第 38 卷 第 4 期
Vol.38 No.4

计 算 机 工 程
Computer Engineering

2012 年 2 月
February 2012

·开发研究与设计技术· 文章编号：1000—3428(2012)04—0263—03 文献标识码：A 中图分类号：TP393.06

基于 TCL 的 6LoWPAN 协议一致性测试

杨德兴，刘钦明，魏 磊，史红周

(中国科学院计算技术研究所，北京 100190)

摘 要：针对 6LoWPAN 协议，提出一种基于 TCL 的一致性测试系统。在该系统中，界面控制部分提供用户操作和结果查看等功能，测试执行部分提供用例解释执行、结果分析等功能，底层通信部分提供物理层收发的功能。使用 TCL 脚本语言设计测试用例和扩展命令，从而增强系统的可扩展性。对 Contiki 系统中的 uIPv6 协议栈进行测试，结果表明，该测试系统的可扩展性较好，可满足 6LoWPAN 协议的一致性测试要求。

关键词：6LoWPAN 协议；802.15.4 标准；IPv6 协议；TCL 脚本语言；一致性测试；uIPv6 协议栈

Conformance Test of 6LoWPAN Protocol Based on TCL

YANG De-xing, LIU Qin-ming, WEI Lei, SHI Hong-zhou

(Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China)

【Abstract】For 6LoWPAN protocol, this paper proposes a conformance test system based on TCL. The system consists User Interface(UI) control, test execution and device communication. Through the UI control, users can interact with the system and view results. The test execution module can provide test case execution, results analysis and so on. The device communication provides physical layer transceiver function. By using TCL script language to design test cases and extend commands, the system scalability is enhanced. The Contiki uIPv6 stack is tested and the results show that this system has the quality of expansibility, and can meet the 6LoWPAN protocol conformance test requirements.

【Key words】6LoWPAN protocol; 802.15.4 standard; IPv6 protocol; TCL script language; conformance test; uIPv6 protocol stack

DOI: 10.3969/j.issn.1000-3428.2012.04.086

1 概述

为了将 IPv6 引入到低速无线个人域网(LoWAPN)中，IETF 于 2004 年 11 月成立了 6LoWPAN 工作组[1]，其主要目标是研究制定在 LoWPAN 上运行 IPv6 协议的一系列技术和标准。6LoWPAN 在底层采用 IEEE802.15.4 标准[2]的 MAC 层和物理层，在网络层使用 IPv6 协议。由于 IPv6 分组报文长度远大于 802.15.4 的载荷长度，因此在 MAC 层和网络层之间引入了适配层，使这两层实现无缝连接。

2007 年，关于 6LoWPAN 的标准(RFC 4944)正式发布[3]，该标准除了定义适配层的功能，如适配层帧格式、包头压缩、分片重组，也对 IPv6 的使用做了规定，如 MTU 值、IPv6 地址构造等。尽管有协议标准作为依托，但由于实现者理解的差异以及自然语言描述的不准确性，6LoWPAN 协议栈的实现不一定能完全符合标准。因此，本文设计并实现 6LoWPAN 协议一致性测试系统，为 6LoWPAN 协议栈之间的相互通信提供基本保障。

2 TCL 语言与 6LoWPAN 协议测试

由于 6LoWPAN 还在不断发展，如路由协议仍处于草案阶段，新的协议标准会不断出现，因此对 6LoWPAN 协议的测试要具有可扩展性，使得新的协议测试能够很容易的集成到现有系统中。TCL 作为一种可嵌入、可扩展的解释脚本语言[4]，它非常适合用于 6LoWPAN 协议的测试。

TCL 解释器的实现相当于 C/C++的函数库，可以把它嵌入到应用程序中，这样在程序中就可以执行 TCL 脚本。TCL 提供了丰富的用于扩展 TCL 命令的 C/C++函数接口，所以，可以对后续的协议，设计新的 TCL 命令，然后封装成动态链接库。测试人员在用 TCL 语言进行测试用例开发时，通过动态加载的方式加载库文件，就可以像使用 TCL 基本命令一样使用 TCL 扩展命令，完成新协议的测试。

3 测试系统结构

一致性测试[5]是一种“功能测试”，它依据一个协议的描述对协议的某个实现进行测试。一致性测试主要关心协议实现的外部行为，对于被测实现的任何结论都是通过观察和控制上测试器和下测试器的接口上发生的事件来做出的。

6LoWPAN 协议主要在 MAC 层和 IPv6 层引入适配层，使 IPv6 的数据包能够在 WPAN 网络中发送。对于 6LoWPAN 协议的功能，主要还是靠 IPv6 层的数据包来驱动，所以，对 6LoWPAN 的测试主要还是在网络层进行。通过构造具有某个特定功能的 IPv6 数据包，经过 6LoWPAN 层的封装后，发送给被测节点，然后等待被测节点的响应，即可实现测试。整个测试系统的拓扑结构如图 1 所示。

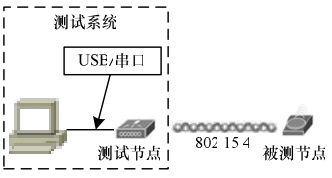

图 1 测试系统拓扑结构

基金项目：国家发改委 CNGI 专项基金资助项目“下一代互联网关键设备测试标准规范”(CNGI-09-03-02)

作者简介：杨德兴(1986－)，男，硕士研究生，主研方向：普适计算，嵌入式系统；刘钦明、魏 磊，硕士研究生；史红周，高级工程师、博士

收稿日期：2011-07-12 **E-mail：**hzshi@ict.ac.cn

测试系统软件平台如图 2 所示，主要包括 3 个部分：界面控制部分，测试执行部分，底层通信部分。界面控制用于提供测试用例编写编辑、测试用例显示、测试参数配置、测试过程控制和测试结果统计等操作；测试执行部分实现 TCL 命令扩展、用例解释执行、测试过程记录分析及与底层协议栈的控制接口等功能；底层通信部分实现 MAC 层和物理层功能，并为测试执行提供控制接口。

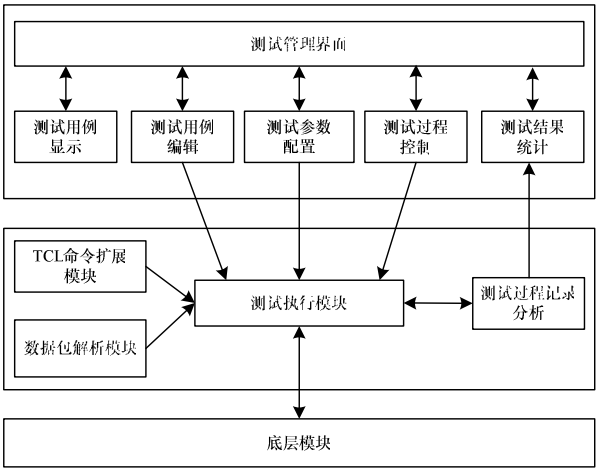

图 2 测试系统软件结构

3.1 TCL 命令扩展模块

TCL 内建的命令对协议测试支持不足，这就要求根据实际需要对命令进行扩展。该模块即完成 TCL 命令扩展的功能，测试人员根据待测协议设计实现 TCL 扩展命令，实现协议规定的功能，使其支持对协议的一致性测试。该模块实现了动态加载的机制，使得新的扩展命令可以很容易的集成到系统中，供测试脚本编写人员使用。

3.2 数据包解析模块

由于 IEEE802.15.4 和 6LoWPAN 的协议头长度都是不固定的，因此必须通过解析得到各个协议层的数据后，才能进行结果对比和分析。鉴于 6LoWPAN 协议还在不断更新和发展，解析模块在实现上要具有可扩展性，各种协议解析器之间不能够出现很强的关联性，相互之间不能够影响。所以，采用管理中心的方式来实现整个解析模块。该模块的结构如图 3 所示。

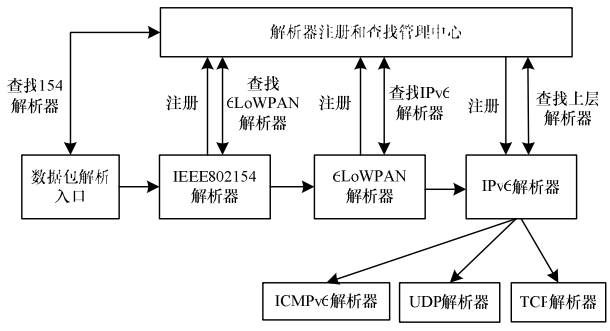

图 3 解析模块结构

各个协议的解析器需先向管理中心进行注册后才能够使用。下层协议解析完成要调用上层协议的解析器时，需要通过解析器名称在注册中心进行查找，从而得到上层协议的解析器，这样通过层层调用完成整个数据包的解析。通过这种方式，各种协议的解析器可以实现动态的注册和卸载，满足了协议测试的可扩展性。

3.3 测试执行模块

该模块实现对 TCL 解释器的封装，使其以线程的方式来运行。它通过与测试系统其他部分的交互完成对被测实现的测试，最后将测试结果输出到界面中。当用户选择了多个测试用例进行测试时，该模块一个接一个的执行各个 TCL 测试脚本，同时输出测试过程中产生的各种信息。通过对线程的控制，实现了对测试的执行、终止、跳过等功能。

3.4 底层模块

6LoWPAN 作为在 WPAN 网络中使用的协议，必须要有无线收发器把测试数据包发送给被测节点。底层模块通过收发器发送经过 6LoWPAN 封装的 IPv6 数据包，并从网络中接收完整的 IEEE802.15.4 数据包。该模块主要包括 2 个部分：协议栈和无线节点，其结构如图 4 所示。

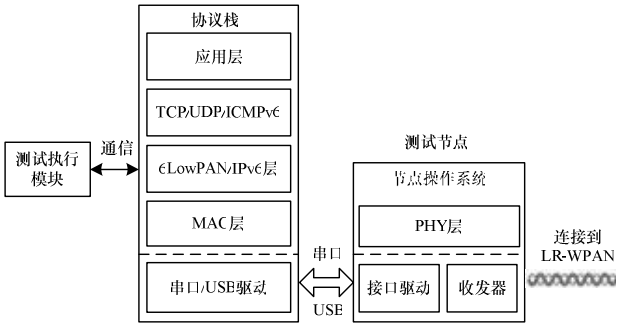

图 4 底层模块结构

协议栈主要运行 MAC 层功能，无线节点只包含 PHY 层的功能，两者在底层通过串口或 USB 连接。将 MAC 层和 PHY 进行分开，具有如下优势：

(1)测试系统可以对 MAC 层进行更细粒度的控制，可以随意设置 MAC 地址，这样就可以模拟多个不同的测试节点。

(2)可以支持多种类型和不同频段的无线节点，节点省去了 MAC 层，简化了对其的开发。

(3)对于后续有状态协议的测试，可以在协议栈进行实现，然后控制协议栈进行相关测试。

协议栈中的 MAC 层根据协议标准对数据包进行 MAC 头的封装，然后通过串口/USB 发送给无线节点，无线节点不提供对任何上层的数据处理功能，其接收和发送的数据包都是一个完整的数据帧。

4 测试用例描述

测试用例用于对协议的某部分功能进行测试，其描述分为 2 个部分：测试数据包描述和测试行为描述。整个测试用例的描述都由 TCL 内建的命令和扩展的 TCL 命令来完成。

4.1 测试数据包描述

测试数据包是一致性测试的基础，它的构造应该具有很强的灵活性，可以根据测试的需求构造任意的协议数据。测试数据包的描述主要基于 2 个扩展的 TCL 命令：header 和 packet。

(1)header 命令定义协议头中各个域的值。在 6LoWPAN 中涉及的协议头主要有 6LoWPAN、IPv6、ICMPv6。3 种协议头用 header 命令进行定义的形式分别如下：

```
header 6LoWPAN {
        Dest_Mac = …
        Src_Mac = …
        Header_Type = …
}
header ICMPv6 {
```

```
        ICMP_TYPE = …
        Code = …
        Message_Body = …
    }
    header IPv6 {
        Version = …
        Traffic_Class = …
        Flow_Label = …
        Payload_Length = …
        Next_Header = …
        Hop_Limit = …
        Source_Address = …
        Destination_Address = …
    }
```

(2)packet 命令用于描述整个数据包的组成部分，它主要是把用 header 命令定义的各种协议头组织在一起，形成一个完整的数据包。packet 命令的使用形式如下：

```
packet packetName {
        header ICMPv6 {…}
        header IPv6 {…}
        header 6LoWPAN {…}
}
```

通过使用上述的形式，即定义了数据包的结构类型，又对各个域进行了赋值，这样就可以描述任意的协议数据包。

除了使用 header 和 packet 命令外，还扩展了如表 1 所示的命令，更易于数据包的定义。

表 1　扩展命令

命令	用法	说明
default	用法与 header 命令相同	用于定义各个协议头的默认值
ipv6_address	ipv6_address varName fe80::212:7400:12e6:43ae	把 IPv6 地址转换成 16 Byte 保存到 varName 中
mac_address	mac_address varName 00:12:74:00:12:e6:43:ae	把 802.15.4 中的 Mac 地址转换成 8 Byte 保存到 varName 中
cat	[cat 60 00 00 00 \ $Src_IPv6 02 01]	把 cat 后面的字符串转为十六进制串。cat 命令后面的数据支持用“\”续行，也可通过“$”引用其他参数
@include	@inclcude /filename	在脚本文件中引入其他的脚本文件，其中，“/”表示当前程序运行路径

4.2　测试行为描述

一致性测试是一个控制、观察 IUT 的过程，测试用例必须要描述测试行为。根据 6LoWPAN 和 IPv6 协议的特性，对 TCL 命令进行了扩展，实现了一系列的测试行为，包括数据包生成、数据包发送、数据包接收和分析、延时等。一个特定功能的测试需要采用多个测试行为，这些测试行为按执行顺序组织在一起。测试系统实现的测试行为包括：

(1)数据包生成：使用 create6LoWPAN 命令来完成，主要是把用 packet 和 header 中定义的各个协议字段的值按照协议规范封装成一个完整的数据包，以便于数据包发送命令使用，其形式为：

```
create6LoWPAN    packetName
```

其中，参数 packetName 即为 packet 定义的数据包名称。

(2)数据包发送：使用 send6LoWPAN 命令来完成，主要是将 create6LoWPAN 命令封装的数据包通过底层协议栈发送给 IUT，其形式为：

```
send6LoWPAN    packetName
```

(3)数据包接收和分析：使用 receive 命令来完成，主要是从底层协议栈中接收 IUT 返回的数据包，然后进行分析，与预期的结果进行比对，得出一致性测试的结果，其形式为：

```
receive timeout flag<type:pos count bytes>…
```

其中，timeout 表示接收数据包的有效时间长度，只有在该时间段内接收的数据包才会进行结果分析；flag 表示是收到包才算测试通过(设置为 0)，还是没有收到包才算测试通过(设置为 1)，因为协议中的某些部分要求在某种情况下不能够收到数据包；<…>表示一个匹配项，其中 type 命令表示要比较的协议类型名称，其值为{wpan, 6lowpan, ipv6, icmpv6}等；pos 表示要匹配的字段的起始位置，其起始位置的计算是以 type 类型协议的数据开始处作为起点；count 表示要匹配的字节数据的个数；bytes 给出了预期的字节序列值，其形式为 0x…的十六进制串或者是以$引用的 TCL 变量。

(4)延时：使用 delay 命令来完成，用来指定 2 个测试行为之间的间隔时间，其使用形式为：

```
delay    timevalue
```

其单位是 10^2 ms。

测试 Echo Requst 和 Echo Reply 的测试行为描述如下：

```
create6LoWPAN Echo_Request
create6LoWPAN NA
send6LoWPAN Echo_Request
send6LoWPAN NA
send6LoWPAN Echo_Request
receive 40 0 <ipv6:6 1 0x3a><icmpv6:0 1 0x81>
        <icmpv6:1 1 0x00>
```

5　测试结果

根据 6LoWPAN 和 IPv6 协议的内容，系统从如下方面进行测试用例的设计：(1)6LoWPAN 报文压缩/解压缩测试；(2)6LoWPAN 报文分片/重组机制测试；(3)IPv6 地址结构测试；(4)地址自动配置测试；(5)邻居发现协议测试；(6)ICMPv6 消息处理测试。

整个系统共设计测试用例 91 个。根据完成的测试用例，测试平台对 Contiki 系统[6]进行了测试。Contiki 是一个简单的事件驱动、多任务的开源嵌入式操作系统，主要针对无线传感器网络和基于 IP 的节点开发。该系统根据 6LoWPAN 的标准，实现了 uIPv6 协议栈。测试结果表明，uIPv6 协议栈在如下 3 个方面不能够满足协议的要求：

(1)在数据包不能够到达指定目的端口时，未返回地址不可达消息。

(2)对于默认路由器的选择，当所有默认路由器的可达性不可知时，未采用循环的方式选择，而是直接选择列表中的第一个默认路由器。

(3)未实现目的地址缓存机制，因此，每次都要进行“下一跳确定”。

6　结束语

随着物联网的不断发展，对小物件、小节点进行 IP 化将是一种趋势，6LoWPAN 作为 MAC 层和 IPv6 之间的适配层协议，也将被广泛使用。本文针对 6LoWPAN 协议，介绍基于 TCL 的一致性测试系统结构和各个模块的功能，讨论编写测试用例所扩展的 TCL 命令，完成了一系列 6LoWPAN 协议的测试用例。结果表明，基于 TCL 的一致性测试系统具有方便、灵活、可扩展性好等优点，是一种有效的一致性测试技术，能够满足 6LoWPAN 协议的一致性测试要求。下一步研究工作将在现有平台的基础上，实现对 6LoWPAN 新协议的测试。

(下转第 268 页)