IPv6邻居发现协议的一致性测试*)

叶新铭　孙美飞
（内蒙古大学计算机学院　呼和浩特010021）

摘　要　Neighbor Discovery(邻居发现,ND)协议是下一代互联网协议IPv6协议中的一个重要组成部分。本文简要介绍了邻居发现协议,提出了一种基于有限状态机和消息序列图相结合的协议一致性测试的形式化方法,实现了邻居发现协议一致性测试集的形式化,给出了用测试例描述语言TTCN描述的测试例的实现,并对Windows2000下的IPv6邻居发现协议进行了一致性测试,给出了测试报告。

关键词　IPv6,邻居发现协议,一致性测试,形式化描述方法

The Conformance Testing of Neighbor Discovery for IPv6

YE Xin-Ming　SUN Mei-Fei
(Department of Computer Science,NeiMongol University,Hohhot 010021)

Abstract　Neighbor Discovery for IPv6 is a main part of IPv6 protocols. This paper introduces the Neighbor Discovery protocol simply and presents a formal method of the protocol conformance testing which based on FSM and MSC. A formal Neighbor Discovery protocol test suit is specified,and the implementation of the test case based on TTCN is given out. Also we test the conformance of the Neighbor Discovery for IPv6 on Windows2000 and give out the test report.

Keywords　IPv6,Neighbor discovery protocol,Conformance test,Formal description method

1　引言

在计算机网络的发展历程中,协议一直处于核心地位。从ARPAnet发展到Internet到今天的IPv6,其关键环节就是形成了国际标准化的协议。目前,IPv6网络协议大多用自然语言描述,实现者对协议文本的不同理解以及实现过程中的非形式化因素都会导致不同的协议实现,在这种情况下就需要一种有效的方法对协议实现进行评价,这就是"协议测试"[1,2,4]。协议测试有多种类型[10],即:一致性测试、互操作性测试和性能测试等。协议的一致性测试主要是验证网络产品的协议实现的准确性,判断网络产品的协议实现是否符合协议的国际标准,以保证协议的各种版本之间能够互通并进行可靠的通信。因此,一致性测试是协议测试的最基本内容,是其它测试的基础。

一致性测试的基本思想是向被测实现IUT(Implementation Under Test)发送一些测试信息,然后将该IUT返回的信息与相应的标准相对比,如果两者不一致,则该IUT未能通过一致性测试。在协议一致性测试过程中,首要的问题是针对协议的文本说明给出形式化的描述,即将协议描述抽象为具有严密逻辑结构的形式描述语言,也称之为"建模"。由于在分析协议性能、测试协议的实现过程中将不断的用到模型,所以模型的建立过程在一致性测试中占重要地位,模型的好坏将直接影响后期的测试工作。依据IPv6邻居发现协议的特性,我们采用有限状态机和消息序列图对其进行建模,采用国际上通用的测试套描述语言TTCN[3](Tree and Tabular Combined Notation)实现了邻居发现协议的实际测试。

本文第2节对IPv6协议族的邻居发现协议进行了简要介绍,包括邻居发现协议的产生、邻居发现协议机制和邻居发现协议消息类型;第3节对邻居发现协议的形式化描述进行了详细介绍,包括邻居发现协议的有限状态机模型和消息序列图;第4节用TTCN实现了对邻居发现协议的测试;最后总结全文。

2　IPv6邻居发现协议介绍

2.1　邻居发现协议机制

邻居发现协议[5,6]是IPv6协议簇的一个基本的组成部分,RFC2461[9]发布了协议的标准文本,它是由因特网工程任务组(IETF)在1998年12月制定的,它实现了在IPv4中的地址解析协议(ARP)、控制报文协议(ICMP)中的路由器发现部分、重定向协议的所有功能,并进行了一定的扩展。邻居发现协议解决了连接在同一条链路上的所有节点之间的互操作问题。具体来说,邻居发现协议定义了解决如下一些问题的机制:路由器发现、前缀发现、参数发现、地址自动配置、地址解析和重定向功能。

2.2　邻居发现消息类型

邻居发现协议使用五种类型的IPv6控制信息报文(ICMPv6)来实现邻居发现协议的各种功能,这五种信息报文是:路由器请求(Router Solicitation)、路由器宣告(Router Advertisement)、邻居请求(Neighbor Solicitation)、邻居宣告(Neighbor Advertisement)和重定向(Redirect)。

3　邻居发现协议的形式化描述

3.1　FSM和MSC简介

为了对IPv6的邻居发现协议进行形式化描述,我们根据此协议中邻居发现的实现机制采用FSM和MSC对其进行建模,首先给出FSM和MSC的定义:

*)基金项目:国家自然科学基金项目(69863001)和内蒙重点领域项目(ZL9902)。叶新铭　教授,博士生导师,主要研究方向为网络协议测试。孙美飞　硕士研究生,主要研究方向为协议一致性测试。

(1)有限状态机

定义1 有限状态机 M 是一个五元式：

$M=\{I,O,S,\delta,\lambda\}$

其中 I、O 和 S 分别是输入字符、输出字符和状态的有限非空集合；δ 是从 $S\times I$ 至 S 的单值部分映射；λ 是从 $S\times I$ 至 O 的单值部分映射。

当有限状态机处于 $s(\in S)$ 状态并收到输入 $a(\in I)$ 时，它就转换到由 $\delta(s,a)$ 说明的下一个状态，并产生由 $\lambda(s,a)$ 说明的一个输出。

有限状态机 FSM 又可以用有向图 $G=(S,E)$ 来表示。其中，顶点集 $S=\{S_1,S_2,\cdots,S_n\}$ 对应着 FSM 中规定的状态。在边集 E 中，从 S_i 到 S_j 的一条有向边对应着 FSM 中从状态 S_i 到状态 S_j 的一个状态转换。图 G 中的每一边都有一输入/输出对作为记号。我们用 $(S_i,S_j;I_k/O_m)$ 表示从 S_i 到 S_j 的一条具有标号 I_k/O_m 的边，它说明当有限状态机 FSM 在状态 S_i 时，接收输入 I_k，将产生输出 O_m，并转换到状态 S_j。

(2)消息序列表 MSC 简介 MSC(Message Sequence Chart)是 IUT-T 提供的一种较新的形式描述语言，它是一种标准迹语言。与传统的形式描述语言相比，它着眼于并发进程之间传递的消息，通过描述消息交换来说明系统成分与环境之间的通信行为。它以直观、透明的方式反映通信行为，省略了传统形式描述技术必须考虑的大多数诸如程序变量以及变量值这样的细节，给测试带来了极大的方便。MSC 有两种表示方法：文本表示方法和图形表示方法。其中，图形方式较直观地反映出消息传递，易于学习、使用和解释；而文本方式则以严格的语法规则和操作语义定义形式模型，适用于进行理论推导和分析。

定义2 一个消息序列表(MSC)M 是一个六元组：

$M=\langle V,<,L,T,m,P\rangle$，

其中：V 是一个事件集合；$<\subseteq V\times V$；P 是一个进程集合；L：$V\rightarrow P$ 是将每个事件与一个进程相关联的映射；T：$V\rightarrow\{s,r,l\}$ 是一个描述事件类型(分别为 send，receive，或 local)的映射；$m\subset V\times V$ 是发送和接收事件之间的一个关系。

3.2 邻居发现协议的 FSM 模型

这里我们需要指出的并不是对邻居发现整个协议进行分析，因为整个邻居发现协议的测试点很多，并不是都可以用形式化的方法描述的，我们只是针对事件触发的状态转换部分进行分析，从控制流和数据流的角度对其进行形式化，抽象出测试目的并通过协议测试描述语言 TTCN 实现这一部分的全部测试例，最后我们对 Windows2000 + SP3上 IPv6协议的实现进行了实际的一致性测试，发现了一些不一致性并给出了测试报告。对邻居发现协议的这部分内容，我们从协议的形式化到实际的测试给出了一套完整的过程。

在邻居发现协议的邻居不可达检测机制中，协议对节点间的可达情况定义了五种状态——未完成状态(INCOMPLETE)、失效状态(STALE)、可达状态(REACHABLE)、延迟状态(DELAY)和探测状态(PROBE)。节点会保存其与邻居间的可达情况，将可达情况的条目保存在邻居缓存(Neighbor Cache)中，并将根据所接收到的邻居宣告消息动态修改邻居缓存的内容。下面，我们将根据协议7.3部分的描述，使用有限状态机来对该部分说明进行建模，为了便于建模和分析，我们增加了一个假设的初始状态 INIT。并且为了图形表示的清晰和提高可读性，我们把引起相同起始和终结状态的事件合并成一个事件子集，把相应的输出也合并成一个事件子集。节点间的状态转换关系如图1所示，图中的两条虚线弧是我们为了采用广度优先遍历算法而添加的，在协议的标准说明文本中，这两个变迁不是通过邻居宣告消息转换的，所以我们用虚线表示。

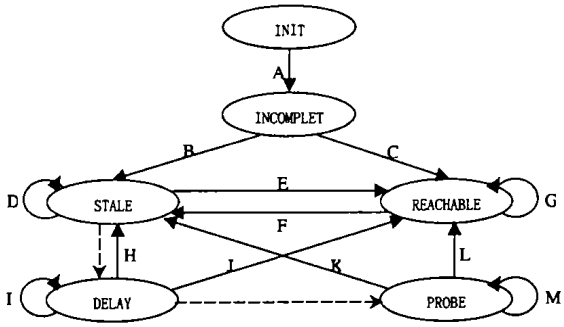

图1 邻居发现协议 NA 消息引起的状态转换关系图

根据协议的标准说明文本得出有限状态机以后，我们就可以根据有限状态机模型从控制流的角度出发生成测试目的。我们的总体测试目的就是要测试被测实现的状态之间的所有转换，也就是要遍历有限状态机的所有的边，其中，每一条边对应着一个测试目的的子集，因为一条边对应着一对状态转换，而每一对状态转换可能是多个事件都可以引起，引起每一对状态转换的任何一个事件就是一条测试目的，所以我们说每一条边对应着一个测试目的的子集。通过下面的广度优先遍历算法遍历有限状态机对应的有向图，我们就得到了所有的测试目的的子集，然后我们再根据协议标准说明文本进一步对每个测试目的的子集进行细化，这样我们就得到了邻居发现协议中的邻居不可达检测机制的所有测试目的。

由于我们要测试状态 S_i 到状态 S_j 的状态转换是否和说明一致，首先，必须使被测实现 IUT 处于状态 S_i，这就要保证从状态 S_0 到状态 S_i 必须正确，确保被测实现 IUT 是处于状态 S_i，并不是用 UIO 方法去验证，而是用已经测试过的状态转换的正确性去保证。这样我们遍历图2中边的集合的算法就采用广度优先遍历算法，按层次去遍历有向图，具体算法如算法1所示，然后按照遍历的顺序去完成邻居发现协议的一致性测试。

算法1 广度优先遍历算法

```
function BFS( AdjMatrix ):void
begin
    InitQueue(Q);//初始化队列
    En_Queue(Q,v);//顶点 v 入对
    visite[v]=1; //访问标志
    while(!Empty_Queue(Q))
    begin
        w=Del_Queue(Q);//取出对头元素,并删除
        visite(w);//对顶点 w 访问,访问函数里记录边的访问顺序
        for(u=0; u<n; u++)
        begin
            if(AdjMatrix[w][u]==1 && visited[u]==0) then
            begin
                En_Queue(Q,u);
                visited[u]=1
            end
        end
    end
end
```

算法1按广度优先遍历每条边，即按照广度优先搜索顶点，然后遍历此顶点扇出的边。也就是验证此状态到与它相邻的其它状态的转换。由于我们对图1做了这样的简化：我们把引起相同起始和终结状态的事件合并成一个事件子集，把相应的输出也合并成一个事件子集。上面得出的只是每一对状态转换的测试目的的子集，为了进一步得到更详细的测试目的，我们需要细化每一条边，也就对每一对转换进行详细的描述，根据协议标准说明文本抽象出引起状态转换的所有的事件，每一个引起状态转换的事件就是一条测试目的。由于邻居

宣告消息中对可达状态有影响的字段有以下三个：Solicited flag、Override flag 和 LLA(Link Layer Address)。下面我们对这三个字段分别作一下解释：

·Solicited flag：如果 S 位置1，表明邻居宣告消息是对邻居请求消息做出的响应；

·Override flag：如果 O 位置1，表明要更新邻居缓存中的 Link Layer Address；

·LLA：选项字段，由此字段可以看出当前的 LLA 和邻居缓存中以前的 LLA 是否一致。

这样，组合这三个字段就会得出八种不同的邻居宣告消息，每个状态收到这八种不同的邻居宣告消息都会做出相应的响应，由于 INCOMPLETE 状态的邻居缓存中没有 LLA 信息，所以它收到任何邻居宣告消息后 LLA 都要更新，只需要四种邻居宣告消息就可以了，据此我们抽象出36条测试目的。

以图1中的边 E 为例，我们对其引起状态转换的其中的一个事件给出详细的解释：边 E 从 STALE 状态指向 REACHABLE 状态，表明被测实现的邻居缓存中对应测试节点的缓存条目当前的可达状态处于 STALE 状态，当被测实现收到邻居宣告消息($S=1$,$O=1$,LLA not equal to cached)后，相应的缓存条目的可达状态转换为 REACHABLE，并且更新 LLA。这就是我们以有限状态机模型作为控制流导向，并根据协议标准说明文本得到的具体的测试目的。我们按状态进行分类，对每一个状态收到邻居宣告消息后的状态转换进行了详细描述，在此我们只给出 REACHABLE 状态的转换关系。

·REACHABLE 状态收到邻居宣告消息后的状态转换

Solicited flag	Override flag	LLA equal to cached	New State	Update LLA
clear	clear	no	STALE	no
clear	clear	yes	REACHABLE	
set	clear	no	STALE	no
set	clear	yes	REACHABLE	
clear	set	no	STALE	yes
clear	set	yes	REACHABLE	
set	set	no	REACHABLE	yes
set	set	yes	REACHABLE	

测试目的产生之后，我们就要对其进行具体的测试，首先我们要生成测试例，在测试例的设计过程中，就我们的测试目的而言，我们仍然借助于有限状态机对应的有向图2的表示，对其进行简单的优化。一个测试例，即一条测试目的对应着一对状态转换，若从状态 S_i 转换到状态 S_j 有这样的转换，它具有输入/输出对 I_k/O_m，那么这个测试例包括以下三个步骤：

1)使被测实现 IUT 处于状态 S_i；

2)输入 I_k，检查输出是否为 O_m；

3)检查此时被测实现 IUT 所处的状态是否为 S_j。

在步骤1中，为了使被测实现处于状态 S_i，我们可以使用 Dijkstra 算法求出从初始状态 S_0 到状态 S_i 的最短路径，采用有向图作为数据结构，具体算法如算法2所示。最短路径上的每边的输入部分，使被测实现 IUT 处于状态 S_i。在步骤3中，可以利用 UIO 方法来确认 FSM 是否处于状态 S_j，但在我们的实际测试中，有时候可以通过观察输出(即被测实现的响应)，来确定当前的状态。

图1用有向图 G 表示，如图2所示，顶点的对应关系如下：INIT↔S_0；INCOMPLETE↔S_1；STALE↔S_2；REACHABLE↔S_3；DELAY↔S_4；PROBE↔S_5。使用带权的邻接矩阵表示有向图 G，任意两点之间有连线的权为1，否则为∞。

算法2 顶点 S_0 到其余各顶点的最短路径

(1)初始化 dist[i]=adjmatrix[v,i],mark[i]=0，mark[v]=1,path[i]=v；

(2)从 dist 中找一个最小的，mark 为0的 dist[k]，令 mark[k]=1；

(3)修改 k 的所有 mark 为0的后继 w 的 dist[w]，若 dist[w]> dist[k]+adimatrix[k,w]，则 dist[w]=dist[k]+adjmatrix[k,w],path[w]=k；

(4)重复(2)、(3)$n-1$次。

其中，dist[i]是顶点 S_0 到 S_i 的距离；adjmatrix[i,j]是顶点 S_i 到顶点 S_j 的边上的权；path[]记录经过的路径。

$$G = \begin{array}{c} \\ S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \end{array} \begin{array}{c} \begin{array}{cccccc} S_0 & S_1 & S_2 & S_3 & S_4 & S_5 \end{array} \\ \begin{bmatrix} \infty & 1 & \infty & \infty & \infty & \infty \\ \infty & \infty & 1 & 1 & \infty & \infty \\ \infty & \infty & 1 & 1 & 1 & \infty \\ \infty & \infty & 1 & 1 & \infty & \infty \\ \infty & \infty & 1 & 1 & 1 & 1 \\ \infty & \infty & 1 & 1 & \infty & 1 \end{bmatrix} \end{array}$$

图2 FSM 的有向图 G 表示

3.3 邻居发现协议的 MSC 模型

用有限状态机对 IPv6邻居发现协议的邻居不可达检测机制进行建模以后，我们只是得到了协议测试目的的控制流模型，根据此模型得出状态转换关系，以及借助于有向图而得到其边集的遍历顺序，即测试目的的实际测试顺序。但是，这并不能很清晰地表述每一个测试例的测试过程，在用有限状态机形式化的基础上，我们采用一种较新的形式描述语言——MSC，进一步对每一对状态转换进行形式化的建模，这样我们就得到了测试目的的数据流模型。它以直观、透明的方式反映通信行为，省略了传统形式描述技术必须考虑的大多数诸如程序变量以及变量值这样的细节，给测试带来了极大的方便。MSC 图形方式较直观地反映出消息传递，易于学习、使用和解释，因此我们采用图形方式对状态转换过程进行描述。以图1中的 INIT→INCOMPLETE→REACHABLE 为例来说明如何使用 MSC 来形式化测试过程。在这个路径中的状态和事件序列为：

1)初始状态 S_0；

2)接到未知节点的包，在邻居缓存中创建新的条目，设置可达状态为 INCOMPLETE，并发送邻居请求包；

3)收到邻居宣告包，其中的 Solicited 和 Override 标志位均为1，此时更改邻居缓存中相应条目的可达状态为 REACHABLE。

这个测试例的整体流程如图3所示，首先 TN(测试节点)和被测节点(NUT)处于非邻居关系，测试节点发送 Echo Request 消息，启动定时器，此时被测节点可能的响应有两类：一类是被测节点不做任何响应，定时器超时，图3中的 exc 表示了这种例外；另一类是被测节点做出响应。但此时的响应也不唯一，图中的 alt 表示了选择关系：Others 表示收到了响应，但是响应并不是和我们的约束相匹配；引用 msc ND1.1表示

收到正确的响应。我们的测试过程还有好多步骤要做，为了表达清晰，使用 MSC 子图4将引用 msc ND1.1进一步细化。

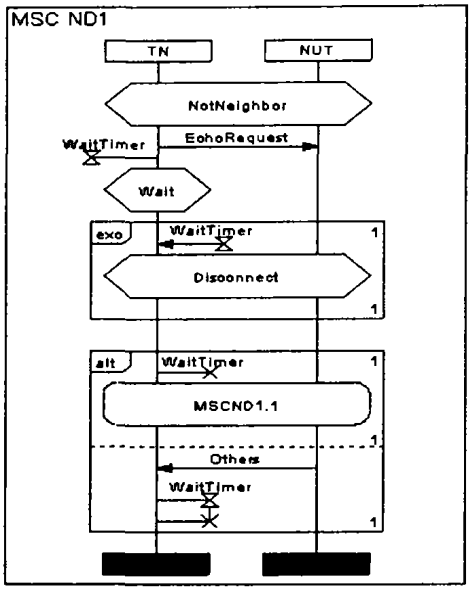

图3　邻居发现协议 MSC 模型举例

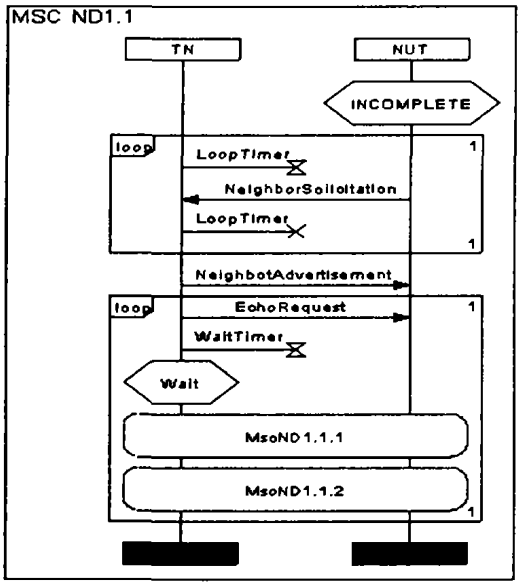

图4　邻居发现协议 MSC 模型举例

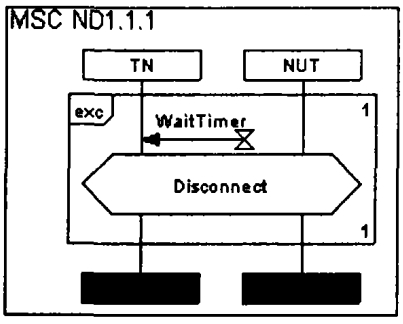

图5　邻居发现协议 MSC 模型举例

图4是图3中引用 msc ND1.1的细化，对应此测试例的事件序列(2)，被测节点收到未知节点的包，在邻居缓存中创建新的条目，设置可达状态为 INCOMPLETE，并发送邻居请求包，Loop 表示被测节点循环发送邻居请求包。测试节点收到邻居请求消息之后，发送邻居宣告消息($S=1$,$O=1$)，被测节点收到此消息后，其邻居缓存中对应条目的可达状态应该转换到 REACHABLE，我们用发送 Echo Request 消息来验证，循环发送3次 Echo Request 消息，如果每次都能收到相应的 Echo Reply 消息，说明邻居缓存中对应条目的可达状态应该转换到 REACHABLE，测试例执行成功。

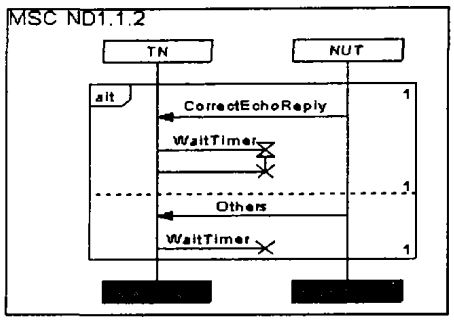

图6　邻居发现协议 MSC 模型举例

4　邻居发现协议一致性测试的实现

4.1　基于 TTCN 的测试例的实现

我们仍以图1中的 INIT→INCOMPLETE→REACHABLE 为例，来说明如何使用 TTCN 来描述具体的测试过程。借助于上一节的 MSC 模型，我们首先给出此测试例的测试拓扑图，如图7所示。测试节点(TN)用我们已经实现的测试执行系统[7]模拟路由器，被测试节点(NUT)是支持 IPv6协议的主机或路由器。

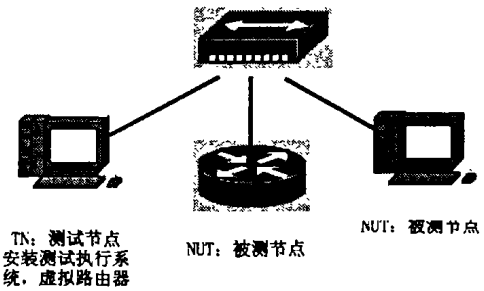

图7　邻居发现协议一个实例的测试拓扑

我们采用 TTCN 的 GR 格式来描述测试例，采用 Telelogic Tau 4.0作为测试例的编辑工具。Telelogic Tau 是 Telelogic 公司开发的用于测试的一个工具，其中包括了一个强有力的 TTCN 编辑器和语法分析器。因此不仅能编辑 GR 格式的 TTCN 测试用例，还能够检查测试例中的 TTCN 语法是否正确，同时可以将 GR 格式的测试例转换成 MP 格式。测试例的动态部分如表1所示。

下面我们对测试过程中用到的数据报进行一下简单的说明：EchoRequestPacketTN 是测试节点发出的 Echo Request 消息；EchoReplyPacketNUT 是被测试节点针对测试节点发出的 EchoRequestPacketTN 回应的 Echo Reply 消息；NeighborSolPacketNUT 是测试节点发出的 Neighbor Solicition 消息。

NeighborAdvPacketA～NeighborAdvPacketH 的定义如表2所示。

4.2　邻居发现协议的一致性测试报告

首先我们介绍一下我们的测试平台以及测试环境：

·测试例编辑工具：Telelogic Tau 4.0

·测试执行系统：测试执行器和测试驱动程序

(下转第68页)

它划分一个大的集合，可以节省删除的开销，还可以减小树结构撤销证书方法发布和验证的通信开销和计算开销。

参考文献

1 Naor M, Nissim K. Certificate Revocation and Certificate Update. In: Proc. of the 7th USENIX Security Symposium San Antonio, Texas, Jan. 1998. 26～29. http://www.usenix.org/publications/library/proceedings/sec98/full_papers/nissim/nissim.pdf

2 Housley R, Ford W, Polk W. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. April 2002. http://www.ietf.org/rfc/rfc3280.txt

3 Berkovits S, Herzog J C. A Comparison of Certificate Validation Methods for Use in a Web Environment: [MITRE Ttechnical Report]. Nov. 1998. 12～13 http://citeseer.nj.nec.com/460528.html

4 Merkle R C. A Certified Digital Signature. In: Proc. Crypto '89, Lecture Notes in Computer Science 435, Springer Verlag, 1989. 234～246

5 Kocher P C. On Certificate Revocation and Validation. FC'98, LNCS 1465, 1998. 172～177

6 Rivest R. The MD4 Message Digest Algorithm. MIT Lab for computer science Internet RFC 1186 Oct. 1990 http://www.faqs.org/rfcs/rfc1186.html

7 NIST PKI Project Team. Minimum Interoperability Specification for PKI Components, Version2 - Second DRAFT, Aug. 31, 2000. http://csrc.nist.gov/pki/documents/MISPC2_public3_20000831.pdf

8 Arens A. Public Key Certificate Revocation Schemes. Feb. 2000. 14 http://citeseer.nj.nec.com/arnes00public.html

（上接第46页）

表1 测试例的动态部分

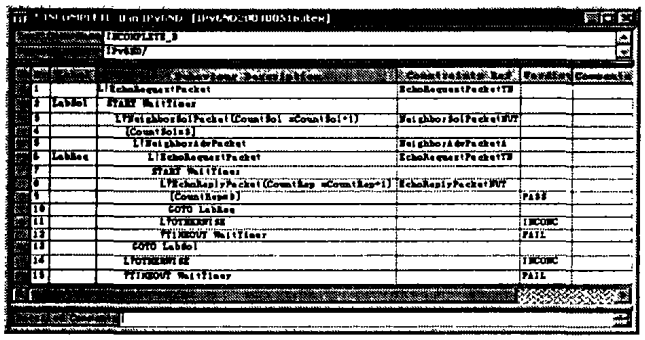

表2 邻居宣告消息定义

Solicited flag	Override flag	LLA equal to cache	Neighbor Advertisement
set	set	no	A
set	set	yes	B
set	clear	no	C
set	clear	yes	D
clear	set	no	E
clear	set	yes	F
clear	clear	no	G
clear	clear	yes	H

·网络侦包工具：Sniffer4.60.01

·被测试系统：Windows2000 + SP3 + tpipv6-001205-SP3-IE6.zip

·测试拓扑图：如图3所示

下面是我们对 Windows2000 + SP3上 IPv6实现的邻居发现协议的节点间状态转换关系的一致性测试报告，限于篇幅，我们只对在测试过程中发现的一些不一致性给出说明，如表3所示。

表3 邻居发现协议不一致性测试报告

Test Case Name	NA	Old State	New State (RFC)	New State (实测)	Update LLA (RFC)	Update LLA (实测)
NC_ReachC	C	REACHABLE	STALE	REACHABLE	NO	NO
NC_DelayC	C	DELAY	DELAY	REACHABLE	NO	YES
NC_DelayG	G	DELAY	DELAY	DELAY	NO	YES
NC_ProbeC	C	PROBE	PROBE	REACHABLE	NO	YES
NC_ProbeG	G	PROBE	PROBE	PROBE	NO	YES

结束语 文中给出了一种验证IPv6邻居发现协议一致性的形式化方法，即用 FSM 和 MSC 相互结合的方法为协议建模，然后用测试例描述语言(TTCN)对测试例进行具体的描述。有了用 TTCN 描述的测试例以后，我们就可以在自己的协议测试执行系统上对测试例进行解释执行，对协议实现进行一致性测试。实践中，我们用自己的协议测试执行系统，对 Windows + SP3上 IPv6邻居发现协议实现的一致性进行了测试。目前，基于本文提出的建模方法生成测试例的过程中，需要手工干预的内容较多，进一步的工作是提高此测试例生成模型的自动化生成程度以及进行优化以减少测试例的数目。

参考文献

1 ISO/IEC 9646-1. Open Systems Interconnection-Conformance testing methodology and framework - Part 1: General Concepts, 1994

2 ISO/IEC 9646-2. Open Systems Interconnection - Conformance testing methodology and framework - Part 2: Abstract Test Suite Specification, July 1988

3 ISO/IEC 9646-3. Open Systems Interconnection-Conformance testing methodology and framework - Part 3: The Tree and Tabular Combined Notation (TTCN), 1994

4 ISO/IEC 9646-4. Open Systems Interconnection - Conformance testing methodology and framework - Part 4: Test Realization, June, 1988

5 Deering S, Hinden R. RFC 2460 Internet Protocol, Version 6 (IPv6) Specification. S. Deering, R. Hinden. Dec. 1998

6 Narten T, Nordmark E, Simpson W. RFC 2461 Neighbor Discovery for IP Version 6(IPv6). Dec. 1998

7 叶新铭，王红霞，石立新. 下一代网络协议一致性测试执行系统的实现. 计算机科学，2003，30(4)：51～54

8 田军，张玉军，余东，等. 邻居发现协议的形式化测试. 计算机研究与发展，2001，38(12)

9 毕军，史美林. 计算机网络协议测试及其发展. 电信科学，1996，12：51～54