状态监测技术的研究

Research on Status Check Technology

赵仲孟　饶芳艳　沈钧毅
（西安交通大学计算机系　西安710049）

Abstract This paper introduces the status check technology, and demonstrates how to use it in a firewall as FTP protocol for example.
Keywords Status Check, Firewall

1. 传统防火墙技术的缺点

传统的防火墙技术可分为包过滤和应用代理两大类。包过滤技术是在网络层依据系统的过滤规则对数据包进行选择和过滤的技术。该技术通过检查数据包的源地址、目标地址、源端口、目标端口及协议状态来确定是否允许该数据包通过。包过滤具有过滤效率高，成本低，易于安装和使用的特点。包过滤之所以能够通过对源目地址和端口进行过滤来达到控制上层应用服务的目的，是和应用层协议的设计紧密相关的。基本上所有的应用层协议都遵循客户/服务器模式，服务器提供服务的端口往往是在协议设计时规定好的，对这一特点进行过滤特征提炼从而制订规则表作为防火墙的安全策略。但是，有的协议需要建立第二条连接，如 FTP，RealAudio。对这样的协议缺乏有效的过滤手段。

应用代理用来提供应用层服务的控制，起到外部网络向内部网络申请服务时中间转接的作用，内部网络只接受代理提出的服务请求，拒绝外部网络其它节点的直接请求。应用代理一般在网关上为每一种服务提供一个代理。服务代理一般不允许通信直接通过内部网和外部网，而是客户先与防火墙服务代理建立一个连接，服务代理再与服务器建立另一个连接。这样服务代理可在其中对客户与服务器之间的通信进行控制。当然，服务代理所起到的中转不是单纯的包中继，每一个包是否被中转要受安全规则控制，以 FTP 为例，不能让外界取走的文件，则不能让 get 通过，外界不能写的文件则这种 get 请求被拒绝。应用代理的一个主要优点是它可以提供严格的身份验证，登录及记帐。还能对应用协议数据进行扫描，以提供内容安全(Content security)手段。但应用代理也有它的弱点：对每一个 TCP 连接，都需要启动一个相应的代理进程或线程进行处理，这样必然占用大量的 CPU 和内存资源，因此代理服务器能够支持的连接容量有限；由于每一个 IP 包都要经历网络层，传输层乃至应用层的拆包才能进行处理，因此系统的吞吐量较低，时延较长，效率低(low performance)；由于对每种系统应用都需要提供各自相应的代理程序，这就是所谓的有限的连接能力(limited connectivity)，随着网络应用的不断增加，代理程序的类型也会越来越多，在管理上造成困难，缺乏必要的灵活性；另外，由于其自身的复杂性，更容易有安全漏洞而受到攻击。

2. 状态检测技术

针对以上两种技术各自的优缺点，Check Point 公司提出了一种新的信息跟踪方式——状态检测(stateful inspection)[1]。它一般工作在系统内核，通过截获网络中的数据流，对信息进行状态提取，由此实现信息监控功能。状态检测的提出基于这样一种思想：防火墙要有效地提供真正的安全，应该能够跟踪并控制穿越它的信息流。孤立地检查某个包是不够的，还要从以往的通信及其他应用中得出的状态信息确定每个新的通信企图，从以往的通信中获得的通信状态和从其他应用中得出的应用状态作出控制决定。

该技术的提出者认为进行控制决定，防火墙要具备获得、分析和利用如下信息：

◇通信信息(Communication information)：报文中所有七层协议的信息。

◇通信导出信息(Communication-derived state)：从以往的通信中获得的通信状态。如在一个 FTP 会话中，向外的 PORT 命令应该被存储起来，以供在向内的 FTP 数据连接请求中进行检查。

◇应用导出状态(Application-derived state)：从其他应用中得出的应用状态。如一个刚被认证的用户可能被允许穿过防火墙访问其他授权的服务。

◇信息操纵(Information manipulation)：根据上述因素，进行灵活的评估。

应用代理能跟踪通信双方的通信状态，但是只能获得部分通信信息，缺乏抵御诸如 IP 欺骗的报文从哪个网络接口进入等信息。包过滤可能获取全部通信信息，但是不对应用层信息进行分析，只能获得通信导出状态信息，而不能获得任何应用导出信息，所以这类防火墙对象 FTP、RealAudio 等需要建立第二条连接的应用没有很好的支持方案。

状态检测综合了这两种技术的优点，在包过滤的基础上，对于报文的分析可达应用层，既高效又灵活。由于要获得所有通信信息，因此需要工作在数据链路层以上。状态监测和包过滤以及应用代理的比较见表1。

表1　三种技术的比较

防火墙获取信息能力	包过滤	应用代理	状态监测
通信信息	部分	部分	有
通信导出信息	无	部分	有
应用导出状态	无	有	有
信息操纵	部分	有	有

但是，这里也遇到了另一个问题。状态检测破坏了网络体系结构中的分层原则，即每层不必了解上层的数据含义的设计思想，企图在数据链路层解释应用层数据。这样做的得益是绕开了协议栈的处理开销，要在包过滤防火墙上实现完整的 TCP/IP 协议无疑是代价高昂和不必要的。这一点很类似于在 IPv4到 IPv6的过渡过程中，有关“做得对”和“做得快”的争论。

3. 应用状态监测技术过滤 FTP 协议

下面以 FTP 协议为例说明状态监测技术的处理方式及

其优点。

FTP 服务在一般的实现中需要两次 TCP 连接[2](如图1):一条用于控制连接,一条用于数据连接。控制连接服务端口发生在服务器端口21号,客户端口号随机;数据连接则由服务器发动连接。由于必须由服务器发动数据传输的呼叫,这样,当外部从20号端口发起一向内的连接(反向呼叫),就无法判断它是正常的 FTP 连接,还是非法攻击。

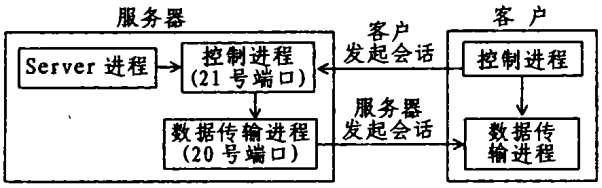

图1 FTP 客户-服务器模型

包过滤不能完善解决 FTP 的过滤问题。可以采取的措施是利用客户端连接端口的范围来决定如何过滤。对于大多数服务的端口,大多数攻击目标都集中在低端口号上,而大多数向外的呼叫使用高端端口,通常大于1024。我们在规则集中加入如表1的三条简单规则,就能对数据包进行较好的过滤。但不幸的是,所有高端的端口都暴露于黑客的攻击之下。黑客通过冒充合法的 FTP 主机地址和20端口,就能够通过防火墙攻击保护子网内主机的高端端口。

表2 过滤 FTP 的规则

动作	协议	源 IP 地址	源端口	目的 IP 地址	目的端口
Permit	Tcp	ourhost	ourport	theirhost	21
Permit	Tcp	theirhost	20	ourhost	>=1024
Deny	Tcp	theirhost	20	ourhost	<1024

应用代理可以解决 FTP 反向呼叫的问题,但是需要建立6条连接,效率低下。

应用状态监测技术分析 FTP 协议报文,对 FTP 协议处理可以控制得更为细致灵活。FTP 协议是建立在 TCP 层以上的。要分析 FTP 协议报文,首先需要对 TCP 连接的状态进行监视。状态监测引擎为每个 TCP 连接维护一个动态变化的状态表,存放于 TCP 缓存中。在 TCP 层,可以根据 TCP 包中所包含的源和目的端口号来区分该包所属的具体应用,从而将具体应用的所有信息记录在一起。TCP 是有连接的协议,所有 TCP 应用均分为三个阶段:连接建立、数据传输、连接拆除。为了记录每一个会话的完整数据,就必须监控和分析每一个阶段的状态。从 TCP 包头部信息中的6个标志位可以得到 TCP 应用的状态信息,这六个标志位(URG,ACK,PSH,RST,SYN,FIN)中与连接状态关系较密切的为以下三个标志:ACK、FIN、SYN。通过对这三个标志的判断即可获得 TCP 连接的状态,下面是 TCP 的连接过程及状态标志。

(1)连接的建立 TCP 的连接建立过程分三步,即三次握手(Three-Way Handshaking)[3],如图2所示。

客户将 SYN 请求同步位置1的连接请求包发给 TCP 服务器,TCP 服务器接收到 SYN 后,利用 ACK 和 SYN 来进行应答,发起方再对应答进行确认,即发送一带 ACK 的包给服务器端,于是双方的连接建立。

因此当状态监测引擎收到 SYN=1的包,表示此时网络上准备新建一个连接,就在 TCP 缓存中为这个新的连接准备必要的空间,并将查找规则表的结果记入缓存。

(2)数据传输 在传输阶段,ACK 始终置位,TCP 通过 ACK 和序列号来确认数据的正确传输,引擎查找缓存来获得过滤动作和日志动作。

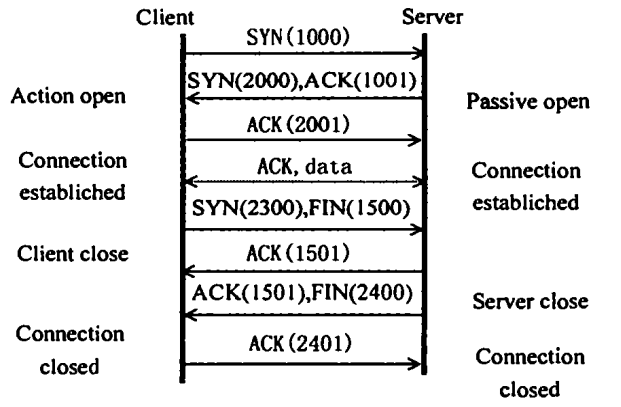

图2 TCP 的连接建立和释放过程

(3)连接的结束 TCP 客户在发送完数据后,为了拆除连接,需要经过两个过程:向服务器方发送带有 FIN 标志的包,服务器方接收到后,应发送 ACK 来确认连接的结束;当连接结束时,把在 TCP 缓存中的空间释放掉。

在对 TCP 连接进行动态监控的基础上,状态监测引擎进一步分析 FTP 协议数据维护状态表。目前 FTP 客户端普遍采用 PORT 命令来通知 FTP 服务器连接端口。PORT 命令的参数包括客户端期待 FTP 服务器连接的端口号(Nc),服务器下次就在20端口和 PORT 命令制定的端口(Nc)间发起连接。在检测到 FTP 报文中的 PORT 命令后,在 TCP 缓存中加入允许该 FTP 数据端口(20)到客户端 Nc 端口之间通信的记录。当 FTP 服务器20端口向 Np 发起连接时,状态监测引擎根据 TCP 缓存记录转发数据包。当连接关闭后,引擎立即将该记录删除。这样,只是在 FTP 会话的过程中一段时间内允许从 FTP 服务器20端口到内部网主机一个特定端口的连接,在满足可用性的同时获得最大的安全性。

如果客户端采用 PASV 命令通知 FTP 服务器进入被动模式,等待来自客户端的连接。FTP 服务器会发给客户端一个回应为227的数据包,通知客户端期待连接的端口号(Ns),被动等待客户端向该端口发起连接。在这种情况下,状态监测引擎也可以用类似 PORT 命令的处理方法向 TCP 缓存动态加入允许 FTP 服务器 Ns 端口和内部网客户端间通信的记录。

总结 未来防火墙的发展方向是将防火墙置于低层和高层之间。网络层的包过滤防火墙会更关心 IP 包的内容,而应用代理会向低层靠拢,变得更加透明。这样,结合两者的优点,使得防火墙成为一个快速的数据包屏蔽/转发系统,提供良好的日志和审计功能。状态检测技术的提出就是这一发展趋势的表现。

状态监测技术工作作为一种新型的防火墙技术,综合了包过滤高效和代理安全性高的优点,工作在数据链路层之上,在传统的包过滤基础上增加对应用协议的理解和对通信状态的追踪,已经在 CheckPoint 的 Firewall-1防火墙上实现;其他基于包过滤的防火墙,如基于 Linux 的 IPChains,正在朝着这个方向努力。我们在基于 Linux 带安全通道的防火墙原型系统——NetGuard 的设计和实现中,也参考了该技术。

参 考 文 献

1 Stateful Inspection Firewall Technology. http://www.checkpoint.com/products/technology/stateful1.html
2 Reynolds J K, Postel J. File Transfer Protocol. RFC959. Oct-01—1985
3 Postel J. Transmission Control Protocol. RFC793. Sep-01—1981