IPv6网络中DNS蠕虫的研究

徐延贵[1,2] 钱焕延[1] 张 凯[1]
(南京理工大学计算机科学与技术学院 南京 210094)[1] (沈阳炮兵学院自行火炮系 沈阳 110162)[2]

摘 要 基于IPv6网络环境分析了网络蠕虫的扫描策略,构建了一种新型网络蠕虫——DNSWorm-V6,该蠕虫应用两层不同的扫描策略,即在本地应用子网内扫描策略,在子网间应用DNS扫描策略。由此两层扫描策略,提出一种双层蠕虫传播模型TLM。仿真实验结果表明,DNSWorm-V6是一种可以在IPv6网络中大范围快速传播的蠕虫。可以预测IPv6网络中新型蠕虫可能带来的威胁。

关键词 IPv6,网络蠕虫,DNS,蠕虫扫描策略,蠕虫传播模型
中图法分类号 TP393 **文献标识码** A

Research of DNS Worm in IPv6 Networks

XU Yan-gui[1,2] QIAN Huan-yan[1] ZHANG Kai[1]
(School of Computer Science and Technology, Nanjing University of Science & Technology, Nanjing 210094, China)[1]
(Department of Self-propelled Gun, Shenyang Artillery Academy, Shenyang 110162, China)[2]

Abstract In IPv6 network environment, the scanning strategy of Internet worm was analyzed. A new type of worm, DNSWorm-V6 was built. The worm applies two different layers scanning strategy. It means that the worm applies subnet scanning strategy in local subnet and applies DNS scanning strategy in inter-subnet. Based on the two layers scanning strategy, a Two-Level worm propagation model, TLM was presented. The results of simulation experiment show that DNSWorm-V6 is a worm that can propagate fastly in the large-scale in IPv6 network, at the same time we can predict the threat probably posed by the new worm in IPv6 network.

Keywords IPv6, Internet worm, DNS, Worm scanning strategy, Worm propagation model

随着互联网应用的深入,网络安全问题日益严重,其中网络蠕虫危害严重、攻击范围大、爆发速度快,已经成为目前互联网所面临的最为严重的安全威胁之一。多样化的传播途径和复杂度应用环境使网络蠕虫的发生频率增高,潜伏性变强,覆盖面更广,造成的损失也更大。近几年来爆发的CodeRed[1],Slammer,Witty,WS32.Blaster等蠕虫,给整个互联网造成了巨大的损失。如何有效地遏制蠕虫在网络中的传播已经成为迫切需要解决的问题。

从网络行为的角度,蠕虫传播的研究主要集中在扫描策略和传播模型两方面。蠕虫利用系统漏洞进行传播之前,首先需要进行目标探测,良好的扫描策略能够加快蠕虫的传播。加利福尼亚大学伯克利分校的Weaver等对网络蠕虫的快速扫描策略进行了分析研究,并实现了Warhol试验蠕虫,理论推测该蠕虫能在30分钟内感染整个互联网[2]。在IPv4互联网中造成巨大破坏的蠕虫大都采用基于网络层的随机扫描策略,如CodeRed,Slapper和Slammer。所谓随机扫描是指感染蠕虫的计算机随机地选择网上计算机进行扫描,所以扫描的目标为IPv4所有地址空间,即2^{32}。其它扫描策略还包括基于目标列表的扫描、路由扫描和DNS扫描等。

基于目标列表的扫描,是指蠕虫在寻找目标之前预先生成一份可能易感染的目标列表,对该列表中的计算机进行传播,然后这些感染的计算机再随机扫描互联网上的其它计算机。路由扫描是指蠕虫利用BGP路由表公开的信息来获取互联网可路由的子网IP地址前缀,从而大大缩小了整个扫描空间,然后再随机扫描子网上的其它计算机。DNS扫描是指蠕虫从DNS服务器获取IP地址来建立目标地址库,这些地址都是可用的,提高了扫描的准确度。

目前广泛使用的Internet网络是建立在IPv4协议基础之上的,传统蠕虫也只是对IPv4网络环境下的主机进行传播。IPv6最终将代替IPv4成为互联网的网络层协议,IPv6拥有128位的巨大地址空间,并且在IPv6网络中的一个子网有64位的地址空间,即一个子网网段内可以有2^{64}台主机[3]。稀疏的地址空间使它对随机扫描蠕虫具有较高的抵制能力,假设已经感染蠕虫的计算机每0.1秒发送1个数据包,扫描2^{64}台主机大约需要585亿年才能完成。笔者经过实验发现,IPv6子网完全可以抵御随机扫描蠕虫的传播。在IPv6网络中基于整个IPv6地址空间的随机扫描蠕虫将彻底消亡。而基于目标列表的扫描和路由扫描的蠕虫在扫描子网时,最终还是要应用随机扫描策略,所以这些扫描策略都无法应用在IPv6网络中。文献[4]主要研究了IPv6网络中DNS延迟对

到稿日期:2009-01-13 返修日期:2009-03-27 本文受国防科工委应用基础资金项目(JI300D004)资助。
徐延贵(1977—),男,博士生,讲师,CCF会员,主要研究方向为网络安全技术等,E-mail:xyangui@163.com;钱焕延(1950—),男,教授,博士生导师,主要研究方向为网络技术与网络安全等;张 凯(1979—),男,博士生,主要研究方向为网络安全技术等。

蠕虫传播的影响，实验结果表明 DNS 蠕虫在 IPv6 网络中具有与 IPv4 网络中的随机扫描蠕虫相似的传播速度，因此，DNS 扫描策略将成为 IPv6 网络中蠕虫传播的首选扫描策略。文献[4]构造了蠕虫通过 DNS 查询在子网与子网间进行传播的模型，但是并没有详细讨论 DNS 蠕虫在子网内传播的扫描策略。

1 DNSWorm-V6 的扫描策略

1.1 DNSWorm-V6 在子网间的扫描策略

DNS 服务是互联网上最基础也是非常重要的服务之一，DNS 服务也能被蠕虫利用来找到易感主机。DNSWorm-V6 在子网间传播时应用的扫描策略定义为 DNS 随机扫描策略，即将随机扫描主机的域名用来代替对主机 IP 地址的扫描，然后使用 DNS 查询来发现在庞大 IPv6 地址空间中的活动 IP 地址，从而定位当前在线的目标主机。

利用 DNS 随机扫描可以分为两个步骤：(1)由 DNSWorm-V6 随机生成一个域名字符串，然后向 DNS 服务器发送此域名查询，从而判断此域名是否对应一台主机；(2)如果得到 DNS 服务器的回应，并且确定此域名对应着一个 IP 地址，那么再向得到的 IP 地址发送扫描数据包，如果此 IP 地址的主机处于活动状态，并且是易感主机，那么 DNSWorm-V6 就可以感染此主机。

DNSWorm-V6 包含一个字符串生成器，用来随机产生 Internet 中可能存在的实际主机的域名字符串。Internet 中的域名由普通的字符串构成，以“.”号分为顶级域名、二级域名、三级域名等等，例如 www.google.cn。顶级域名常用的有 com,net,org,edu,cn 等等。域名由字母(A－Z,a－z,大小写等价)、数字(0－9)和连接符(-)组成。二级域名一般都是常用的词组，例如 net.cn,taobao.com 等等。因此，改进的域名字符串生成器，可以以常用的词组来生成域名字符串。这样通过 DNS 扫描就可大大缩小整个扫描空间。

1.2 DNSWorm-V6 在子网内的扫描策略

由网络及系统安全方面的专家组成的非赢利机构 THC[5](The hacker's choice)，致力于安全解决方案的分析、设计和实现。Alive6 是 THC 开发的攻击工具集的一个针对 IPv6 网络的扫描工具。借鉴 Alive 的扫描方法，可以实现 DNSWorm-V6 在本地子网的扫描策略。

DNSWorm-V6 首先发送一个探测报文—带有逐跳选项报头的 ICMPv6 回送请求报文，源地址为本机 IP 地址，目标地址为 ff02::1，即链路本地范围所有节点组播地址，本地链路所有在线主机都将收到此数据报，仿真实验中的报文如下：

```
Internet Protocol Version 6
    Version:6
    Traffic class:0x00
    Flowlabel:0x00000
    Payload length:48
    Next header:IPv6 hop-by-hop option (0x00)
    Hop limit:255
    Source address:fe80::20d:56ff:fe6d:6ffc
    Destination address:ff02::1
Hop-by-hop option Header
    Next header:ICMPv6 (0x3a)
    Length:2 (24 bytes)
    OptionType:0x90
    OptDataLength:2
    Data:0x0
Internet Control Message Protocol v6
    Type:128(Echo request)
    Code:0
    Checksum:0xf0d7
    ID:0xdead
    Sequence:0xbeef
    Data (10 bytes)
```

IPv6 报头后直接跟随逐跳选项报头(Hop-by-hop option header)，其第一个字节是下一报头，值为 0x3a 表明下一报头为 ICMPv6。第二字节为长度，其值为 0x02，表明逐跳选项报头的长度为 24 字节。第三字节为选项类型，值 0x90 表明该字节的最高位和次高位为二进制“10”。

选项类型字段的最高两位控制着当处理该选项的节点不能识别选项类型时，该如何处理这个数据报。当最高两位为二进制“10”时，采取的动作首先是丢弃数据报，如果 IPv6 报头的目的地址是一个单播或多播地址，就向发送方发出一个 ICMPv6 参数问题报文。因此，本地链路上的所有在线主机收到该数据报后，检查该逐跳选项类型为 0x90，丢弃该数据报，并向已感染 DNSWorm-V6 的主机发送一个 ICMPv6 参数问题报文，仿真实验中的报文如下：

```
Internet Protocol Version 6
    Version:6
    Traffic class:0x00
    Flowlabel:0x00000
    Payload length:96
    Next header:ICMPv6 (0x3a)
    Hop limit:64
    Source address:fe80::2e0:fcff:fe20:d6a8
    Destination address:fe80::20d:56ff:fe6d:6ffc
Internet Control Message Protocol v6
    Type:4(Parameter problem)
    Code:2 (option)
    Checksum:0xa4f6
    Problem pointer:0x002a
```

ICMPv6 的类型码(报头的第一字节)为 4，说明该报文是一个参数问题报文；代码(报头的第二字节)为 2，说明接收端遇到无法识别的 IPv6 选项。

对比两个报文，可以看出，主机 fe80::2e0:fcff:fe20:d6a8 被已感染 DNSWorm-V6 的主机 fe80::20d:56ff:fe6d:6ffc 扫描发现，而其他在线的主机同样会返回如上的数据报。DNSWorm-V6 通过以上扫描原理来发现在本地子网内所有在线主机，从而找到易感主机来传播自身。

2 DNSWorm-V6 的传播模型

理想的网络蠕虫传播模型能够充分反映蠕虫的传播行为，识别网络蠕虫传播中存在的薄弱环节，同时可以预测网络蠕虫可能带来的威胁。在恶意代码传播模型的研究中，病毒传播模型较多，而针对网络蠕虫的传播模型较少。由于蠕虫和传染病的传播方式有着较大的相似性，实际研究中常利用各种传染病动力学模型建立蠕虫的传播模型，如常见的简单传染病模型(Simple Epidemic Model, SEM)[6]、Kermack-

Mckendrick(KM)模型[7]、双因素模型(Two-Factor Model, TFM)[8]等。但是,这些模型都不适合于研究两层扫描策略的蠕虫——DNSWorm-V6。文献[9]构建了本地优先扫描蠕虫的传播模型,但是此模型中蠕虫在本地和外网都是应用随机扫描策略,不适合研究在本地子网内和外地子网间应用不同扫描策略的蠕虫,而且此模型并没有考虑到子网与子网之间传播的延迟和本地子网内传播的延迟所带来的影响。

假设每台主机处于两种状态中的一种:易感染的状态 S 和已感染的状态 I。一台主机一旦被感染就始终保持已感染的状态 I。因此在此模型中每台主机的状态转变过程是:易感染状态 S→已感染状态 I。用 $I(t)$表示 t 时刻已经被感染的主机数,用 $S(t)$表示 t 时刻易感染的主机数,主机总数为 N。因此,有 $S(t)=N-I(t)$。当 $t=0$ 时,$I(0)$表示网络起始时已感主机数,$S(0)=N-I(0)$表示起始网络中易感染的主机数。

假设网络蠕虫的平均扫描率为 k,即已感主机单位时间内平均发送的扫描数据包数量。在时间 δ 内某台已感主机将发送 $k\delta$ 次扫描。设扫描的整个 IP 地址空间为 Ω,一次扫描中任意一个 IP 地址被扫描到的概率为 $1/\Omega$,所以在时间 δ 内,一个特定的 IP 地址被一台已感主机扫描到的概率为

$$q=1-(1-\frac{1}{\Omega})^{k\delta} \tag{1}$$

因为 $1/\Omega<<1$,有

$$q\approx\frac{k\cdot\delta}{\Omega} \tag{2}$$

若 δ 足够小,则可以忽略在时刻 t 到 $t+\delta$ 的时间内,一台已感主机两次扫描同一台易感主机的情况,因此在 δ 的时间段内,一台已感主机将平均感染 $qS(t)$ 台易感主机;若 δ 足够小,也可以忽略在时刻 t 到 $t+\delta$ 的时间内,两台已感主机扫描同一台易感主机的情况,因此在 δ 的时间段内,新增加的已感主机数为 $qS(t)I(t)$。于是有

$$I(t+\delta)=I(t)+qS(t)I(t) \tag{3}$$

$\delta\to 0$ 时,得到单一扫描策略蠕虫简单传染病传播模型,即 SEM,如式(4)所示。

$$\frac{dI(t)}{dt}=\beta I(t)[N-I(t)] \tag{4}$$

其中,$\beta=k/\Omega$,表示感染率。

DNSWorm-V6 应用了两层扫描策略,设 DNSWorm-V6 扫描本地子网的概率为 p,扫描外地子网的概率为$(1-p)$。用 β'和β''分别表示 DNSWorm-V6 在本地子网的感染率和外地子网的感染率。为了简化传播模型,假设整个网络分为 m 个子网,服从均匀分布,即 $N_1=N_2=\cdots=N_m=N/m$,$I_n(t)$为所有子网中第 n 个网络在时刻 t 的已感主机数,因为 DNSWorm-V6 在应用子网内的扫描策略时,在子网内不是对 IP 地址空间 Ω 的扫描,而是对所有本地子网在线主机 N/m 的扫描,因此有 $\beta'=km/N$。这样就可以很快感染整个子网,结果造成有的子网主机已经全部被感染,而有的子网主机全部没有感染,因而 $I_n(t)$不是均匀分布的。DNSWorm-V6 在子网间传播时应用 DNS 扫描策略,因此可以把单个子网看成 DNSWorm-V6 要感染的主机,设 $J(t)$为 t 时刻已感子网的个数,根据式(4),有

$$\frac{dJ(t)}{dt}=\beta_J J(t)[m-J(t)] \tag{5}$$

其中,β_J 为子网的感染率。设 $I_i(t)$为已感子网中第 i 个网络在时刻 t 的已感主机数。为了简化传播模型,假设 t 时刻已感子网中的已感主机 $I_i(t)$服从均匀分布,即 $I_1=I_2=\cdots=I_{J(t)}$,因此每个已感子网有 $I_i(t)=I(t)/J(t)$个已感主机,于是有

$$\beta_J=(1-p)\beta'\frac{I(t)}{J(t)} \tag{6}$$

化简得

$$\frac{dJ(t)}{dt}=(1-p)\beta' I(t)[m-J(t)] \tag{7}$$

DNSWorm-V6 在本地子网传播速度远快于在子网间的传播速度,可以忽略已感子网内的易感主机受到外地子网已感 DNSWorm-V6 主机的 DNS 扫描的影响。根据式(4),有

$$\frac{dI(t)}{dt}=\sum_{i=1}^{J(t)}p\beta' I_i(t)[\frac{N}{m}-I_i(t)]+(1-p)\beta' I(t)(N-J(t)\frac{N}{m}) \tag{8}$$

化简得

$$\frac{dI(t)}{dt}=p\beta'\frac{I(t)}{J(t)}[J(t)\frac{N}{m}-I(t)]+(1-p)\beta' I(t)(N-J(t)\frac{N}{m}) \tag{9}$$

DNSWorm-V6 在本地子网内传播之前,需要获取子网内在线的 IPv6 主机地址列表,而该扫描过程包括发送一个探测报文,收到其他所有在线主机返回的 ICMPv6 参数问题报文,该过程完成后 DNSWorm-V6 才可以感染易感主机。为了描述这种延迟,加入了延迟因子 η,因为已感主机需要经过时间 η 才能在子网内传播 DNSWorm-V6。DNSWorm-V6 在子网与子网间传播时,需要 DNS 查询来获取子网外在线的 IPv6 主机地址列表。为了描述 DNS 延迟,加入了延迟因子 θ,因为已感主机需要经过时间 θ 才能在子网间传播 DNSWorm-V6。DNSWorm-V6 在子网间的感染率与文献[4]研究的 DNS 蠕虫的感染率相类似,利用其研究的结果,取 $\beta'=k/(50\times N)$,从而得出整个网络的 DNSWorm-V6 传播模型,如式(10)所示。

$$\begin{cases}\dfrac{dI(t)}{dt}=p\beta' I(t-\eta)[\dfrac{N}{m}-\dfrac{I(t)}{J(t)}]+(1-p)\beta' I(t-\theta)(N-J(t)\dfrac{N}{m})\\ \dfrac{dJ(t)}{dt}=(1-p)\beta' I(t-\theta)[m-J(t)]\\ \beta'=km/N\\ \beta'=k/(50\times N)\end{cases} \tag{10}$$

本文定义此传播模型为双层模型(Two-Level Model, TLM)。该传播模型适合于研究本地和外网应用不同扫描策略的蠕虫,而且尤其适合模拟本地子网传播比较快、外地子网相对传播比较慢的蠕虫传播过程。可以在此模型的基础上,构建双层 KM 模型、双层双因素模型等等。

3 仿真实验及结果分析

3.1 基于 SEM 对 CodeRed 传播速度的分析

为了比较 DNSWorm-V6 与 IPv4 下的随机扫描蠕虫的传播速度,首先基于 SEM 对 IPv4 下的典型随机扫描蠕虫 CodeRed 进行了仿真实验。假设 CodeRed 以平均扫描速度 $k=$

358 次/分钟[10]，分别在一个 $N=360000$ 台主机的 A 网络和一个 $N=3600000$ 台主机的 B 网络中传播，CodeRed 扫描的空间 $\Omega=2^{32}$ 个 IP 地址，假设初始状态 $I(0)=1$，即除 1 台主机是已感染主机外，其他所有主机都是易感染主机。因为 CodeRed 是采用随机扫描策略在 Ω 地址空间随机扫描，所以 CodeRed 的传播与子网个数无关。基于 SEM 得到其仿真结果，如图 1 所示。

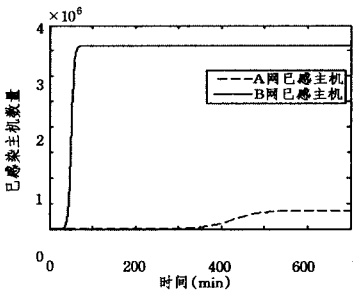

图 1　CodeRed 的实验仿真结果

3.2　基于 TLM 对 DNSWorm-V6 传播速度的分析

再将上述网络移植到 IPv6 网络中，因为 DNSWorm-V6 两层扫描策略都不是对 IP 地址空间的扫描，所以 IPv6 网络的巨大地址空间对 DNSWorm-V6 的传播无影响。假设 DNSWorm-V6 与 CodeRed 具有相同的平均扫描速度 $k=358$ 次/分钟，把 A 网络分为 $m=36000$ 个子网，把 B 网络分为 $m=360000$ 个子网，$p=0.5$，DNSWorm-V6 在本地子网的延迟时间 $\eta=1$min，DNSWorm-V6 的 DNS 查询延迟时间 $\theta=2$min，其他参数不变。基于 TLM 得到其仿真结果，如图 2 和图 3 所示。

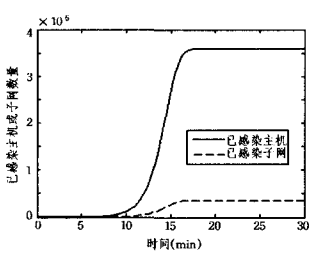

图 2　DNSWorm-V6 在 A 网中的实验仿真结果

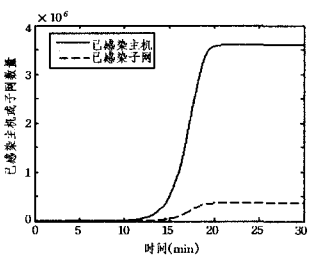

图 3　DNSWorm-V6 在 B 网中的实验仿真结果

3.3　仿真结果分析及结论

由图 1 仿真结果显示，CodeRed 在 IPv4 网络中的 A 网传播过程中，在 580min 左右达到传播的顶峰，网络中处于已感染状态的主机占主机总数的 100%，而在 B 网传播过程中，在 90min 左右就感染了 100%的主机，可见随着主机总数的增加，CodeRed 传播的速度在加快。由图 2、图 3 仿真结果显示，DNSWorm-V6 在 IPv6 网络中的 A 网传播过程中，仅在 18min 时，网络中处于已感染状态的主机就达到了 100%，而在 B 网传播过程中，在 22min 时才感染了 100%的主机，可见随着主机总数的增加，DNSWorm-V6 传播的速度在减慢，但是并不明显。从以上仿真实验结果可以得出结论，无论在 A 网还是在 B 网中，DNSWorm-V6 的感染速度要远远快于 CodeRed，DNSWorm-V6 是完全可以在 IPv6 网络中大范围快速传播的蠕虫。

DNSWorm-V6 能大范围传播的关键参数是 β'，在实验中 β' 的取值为 $k/(50\times N)$，也就相当于 50 次 DNS 查询命中一台在线易感主机。实际中，DNS 查询的命中概率取决于域名字符串生成器，此生成器命中在线易感主机域名的概率是 DNSWorm-V6 大范围传播的关键因素。为了了解 β' 对 DNSWorm-V6 传播的影响，在 A 网中做仿真实验，取 $\beta'=k/(500\times N)$，其他参数不变，得到的仿真结果如图 4 所示。

在此实验中，DNSWorm-V6 在 70min 时才感染了 100%的主机，可见随着 β' 的降低，DNSWorm-V6 传播速度大大减慢。

此外，DNSWorm-V6 可以降低本地子网的扫描概率 p 来达到优化扫描机制的目的。例如，在 A 网中做仿真实验，分别取 $p=0.5$，$p=0.3$，$p=0.1$，其他参数不变，得到的仿真结果如图 5 所示。

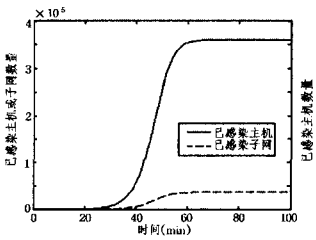

图 4　$\beta'=k/(500\times N)$ 的 DNSWorm-V6 在 A 网中的实验仿真结果

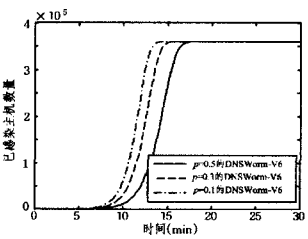

图 5　不同 p 值的 DNSWorm-V6 在 A 网中的实验仿真结果

从实验中可以看出，DNSWorm-V6 通过改变 p 的取值，可以在有限范围内优化扫描机制，提高传播速度。

结束语　通过仿真实验，可以认为：在 IPv6 网络中通过应用两层扫描策略，DNSWorm-V6 蠕虫的感染速度要远远快于在 IPv4 网络中传播的 CodeRed 蠕虫，DNSWorm-V6 是一种可以在 IPv6 网络中大范围快速传播的蠕虫。DNSWorm-V6 的域名字符串生成器命中在线易感主机域名的概率是 DNSWorm-V6 大范围传播的关键因素。因此，域名字符串生成器的设计及优化是进一步研究的课题之一。DNSWorm-V6 利用 DNS 服务来发现易感主机，从而在子网间进行传播，因此在未来的 IPv6 网络中，DNS 服务器的安全机制需要进一步强化，同时要深入研究遏制此类 DNS 蠕虫在 IPv6 网络中大范围快速传播的有效措施。

参 考 文 献

[1] Moore D, Shannon C, Brown J. Code-Red: A case study on the spread and victims of an Internet worm[C]// Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement. Pittsburgh, 2002: 273-284

[2] Staniford S, Paxson V, Weaver N. How to own the Internet in your spare time[C]// Proceedings of the 11th Usenix Security Symposium. San Francisco, 2002: 149-167

[3] Hinden R, Deering S. IP Version 6 Addressing Architecture. RFC 4291, Internet Engineering Task Force, 2006

[4] Kamra A, Feng H, Misra V, et al. The Effect of DNS Delays on Worm Propagation in an IPv6 Internet[C]// Proceedings of the IEEE INFOCOM 2005. Miami, 2005: 2405-2414

[5] van-Hauser THC. Attacking the IPv6 Protocal Suite[OL]. http://pacsec.jp/ psj05/psj05-vanhauser-en.pdf, 2005

[6] Streftaris G, Gibson G J. Statistical inference for stochastic epidemic models[C]// Proc. of the 17th Int'l Workshop on Statistical Modelling. Chania, 2002: 609-616

[7] Frauenthal J C. Mathematical Modeling in Epidemiology [M]. New York: Springer-Verlag, 1980

[8] Zou C C, Gong W B, Towsley D. Code Red worm propagation modeling and analysis[C]// Proceedings of the 9th ACM Symposium on Computer and Communication Security. Washington, 2002: 138-147

[9] Zou C C, Towsley D, Gong W. On the performance of Internet worm scanning strategies[J]. Performance Evaluation, 2006, 63: 700-723

[10] Zou C C, et al. The monitoring and early detection of Internet worms[J]. IEEE/ACM Transactions on Networking, 2005, 13 (5): 961-974

(上接第 7 页)

3.2.4 U 方法

区分序列和 W 集合都是在有限状态机所处的状态完全未知的情况下对状态进行确定的一种方法，但是实际上，在测试子序列生成过程的第三步，对 IUT 所处的状态有一个期望值 s_j，在测试中只需要确定 IUT 的实际状态是否为 s_j。由此来看，区分序列和 W 集合的状态确认能力相对于测试要求来说太强了，这就是 U 方法的基本出发点。

IUT 状态 s_j 的 UIO 序列是 IUT 所有其它状态不能表现的 I/O 行为，它唯一地标识状态 s_j。为了找出各个状态的 UIO 序列，必须罗列出 IUT 各个状态的 I/O 序列(一棵 I/O 序列树)，从树的根部开始比较各个状态的 I/O 序列，直至为每个状态找到唯一的 I/O 序列为止。

相对于 D 方法和 W 方法，U 方法能够生成更短的测试序列，并且 UIO 在大部分的有限状态机中是存在的，因而其适用范围更广。

3.3 测试序列到抽象测试集的转换

测试集可分为通用测试集 GTS、抽象测试集 ATS 和可执行测试集 ETS。测试生成阶段得到的测试序列属于通用测试集的初级阶段，需要经过规范化转换到通用测试集。采用适当的测试描述法描述通用测试集就能够得到抽象测试集；输入到特定的测试系统并结合被测协议实现信息就能够得到针对该实现的可执行测试集。

实际测试时，把自动生成的测试序列按照相当确定的测试目标进行分解，得到针对每一个测试目标的测试子序列，在测试子序列的基础上构造每一个测试例。分析每一个测试子序列，找到其明确的测试点(比如测试一个状态、一个变迁等)，对子测试序列进行分割，把从子序列起始位置到测试点的初始位置作为前测试步序列，完成测试驱动和状态验证的剩下部分作为测试体序列。根据子序列执行后的最终状态，添加能够使被测协议转换到初始状态的后测试步序列。三部分序列组合起来，得到针对特定测试点的完整的测试序列。采用适当的测试描述法对上述得到的测试序列进行描述，就得到一个完整的测试例。如此构造出针对每一个特定测试点的测试例，就能够组合成抽象测试集。

4 测试实现和测试执行

4.1 测试实现

在测试实现阶段，根据协议实现的 PICS 和 PIXIT 从一致性测试集中选取适当的测试例，去除没有意义的测试，并使用 PIXIT 提供的信息来量化这些测试例。从抽象测试集生成可在一实际的测试系统上执行的参数化的可执行测试集，即可在特定测试设备上对某个 IUT 进行测试运行的测试集。

4.2 测试执行

在测试执行阶段，一个特定的 IUT 被实际测试，并得出 IUT 一致性判定结果。测试执行过程分为两步：第一步是静态一致性需求检查，根据协议标准的静态一致性需求对 IUT 的 PICS 进行检查；第二步为在测试器上执行测试例来检查 IUT 对动态一致性要求的满足程度，对每个测试例做出测试判断：通过、失败或不确定。最后，静态一致性检查的结果和所有的测试例的执行判定结果组合在一起，形成一个有关 IUT 的一致性判决。当且仅当所有的测试都未失败时，最终的判决才会是通过。

现有的测试执行方法可划分为两类：基于编译的测试执行(CTE)和基于解释的测试执行(ITE)。基于编译的测试执行，是指在测试执行之前，由抽象测试集 ATS 到可执行测试集 ETS 的转换已经由转换器或编译器完成，这一过程非常耗时，但是提高了测试执行的效率。在基于解释的测试执行中，从 ATS 到 ETS 的转换是在测试执行过程中完成的，这种方法使得用户可以对测试过程进行动态观察和控制，但测试执行的效率较低。

结束语 协议一致性测试所面临的挑战是双重的。一方面，随着协议的全方位发展，协议的功能越来越强，协议的复杂性也越来越高，使得协议的一致性测试变得越来越困难；另一方面，随着形式化验证技术的发展，对于协议一致性的验证必然会面临如何提高形式化验证的效率的问题。

目前一致性测试的研究和实践中需要解决的几个关键问题包括测试理论的形式化、高速计算机网络协议和路由协议的测试、通用测试平台的研制。所有这些都需要新的、高效可行的形式化验证技术的支持。

我们认为，面对协议一致性测试领域的困境，形式化的领域本体可能会对协议的形式化验证起到关键的作用，我们未来的工作将主要沿着这条路径展开。

参 考 文 献

[1] Zhang Y, Li Z. A New Formal Test Suite Specification Language for IPv6 Conformance Testing[C]// Proceedings of ICCT2003. 2003: 174-177

[2] Wu J, Samuel T, Gao Q. Formal Methods for Protocol Engineering and Distributed Systems[M]. Kluwer Academic Publishers, 2001

[3] Tian J, Li Z. The Next Generation Internet Protocol and its Test [C]// Proceedings of IEEE International Conference on Communication. 2001: 210-215

[4] Zhang F, Cheung T. Optimal Transfer Trees and Distinguishing Trees for Testing Observable Nondeterministic Finite State Machines[J]. IEEE Transactions on Software Engineering, 2003, 29 (1): 1-14

[5] 朱雪峰，金芝. 关于软件需求中的不一致性[J]. 软件学报，2005, 16(7): 1221-1232