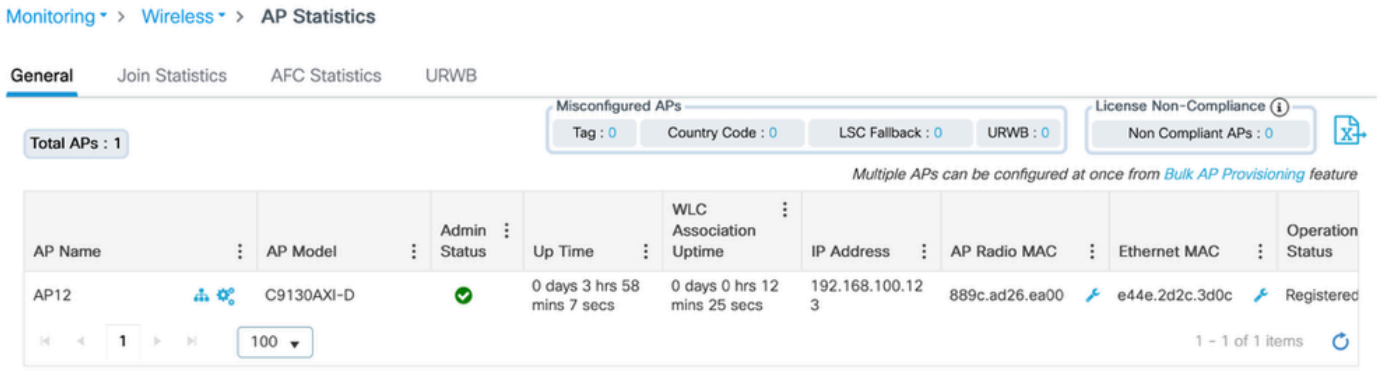
在ap调试中，我们可以看到AP和wlc之间没有发生数据保持连接数据包交换，我们只看到控制数据包交换。

AP12#debug capwap client keepalive

```
AP12#[*07/19/2026 15:24:33.4087] Echo Request: Send count 0
[*07/19/2026 15:24:33.4087] [TX]Echo Request: Sent to 10.105.60.132
[*07/19/2026 15:24:33.4112] [RX]Echo Response from 10.105.60.132
[*07/19/2026 15:24:34.0006] [RX]Echo Response from 10.105.60.132 RttCount 1
[*07/19/2026 15:25:34.0032] Echo Request: Send count 1
[*07/19/2026 15:25:34.0032] [TX]Echo Request: Sent 2 Lost 2
[*07/19/2026 15:25:34.0056] [RX]Echo Response from 10.105.60.132
[*07/19/2026 15:25:34.0006] [RX]Echo Response from 10.105.60.132 RttCount 2
[*07/19/2026 15:27:34.0327] Echo Request: Send count 2
[*07/19/2026 15:27:34.0327] [TX]Echo Request: Sent 3 Lost 2
[*07/19/2026 15:27:34.0347] [RX]Echo Response from 10.105.60.132
[*07/19/2026 15:27:34.0004] [RX]Echo Response from 10.105.60.132 RttCount 1
```

```
[*07/19/2026 15:28:34.0025] Echo Request: Send count 3
[*07/19/2026 15:28:34.0025] [TX]Echo Request: Sent 4 Lost 2
[*07/19/2026 15:28:34.0050] [RX]Echo Response from 10.105.60.132
[*07/19/2026 15:28:34.0022] [RX]Echo Response from 10.105.60.132 RttCount 2
AP12#[*07/19/2026 15:29:43.5772] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.
[*07/19/2026 15:30:04.0140] Echo Request: Send count 4
[*07/19/2026 15:30:04.0140] [TX]Echo Request: Sent 5 Lost 2
[*07/19/2026 15:30:04.0164] [RX]Echo Response from 10.105.60.132
[*07/19/2026 15:30:04.0011] [RX]Echo Response from 10.105.60.132 RttCount 1
[*07/19/2026 15:30:27.7695] capwap_ap_upstream_send ip: 192.168.100.123, port: 5272 => ip: 10.105.60.13
```

即使数据keepalive数据包被丢弃，但由于我们禁用了数据keepalive检查，我们可以看到AP在wlc上保持稳定，而不会受到keepalive数据包丢弃的影响。



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）。