第35卷 第21期 计 算 机 工 程 2009年11月
Vol.35 No.21 Computer Engineering November 2009
·开发研究与设计技术· 文章编号：1000—3428(2009)21—0253—03 文献标识码：A 中图分类号：TP393

TTCN-3 测试系统协议相关部分的分析与实现

李 华[1]，张巨萍[2]，叶新铭[1]，吴承勇[1]
(1. 内蒙古大学计算机学院，呼和浩特 010021；2. 内蒙古财经学院计算机信息管理学院，呼和浩特 010050)

摘 要：在TTCN-3测试系统的实现过程中，需要划分系统中与被测相关和无关的部分。分析与被测相关部分的编解码器和TRI中与协议相关的功能函数，并讨论它们的实现方法。以DHCPv6协议为例给出相关函数的实现算法，通过测试系统各部分的协作图阐述测试系统的工作过程。

关键词：测试；协议；实现

Analysis and Implementation of Protocol Related Part of TTCN-3 Test System

LI Hua[1], ZHANG Ju-ping[2], YE Xin-ming[1], WU Cheng-yong[1]
(1. School of Computer Science, Inner Mongolia University, Hohhot 010021;
2. College of Computer Information Management, Inner Mongolia Finance and Economics, Hohhot 010050)

【Abstract】During the processing of implementation of TTCN-3 test system, there are requirements to divide the test system into protocol related part and System Under Test(SUT) unrelated part. This paper analyzes Coder/Decoder(CD) and Test Runtime Interface(TRI) functions of protocol related part. The implementation approach is discussed. As an example, DHCPv6 related algorithms are given. And the working process of the test system is interpreted by the collaboration diagram of TTCN-3 test system component.

【Key words】test; protocol; implementation

TTCN-3[1]是由ETSI提出的新一代测试语言，可被用于一致性测试、互操作测试等多种测试。TTCN-3的通用性很强，可读性好，测试系统独立于测试平台，可移植性好。用TTCN-3进行测试的文献非常多，包括无线方面、实时系统方面、移动方面[2]等。这些文献都是针对某一个被测进行测试实现，并没有深入分析TTCN-3的系统结构来划分被测相关部分和无关部分，使得针对不同的被测进行测试系统的实现时，极容易导致系统构件的重复实现，浪费人力物力。

1 TTCN-3测试系统功能结构

一个TTCN-3测试系统在概念上被认为是一组交互实体的集合，每个实体对应于一个测试系统实现的某一特殊方面的功能。这些实体管理测试执行，解释或执行编译的TTCN-3代码，实现同SUT的正确通信和外部函数并处理定时器操作。图1描述了TTCN-3测试系统的一般结构[1]，该结构主要给出了实现一个TTCN-3测试系统的多个功能块。

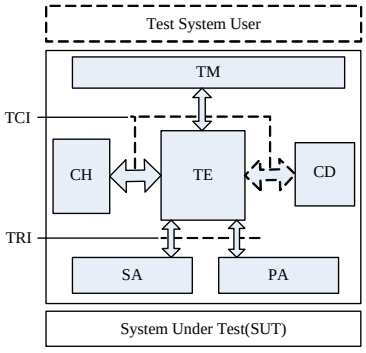

图1 TTCN-3测试系统的一般结构

各功能块的说明如下：

(1)测试管理器TM(Test Management)负责测试系统的整体管理。

(2)成分处理器CH(Component Handler)负责分布并行测试成分。这个分布可能跨越一个或多个物理系统。CH实体允许测试管理器以透明并独立于TE的方式创建并控制分布式测试系统。

(3)测试执行器TE(Test Execution)负责TTCN-3抽象测试套的解释或执行。

(4)编解码器CD(Coder/Decoder)主要用来按照特定的传输语法来编码解码测试数据。

(5)系统适配器SA(System under test Adapter)负责将TTCN-3测试系统和被测系统SUT间的通信适配到测试系统特定执行平台上。

(6)平台适配器PA(Platform Adapter)实现TTCN-3外部函数，并为TTCN-3测试系统提供统一的时间概念。

(7)TCI(Test Control Interface)是TM和TE实体间的接口。

(8)TRI(Test Runtime Interface)是TE与SUT适配器SA和平台适配器PA实体间的接口[3]。

基金项目：国家自然科学基金资助项目(60863015)；内蒙古自然科学基金资助重点项目(200711020803)；中欧联合基金资助项目“GO4IT”
作者简介：李 华(1964－)，女，教授，主研方向：分布式系统，网络计算；张巨萍，讲师；叶新铭，教授、博士生导师；吴承勇，教授
收稿日期：2009-04-30 **E-mail：**cslihua@imu.edu.cn

TM 和 CH 部分与被测内容无关，可以开发成通用实现。而对于编解码部分，虽然现在有一些成熟的编解码技术，但仍需要将现有的 TTCN-3 值类型映射为标准编码。在收到消息时，需要将接收到的网络数据包中的数据解码成接收模板的格式，并将其填充到接收模板的各个域中去，而这部分功能需要测试人员补充，不同协议对应不同的代码。另外，在测试运行时接口 TRI 中所涉及到的系统适配器 SA 的功能中，关于数据的发送和接收与测试系统的开发环境有关。例如，在基于 C 语言环境的系统中，需要利用 C 语言的 Socket 函数来进行与被测实现的通信。因此，有一些函数也需要由测试人员完成。由此可见，在 TTCN-3 测试系统的实现中，与协议相关的功能部分为编解码功能的实现和部分 TRI 函数的实现。

在 TTCN-3 测试系统中，测试还可以是并行执行的，此时，TM 负责整个系统的管理，CH 负责执行调度，对每个测试执行器 TE 来说，都需要一个自己特定的编解码器、SA 和 PA[1]。

2 测试系统分析

在 TTCN-3 测试系统的实现中，可以分为协议相关部分和协议无关部分(如 CD, SA, PA 部分)。目前有许多开源的 TTCN-3 实现，主要实现的是协议无关部分，对于协议相关部分的分析与实现讨论较少。

2.1 TTCN-3 测试系统中编解码器的工作流程

在 TTCN-3 测试系统中，编解码器 CD 是与 TE 进行交互的，需要被编码的 TTCN-3 实例化模板由 TE 传给 CD，CD 将编好码的值传回给 TE，由 TE 调用 SA 发送数据，该过程的流程如图 2 所示[4]。

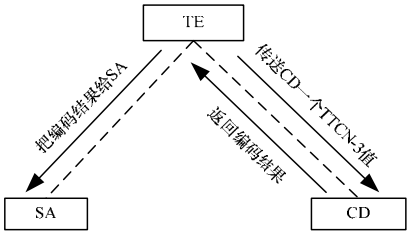

图 2 编码协作图

接收到的需要解码的数据首先由 SA 传给 TE，由 TE 交给 CD 解码，CD 将解码得到的 TTCN-3 实例化模板交给 TE，由 TE 对它进行处理，得出最后的测试判定，该过程的流程如图 3 所示[4]。

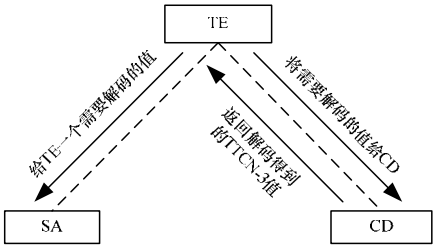

图 3 解码协作图

2.2 测试系统中编解码器的实现算法

(1)测试系统中编码部分的分析

编码器负责将 TTCN-3 值编码为可被网络传输的比特串。对于不同的 TTCN-3 值，采用不同的处理方法。对于 TTCN-3 中比较复杂的数据类型，如 record, union 等，可以采用递归的方法来进行编码。首先确定复杂类型中域的个数，然后依次对每个域的值进行编码。对于复杂类型外的其他类型，如简单类型可以直接进行编码。

编码部分的算法描述如下：

```
recursive_encode(ttcn3Value,encoded_data)
{ //判断 ttcn3Value 的类型
  {
    if (type of ttcn3Value is  复杂类型)
      {
        递归编码 ttcn3Value 的每个域的值;
      }
    if (type of ttcn3Value is  简单类型)
      {Binarystring=Change(ttcn3Value);
        Concat(encode_data, Binarystring);
      }
    if (type of ttcn3Value is bitstring)
      Concat(encode_data, ttcn3Value);
    }
}
```

算法中 ttcn3Value 为 TE 传给 CD 的要被编码的 TTCN-3 值，而 encoded_data 是 CD 返回给 TE 的编码值。算法采用递归思想实现，递归参数的分字段存取可以利用 TTCN-3 核心部分实现的功能获得复杂类型中域的个数。该算法思想已被用于 DHCPv6 协议的测试实验，可以完成对 DHCPv6 协议数据包的编码工作。其中，算法中使用到的类型转换算法(Change)可以适用于所有测试程序的编码，不需要由测试人员自己编制，可以做成通用的。

(2)测试系统中解码部分的分析

相对于编码，解码算法比较复杂，它与被测的耦合性非常大，几乎完全取决于被测。以 DHCPv6 为例说明其中的一般性以及特殊性。

在 DHCPv6 的测试实现中，解码操作的算法如下：

```
DHCP_tri_decode(encode_data, type, strategy, decoded_data)
{
  Len=length_of(encode_data);
  decoded_data=type_instan_value(type,strategy,ctx);
  读取 DHCPv6 消息包的类型值，根据类型调用相应的解码子
程序;
}
```

在上述算法中，encode_data 是由 TE 传给 CD 的需要解码的值；type 是解码类型；strategy 为空间分配策略，确定空间是临时分配还是永久分配。decoded_data 为由 CD 传回给 TE 的解码后的 TTCN-3 值。

解码子程序的算法如下：

```
DHCP_tri_sub1_decode(encode_data, type, decoded_data)
{
从 encode_data 中取能够解码出一个 TTCN-3 值的长度的数据;
//如需要解码得到一个长度为 2 的 hexstring 类型的值，则需要
//从 encode_data 中取 1 个字节的数据
对取出的数据，将它转化成合法的 TTCN-3 类型的值;
}
```

在解码子算法中，一般先将接收到的数据包以 bitstring 类型存储，然后再将它转成其他的 TTCN-3 类型。一次解码取出的数据长度与将要解码的值域大小相关。

在该算法中，数据类型的转换、数据域的填充、在 decoded_data 中取得解码后的 TTCN-3 值等操作，也可以做成通用的。

测试人员在编写特定协议的解码操作时，直接调用这些通用函数即可，从而可以降低重复开发的工作量并提高测试人员的工作效率，使得测试人员可以集中精力来开发测试例，而不需过多地考虑底层的实现。

另外，本文的编解码算法使用的方法既不是 Basic Encoding Rules(BER)也不是 Packed Encoding Rules(PER)，而是按照 RFC 要求的格式组包。在编码时，直接将发送的 TTCN-3 实例化模板通过上面的编码算法编码成比特串发送出去；在解码时，按上述算法作编码操作的逆操作。而发送给被测的数据包完全符合被测所需要的数据包格式，可以被被测识别。

3 测试系统中 TRI 的实现

在 TTCN-3 测试系统[1]中，另外一个与协议关联密切的就是 TRI[4]部分。在这一部分中，除了标准中提供的基本函数(通用)外，还需考虑协议测试所要处理的数据都是与特定协议相关的，所以，还必须在通用 TRI 之上再包装一层与协议相关的 TRI 功能函数，而这组函数是无法开发成通用形式的，它们与被测密切相关。这部分函数功能需要由用户开发实现。

以 DHCPv6 为例，其结构如图 4 所示。

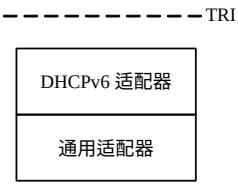

图 4　DHCPv6 的适配器结构

DHCPv6 适配器中需要实现的函数主要有 triSAReset, triSend 等，此处仅介绍前者。

triSAReset 函数在测试开始运行时由 SA 调用，负责建立与 SUT 的连接。以 C 语言为例，给出 triSAReset 的实现方法。由于 DHCPv6 协议是基于 UDP 的，因此使用数据报 Socket。

```
triSAReset() {
定义常量，如被测设备的 IP 地址、端口号等;
创建用于发送数据的 Socket，将其与对方端口连接;
创建用于接收数据的 Socket，将其与本地接收端口绑定;
启动接收线程，准备接收数据;
}
```

在一个成分端口执行一个 TTCN-3 发送操作时，TE 调用 Trisend 操作来执行真正的数据发送。在使用 C 语言实现的测试系统中，发送操作可以调用 C 的 Socket 函数来实现。

```
triSend (const TriComponentId *componentId,
        const TriPortId *tsiPortId,
        const TriAddress *sutAddress,
        const TriMessage *sendMessage)
{
//调用 C 语言中用于 UDP 的发送操作即可，但此处的参数
//sendMessage 必须是已编好码的数据
}
```

4 测试系统的实现示例

DHCPv6 测试系统遵循 TTCN-3 测试系统的基本结构，主要由 TTCN-3 描述的测试例的编辑部分、将 TTCN-3 描述的测试例编译映射为 C 语言的部分、编解码部分、适配器(SA, PA)部分以及 C 的集成系统组成。各个功能单元的协同工作情况如图 5 所示。

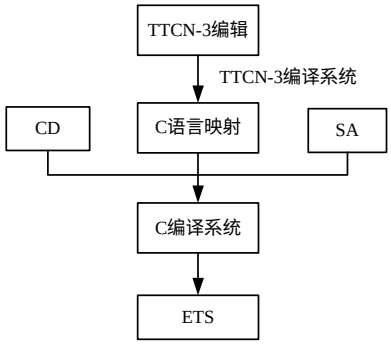

图 5　TTCN-3 测试系统各部分协作图

为完成各个功能单元的协同工作，首先需要在 TTCN-3 的编辑环境中对 TTCN-3 描述的测试套进行编辑，由 TTCN-3 的编译系统负责检查语法错误，确认没有语法错误后，将该 TTCN-3 测试例映射为 Java 或 C 语言描述的相应程序。再针对特定协议编写编解码器和 SA 中的 TRI 函数，保证通信的正确进行。然后将 CD, SA 及编译好的 TTCN-3 测试套在 C 语言环境中进行联编后，得到可执行测试套(Executable Test Suite, ETS)。将这个可执行测试套与具体协议(如 DHCPv6)的被测通信进行测试，根据预期的结果进行测试判定[4]。

5 结束语

综上所述，笔者划分了 TTCN-3 测试系统中的被测相关和无关部分，并对相关部分进行了分析，提取出了与被测相关的编解码部分和 TRI 中的部分函数，并在现有的开源代码的基础上开发了基于 TTCN-3 的 DHCPv6 协议测试系统。通过测试实践，该系统可以在纯 IPv6 环境中进行 DHCPv6 一致性测试和互操作测试。

参考文献

[1] ETSI. ETSI ES 201 873-1 V3.2.1-2007 Methods for Testing and Specification(MTS)——The Testing and Test Control Notation Version 3, Part1: TTCN-3 Core Language[S]. 2007.

[2] 王　玮, 黄小红, 常　婧, 等. 基于 TTCN-3 的移动 IPv6 并行测试的设计与实现[J]. 厦门大学学报: 自然科学版, 2007, 46(11): 14-17.

[3] ETSI. ETSI ES 201 873-5 V3.2.1-2007 Methods for Testing and Specification(MTS)——The Testing and Test Control Notation Version 3, Part 5: TTCN-3 Runtime Interface(TRI)[S]. 2007.

[4] 张巨萍. 基于 TTCN-3 的 DHCPv6 协议互操作性测试研究[D]. 呼和浩特: 内蒙古大学, 2008.

编辑　顾逸斐