基于LACP协议的链路聚合状态机模块的实现①

郑　涛　郭裕顺 (杭州电子科技大学 电子信息学院 浙江 杭州 310018)

摘　要： 首先介绍了链路聚合技术的背景和基本概念，接着说明了 LACP 协议(Link Aggregation Control Protocol)的内容和原理，其中 LACP 状态机模块控制着整个 LACP 协议的运转，是保证基于 LACP 协议的链路聚合可以有效工作的核心和关键模块。最后以 802.3ad 的 LACP 协议为基础，利用有限状态机的设计方法，实现并优化了 LACP 协议状态机模块的功能。经测试，该模块功能运行正常，能够正确处理协议运行的各种情况。

关键词： Link Aggregation；动态链路聚合；LACP；有限状态机；网络协议

Design of Link Aggregation Finite State Machine Module Based on LACP

ZHENG Tao, GUO Yu-Shun

(School of Electronics & Information, Hangzhou Dianzi University, Hangzhou 310018, China)

Abstract: This article first introduces the background and the basic concepts of link aggregation technology. It then describes the contents and principles of LACP. LACP state machine module controls the operation of the LACP. It is the core and key module to ensure LACP work effectively. Finally the LACP protocol state machine module function is implemented and optimized based on LACP and Finite state machine design method. In testing, the module operates normally. It can handle all circumstances correctly when the protocol is running.

Keywords: link aggregation; dynamic link aggregation; LACP; finite state machine; network protocol

网络技术的快速发展使得网络需要传输的数据量急剧增加，网络的带宽成为提高网络服务质量的主要瓶颈之一，越来越多的用户面临网络带宽不足的问题，最直接的手段当然是把旧网络升级成新网络来增加网络带宽。但是这种方法成本太高，而链路聚合技术提供了一种更廉价的选择。链路聚合(Link Aggregation)技术简言之就是将多条物理链路聚合成一条带宽更高的逻辑链路，该逻辑链路的带宽等于被聚合在一起的多条物理链路的带宽之和。聚合在一起的物理链路的条数可以根据业务的带宽需求来配置。因此链路聚合具有成本低，配置灵活的优点，此外，链路聚合还具有链路冗余备份的功能，聚合在一起的链路彼此动态备份，提高了网络的稳定性。早期链路聚合技术的实现没有统一的标准，各厂商都有自己私有的解决方案，功能不完全相同，也互不兼容。因此，IEEE专门制定了链路聚合的标准，目前链路聚合技术的正式标准为 IEEE Standard 802.3ad，而 LACP 是该标准的主要内容之一，是一种实现链路动态聚合的协议[1]。

1　LACP的原理

采用 LACP 聚合的双方(分别称为 Actor 和 Partner)通过称之为 LACPDU(LACP Data Unit)的协议报文来交互本端(Actor)和对端(Partner)的聚合信息，以对整个链路聚合的认识达成一致。协议报文主要包含以下信息：本端和对端系统优先级、本端和对端系统 ID、本端和对端的端口操作 key、本端和对端的端口优先级、本端和对端的端口 ID、本端和对端的端口状态[2]。聚合的双方就根据这些信息，按照一定的选择算法选择合适的链路，控制聚合的状态。被选中的成员链路可以正常转发流量，而未被选中的成员

① 收稿时间:2009-09-07;收到修改稿时间:2009-10-26

链路将被置为阻塞状态，不能转发任何流量。聚合链路的总带宽等于被选中的成员链路的带宽之和，并且聚合链路上的流量会按照一定的规则分担到各个选中的成员链路上，由于 LACPDU 是周期性交互，即聚合的双方每隔一段时间便互发一次协议报文，所以当有选中成员链路因为某种原因不能工作时，链路聚合可以很快的感知到，并重设链路状态，置该链路为阻塞，流量被重分配给其他选中成员链路。这样就实现了增加带宽，链路动态备份的功能。

2 LACP协议状态机的实现

动态链路聚合是一个完全的事件驱动系统，它的整个工作过程中的每个动作都是由事件触发的，并且这些事件的发生的顺序和时间是随机的，事件能以任何顺序、在任何时刻出现。而对于这种系统，有限状态机是一种最直观和有效的方法[3]。为了能够高效稳定的控制动态链路聚合的运行，本文以 802.3ad 的 LACP 协议为基础，运用有限状态机方法，通过一组状态机来管理和控制 LACP 协议的运转，主要有 RX 状态机(Receive machine)、PTX 状态机(Periodic Transmission machine)、MUX 状态机(Mux machine)和 TX 状态机(Transmit machine)。它们在加入聚合链路的每个成员端口都会运行，这一组状态机控制 LACP 的各个方面，包括本端 LACP 协议报文的发送及发送的周期、解析收到的对端 LACP 协议报文、调整链路的状态、处理各种异常情况等等。它是实现动态链路聚合的核心模块之一。

2.1 状态机模块整体结构

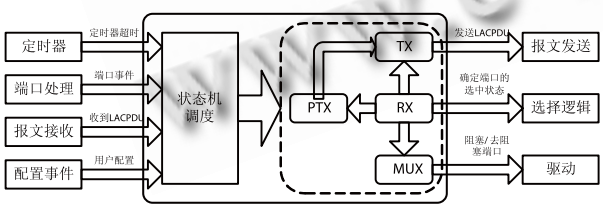

图 1 状态机模块整体结构图

整体状态机模块的结构如图 1 所示，为了降低模块间的耦合性，用状态机调度子模块作为处理状态机外部事件的统一接口。这些外部事件包括定时器超时事件、端口事件等等。各个状态机将无法直接响应这些外部事件。状态机调度子模块根据不同的事件激励相应的状态机。比如当收到 LACPDU 时，状态机调度模块首先处理，再激励 RX 状态机，RX 状态机解析报文中的信息后调用选择逻辑模块设置端口的选中状态，选择逻辑模块负责通过一定的选择算法选择合适的端口，只有被选中的端口才可以转发数据。这一组协议状态机和状态机调度模块有机的结合，分工合作，完成对协议运行的控制。各个状态机的具体功能在下一小节介绍。

2.2 状态机的功能与设计

2.2.1 RX 状态机

RX 状态机(接收状态机)：主要用来处理接收到的 LACP 协议报文，解析报文，记录对端的聚合相关信息，并调用选择逻辑模块来设置端口的选中状态，根据对端的信息，设置链路聚合的相关数据，以及判断本端所保存的对端聚合相关的数据是否老化并做相应的处理，并且激励 LACP 协议的其它状态机运行，它是 LACP 协议这一组状态机的核心。状态转换图如图 2 所示。

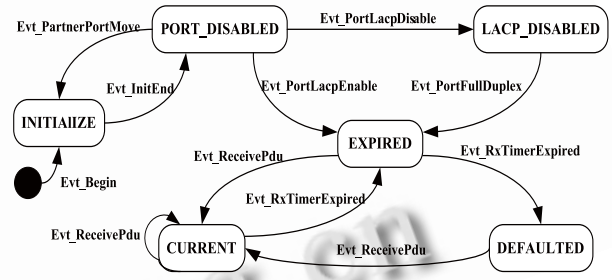

图 2 RX 状态机状态转换图

(1) INITIALIZE 状态：LACP 启动时进入这个状态，完成整个状态机数据的初始化。激励 MUX 状态机，完成后直接转换到 PORT_DISABLED 状态。

(2) PORT_DISABLED 状态：设置未同步标记，激励 MUX 状态机运行。当事件 Evt_PartnerPortMove (对端端口移除)发生时，转换到 INITIALIZE 状态。当端口 up 但是工作于半双工模式时，转换到 LACP_DISABLED 状态。若端口 up 且是全双工模式则进入 EXPIRED 状态。

(3) LACP_DISABLED 状态：设置端口为非选中，清零保存的对端信息同时将对端信息记为默认值。当事件 Evt_ProtFullDuplex(端口变成全双工)发生时，转化到 EXPIRED 状态。

(4) EXPIRED 状态：进入该状态时会启动短接收

定时器，如果在该定时器超时之前还没有收到对端发送的 LACP 报文(即事件 Evt_RxTimerExpired 发生)，就进入 DEFAULTED 状态。反之，若在此之前收到了对端的 LACPDU(即事件 Evt_ReceivePdu)，则进入 CURRENT 状态。

(5) DEFAULTED 状态：设置端口为非选中，激励 MUX 状态机，当接收到对端的协议报文时(即事件 Evt_ReceivePdu)，进入 CURRENT 状态。

(6) CURRENT 状态：解析接收到的 LACPDU，解析报文，得到当前对端的聚合相关信息，并与之前保存的对端信息相比较，如果不同，表示对端已发生了变化。此时要调用选择逻辑模块重新计算端口的选中状态。如果相同，则无需重新计算。另外还要判断接收到的对端的 LACPDU 的发送状态。如果是长周期发送，则启动长接收定时器。如果是短周期发送则启动短接收定时器。当定时器超时时(即事件 Evt_RxTimerExpired 发生)，进入 EXPIRED 状态。如果在定时器超时之前接收到 LACPDU(即事件 Evt_ReceivePdu 发生)，重新进入 CURRENT 状态。

2.2.2 TX 状态机

TX 状态机：它负责填充 LACP 协议报文的内容并处理协议报文的发送，当收到通知时，TX 状态机马上调用报文发送模块发送报文，由于 LACP 协议属于慢速协议，每秒最多允许发送 10 个 LACP 协议报文。TX 状态机要确保这个要求，所以 TX 状态机内部维护 NTT(Need To Transmit)标志和全局变量 ucSendPduNum(用来记录已发送报文的个数)，每隔 1 秒 ucSendPduNum 会被清零，处理流程见图 3。

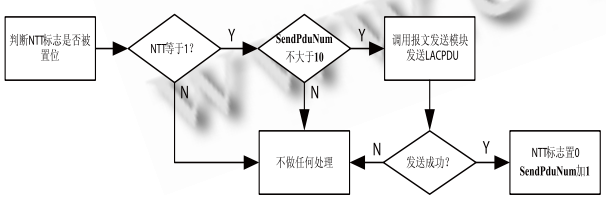

图 3　TX 状态机处理流程图

2.2.3 PTX 状态机

PTX 状态机：维护周期发送定时器的发送周期，即用来控制 Actor 和 Partner 交互 LACPDU 的频率。因为当系统处于平稳状态时，交互频率可以低一些，以减轻系统处理协议报文的负担。而当 Actor 或 Partner 有一方发生了对聚合有影响的变化时，为了让系统尽快的稳定，两端交互的频率越快越好，但是前面已提到过，LACP 协议属于慢速协议，每秒最多发送 10 个 LACP 协议报文。交互的频率不能超过此限制，所以维护快速发送和慢速发送两种定时器，并且慢速周期发送定时器的周期设置为 2500ms，快速周期发送定时器的周期设置为 150ms。当定时器超时的时候通知 TX 状态机(发送状态机)发送 LACPDU。状态转换图如图 4 所示。

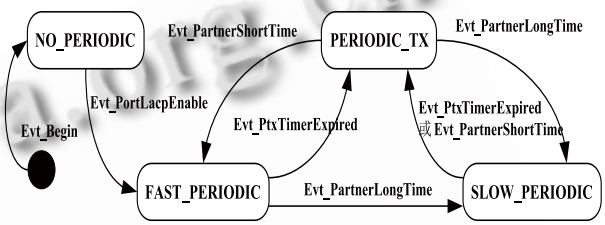

图 4　PTX 状态机状态转换图

(1) NO_PERIODIC 状态：初始化时默认进入 NO_PERIODIC 状态,此时周期发送定时器处于关闭状态。当 Evt_PortLacpEnable(端口 UP 且端口工作在全双工模式)事件发生时，进入 FAST_PERIODIC 状态。

(2) FAST_PERIODIC 状态：启动快速周期发送定时器。若在定时器超时之前 Evt_PartnerLongTime(对端的发送周期为长周期)事件发生时，进入 SLOW_PERIODIC 状态。而 Evt_PtxTimerExpired(周期定时器超时)事件发生时，进入 PERIODIC_TX 状态。

(3) SLOW_PERIODIC 状态：启动慢速周期发送定时器。若 Evt_PtxTimerExpired(周期定时器超时)事件或者 Evt_PartnerShortTime(对端的发送周期为短周期)事件发生时，进入 PERIODIC_TX 状态。

(4) PERIODIC_TX 状态：该状态只维持很短暂的时间，进入这个状态时，首先将 NTT 置为 1(即激励 TX 状态机发送 LACPDU)，随后就转换到 FAST_PERIODIC 状态或者 SLOW_PERIODIC 状态。如果 Evt_PartnerShortTime(对端的发送周期为短周期)事件发生，则进入 FAST_PERIODIC 状态，反之若 Evt_PartnerLongTime(对端的发送周期为长周期)事件发生时，进入 SLOW_PERIODIC 状态。

2.2.4 MUX 状态机

MUX 状态机是用来根据本端和对端的选中状态，决定是否阻塞端口以控制端口接收和发送数据功能的

开启和关闭。为了能够更快的响应事件和简化处理，当本端端口选中时，不采用延时处理的方式，状态直接转化，并且采取接收和发送功能捆绑控制的方式，即接收和发送只能同时开启或关闭，而不能一个功能开启，另一项关闭。状态图如图 5 所示。

图 5 MUX 状态机状态转换图

(1) DETACHED 状态：初始化时默认进入 DETACHED 状态，设置同步标志位为 FALSE，端口处于阻塞状态，不能转发数据。当 Evt_ActorSelect(本端被选中)事件发生时，进入 ATTACHED 状态。

(2) ATTACHED 状态：设置同步标志位和 NTT 标志位为 TRUE，端口仍然处于阻塞状态。此时若 Evt_ActorUnSelect(本端未被选中)事件发生，会回到 DETACHED 状态。若 Evt_PartnerSelect(对端已选中)事件发生，则进入 COLLECTING_DISTRIBUTING 状态。

(3) COLLECTING_DISTRIBUTING 状态：设置 NTT 标志位为 TRUE，解除端口的阻塞状态，此时端口才开始可以正常的转发数据流量。在 LACP 系统稳定时，MUX 状态机会一直处于这个状态。而当 Evt_ActorUnSelect(本端未被选中)或 Evt_Partner-UnSelect(对端未选中)事件发生时，系统就进入 ATTACHED 状态。

2.3 状态机的软件实现

用软件实现状态机通常有以下几种：nestted switch statement(嵌套 switch)、state table(状态表)、Function Address As State(用函数指针作为状态)、面向对象的设计方法等等[4]。嵌套 switch 方法是状态机软件实现的最直接的方法，它用一个 switch 判断处于何种状态，再用一个 switch 判断发生何种事件。这种方法的优点是代码结构简单，十分直观，便于理解。但采用这种方法，代码显得很冗余，尤其是当状态机比较复杂时更甚。所以这种方法最好当状态机比较简单时使用。

state table 方法采用二维数组的数据结构来实现状态机。状态和事件均采用枚举值，数组的行对应于状态，列对应于事件。数组的项就对应于要做的动作和下一个状态。优点是结构简单有序，效率是这几种方法中最高的。但是数组中的很多项可能为空，浪费空间。且耦合性较强，不方便扩展。

Function Address As State 方法用函数地址代替状态值，每个状态对应于一个函数指针。状态的动作和转换统一于函数指针里。它的优点是代码框架好，比较直观。效率较高。

面向对象的设计方法把 OOP 中的抽象、继承等概念用于状态机的软件实现中。但它需要对面向对象特性的支持。本文采用 C 语言来实现，但 C 语言实现面向对象的特性不够自然。所以不采用面向对象的设计方法。LACP 状态机的软件实现借鉴了上述各种方法，状态值采用枚举值，状态的动作和转换采用函数指针的方式。这样代码结构直观清晰，扩展方便。端口对应的本端和对端的聚合相关信息保存在结构体指针全局变量 pstPortInfo 指向的内存中。代码框架具体如下(以 RX 状态机为例,对于 RX 具体的每个状态的处理函数，以 RX 状态机处理 Port_Disabled 状态的函数为例，其它状态略)：

```
/* 状态机处理函数原型 */
typedef VOID (*LACP_FSM_PF) (LACP_PORT_INFO_S *pstPortInfo)；
/* RX 状态机的状态枚举 */
typedef enum tagLAGG_LacpFsmRXState {
  LACP_RX_STATE_BEGIN = 0,
  LACP_RX_STATE_CURRENT,
  LACP_RX_STATE_EXPIRED,
  LACP_RX_STATE_DEFAULT,
  LACP_RX_STATE_INIT,
  LACP_RX_STATE_LACP_DISABLED,
  LACP_RX_STATE_PORT_DISABLED
}LAGG_LACP_RXSTATE_E;
/* RX 状态机对应每个状态的处理函数 */
LACP_FSM _PF g_stLACPFsmRXPrcsFn[] = {
  LACP_FSM_RXBegin;
  LACP_FSM_RXCurrent;
  LACP_FSM_RXExpired;
  LACP_FSM_RXDefaulted;
  LACP_FSM_RXInitialize;
  LACP_FSM_RXLacpDisabled;
  LACP_FSM_RXPortDisabled;
```

```
};
/* RX 状态机的主函数，用来实现 RX状态机的功能 */
VOID LACP_FSM_RXMachine(LACP_PORT_INFO_S *pstPortInfo) {
    …; /*一些处理*/
    g_stLACPFsmRXPrcsFunc[pstPortInfo->ucRxState](pstPortInfo);
    return;
}
/* RX状态机处理 Port_Disabled状态的函数 */
VOID LACP_FSM_RXPortDisabled (LACP_PORT_INFO_S *pstPortInfo){
    if(事件 Evt_PartnerPortMove 发生) {
    …; /*进入 INITIAlIZE状态所要做的一些动作*/
    pstPortInfo->ucRxState = LACP_RX_STATE_INIT;
    LACP_FSM_RXInitialize(pstPortInfo);
  }
  if(事件 Evt_PortLacpEnable发生) {
    …; /*进入 EXPIRED状态所要做的一些动作*/
    pstPortInfo->ucRxState = LACP_RX_STATE_EXPIRED;
    LACP_FSM_RXExpired(pstPortInfo);
  }
  If(事件 Evt_PortLacpDisable 发生) {
    ……; /*与上面的结构类似*/
  }
  )
```

3 结论

把状态机模块合入某通信公司基于 Linux 的网络平台并编译成 APP 文件，加载到两台该公司二层交换机产品上，把它们配置成动态链路聚合模式，经测试，状态机模块各项功能正确，对相关事件均能快速响应。LACP 协议的链路聚合模式能够自动配置链路状态、自动响应各种相关事件、自行处理各种异常情况等。对用户而言，具有使用方便，维护简单的优点，但也导致链路聚合的控制相对复杂。本文在 802.3ad 的 LACP 协议的基础上，实现并优化了状态机模块来控制整个协议的运行。应用有限状态机的设计模式，降低了软件开发的复杂度，使得模块的结构更加清晰合理。提高了软件的可扩展性和可维护性。

参考文献

1 金星亮.基于 MSTP 的链路聚合的研究与实现[硕士学位论文].成都:电子科技大学, 2007.

2 Amendment to Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications—Aggregation of Multiple Link Segments. IEEE Std 802.3ad, 2000. 116-120.

3 Samek M. Practical Statecharts in C/C++ Quantum Programming for Embedded Systems.1th ed, USA: CMP Books, 2004. 25－26.

4 池元武.用状态机原理进行软件设计.[2009-8-20] http://www.docin.com/p-26276442.html