第28卷 第1期 计 算 机 工 程 2002年1月
Vol.28 № 1 Computer Engineering January 2002
·安全技术· 文章编号：1000—3428(2002)01—0169—03 文献标识码：A 中图分类号： TP393.08

网络型入侵检测系统存在的漏洞及其对策研究

张铭来， 金成飚， 赵文耘
(复旦大学计算机科学与工程系，上海 200434)

摘 要：网络安全技术中的一个重要方面就是入侵的检测，从数据链路层、网络层和传输层3方面讨论了基于网络的入侵监测系统存在的一些漏洞，分析其可能遭受的攻击，以及相应的防范措施。

关键词：网络入侵检测；防火墙；拒绝服务攻击

Problems of Network-based Intrusion Detection System and Their Countermeasures

ZHANG Minglai， JIN Chengbiao， ZHAO Wenyun
(Computer Science and Engineering Department of Fudan University, Shanghai 200434)

【Abstract】Intrusion Detection is an important technology in network security domain . This paper discusses some problems of network -based intrusion detection system in three ways: the data link layer, the network layer and the transport layer， and analyzes the attack on the network-based IDS and its countermeasure.

【Key words】Network intrusion detection； Firewall； Denial of service

1 入侵检测系统的CIDF模型

Common Intrusion Detection Framework (CIDF) (http://www.gidos.org/)阐述了一个入侵检测系统的通用模型。它将一个入侵检测系统分为以下组件：

- 事件产生器(Event generators)
- 事件分析器(Event analyzers)
- 响应单元(Response units)
- 事件数据库(Event databases)

CIDF将IDS需要分析的数据统称为事件(event),可以是网络中的数据包，也可以是从系统日志等其他途径得到的信息。事件产生器的目的是从整个计算环境中获得事件，并向系统的其他部分提供此事件。事件分析器分析得到的数据，并产生分析结果。响应单元则是对分析结果作出反应的功能单元，可以作出切断连接、改变文件属性等反应，也可以只是简单的报警。事件数据库是存放各种中间和最终数据的地方的统称，可以是复杂的数据库，也可以是简单的文本文件。

2 “插入”攻击

一个基于网络的IDS从网上获取数据包从而判断被其监视的主机会发生什么反应，网络IDS要预测被监视的主机交换这些数据包所采取的行为，而问题就在于一个被动的网络监视器无法准确地预测目的主机是否能看见这个数据包，更不用说这个数据包是如何处理的了。

一个IDS可能接收了一个目标系统(被IDS监测的主机)拒绝的数据包，这将使IDS误以为目标系统接收并处理了该数据。攻击者通过发一些目标系统拒绝而IDS又认为有效的数据包来实现这一过程，实际上是攻击者“插入”了一些数据到IDS——网络上的其它系统都不会理睬这些包。目前不少IDS产品都存在漏洞使“插入”攻击成为可能。攻击者可以通过"插入"攻击避开IDS的基于标志(signature-based)的分析。

要理解“插入”攻击是如何欺骗基于关键字的标志分析，先要了解IDS中标志的识别技术。标志识别通常用"关键字比较"(pattern-matching)算法来检测一个数据流里面是否存在某一个特定的字符串。例如，一个要检测PHF攻击的IDS将在HTTP的GET请求中寻找phf字符串， 实际上可能是这样一个字符串“GET /cgi-bin/phf ”。IDS通过搜寻子串能很容易地在HTTP请求中发现phf， 然而，当Web服务器收到这个请求而IDS看见的是另外一个字符串时，事情就变得复杂了，比如IDS看到的字符串是“GET /cgi-bin/pheatfg”，攻击者利用“插入”攻击在原先的字符串中加上了eat和g。这时IDS就无法从数据流中发现字符串“phf ”。如图1所示。

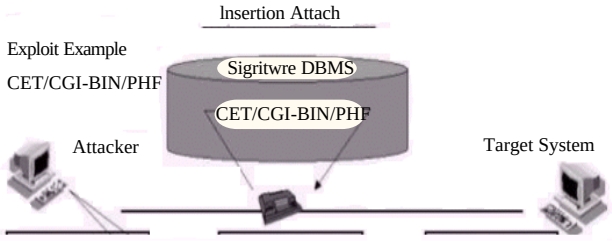

图1 “插入”攻击

3 “逃避”攻击(Evasion)

目的主机接受了一个被IDS拒绝的数据包，称之为“逃避”攻击。Evasion 和Insertion一样能骗过IDS的关键字识别，因为IDS看到的数据流和目的主机看到的不一样，这次目的主机看到的数据比IDS多，而IDS错过的数据恰恰是发现攻击很重要的数据。攻击过程如图2所示。

作者简介：张铭来(1977～)，硕士生，研究方向：软件工程，网络管理；金成飚，硕士生；赵文耘，教授

收稿日期：2001-06-05

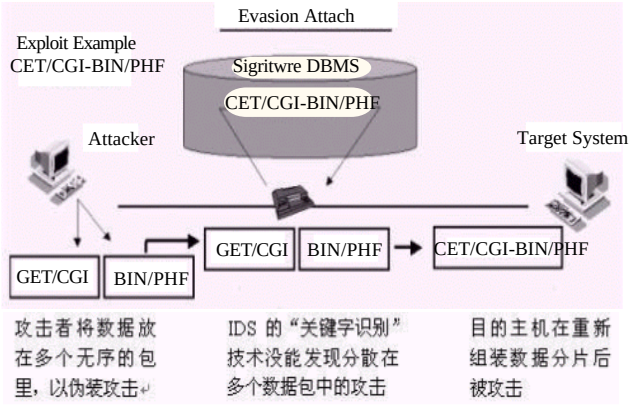

图2 “逃避”攻击

4 攻击实例

攻击者有许多方法使发出的数据包只有IDS才能收到，或者IDS收不到，因为每一种操作系统对数据包的处理都不尽相同，而这些潜在的不同就可能被攻击者利用。以下是几种常见的攻击手段：

⑴ 数据链路层的攻击

在数据链路层存在“插入”攻击。一个被攻击者利用的主机如果和IDS在同一个LAN中，并且知道IDS的MAC地址，就可以直接将伪造的数据包发送给IDS，LAN上的其它主机都不会处理这个数据包，如果IDS不检查MAC地址就不会知道这些。如图3所示。

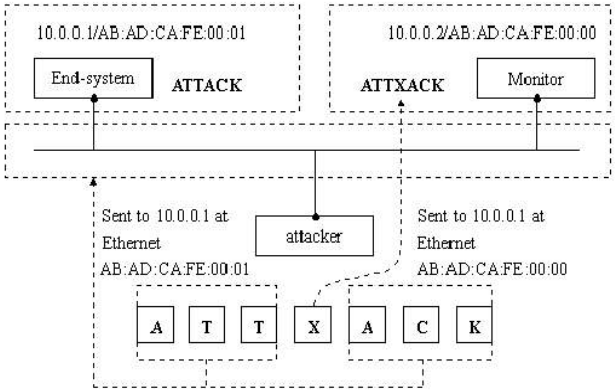

图3 数据链路层的攻击实例

一个攻击者通过直接发送数据链路层的包来实现"插入"攻击。即使攻击者不知道IDS的MAC地址，也可以利用IDS网卡的“混杂模式”(promiscuous mode)来攻击：发送数据包到不存在的MAC地址，IDS同样会收到该数据包。因此IDS可能被数据链路层错误的寻址所欺骗，除非IDS对IP报头的目的地址和正确的MAC地址进行核实。

⑵ IP层的攻击

1) 错误的IP报头　使IP包被目的主机丢弃的最简单的方法是无效的IP报头。但其中存在一个问题，无效的IP报头在Internet的传输过程中可能被路由器丢弃，使用这样的数据包来攻击变得困难，除非IDS和攻击者在同一个LAN中。

另一种攻击来自于IP报头的语法错误，攻击者可以指定一个错误的IP报头大小，或是指定错误的IP报头的大小，从而使IDS无法知道传输层的数据从哪里开始。

IP报头中另一个容易被忽视的地方是校验和。IDS对每一个获得的IP包都检查其校验和似乎是不必要的，然而，一个校验和错误的IP包不会被大多数操作系统处理。一个IDS如果不拒绝具有错误校验和的包就很容易受到“插入”攻击。

一个较难解决的问题是TTL域。IP报头的TTL域规定了一个数据包在网络上最多能经过几个路由器，每经过一个路由器TTL的值就会减1，当TTL为零时该数据报就被丢弃。如果IDS和被监视的系统不在同一网段上，就存在这么一种可能：设定一个恰当的TTL值使IDS能收到该数据包，而目地主机收不到该数据包，从而实现“插入”攻击。

一个类似的问题同样存在于IP报头的DF标志中。DF标志告诉网关当某一网络的最大分组长度小于数据包的长度时不进行分片，而是丢弃该数据包。如果IDS所在的网段的最大分组长度大于目的主机所在网段的最大分组长度，就有可能IDS收到该数据包而目的主机没有收到，从而产生"插入"攻击。

2) IP选项　IP校验和的问题是很容易处理的，IDS可以假定校验和错误的数据包不会被目的主机接受。一个稍难的问题是IP选项，不同的主机对其解释和处理有可能不同。例如，如果数据包的“严格源路由”选项作了设置，而主机的IP地址又不在其中，大多数主机的处理是丢弃该包。IDS为了避免插入攻击而丢弃这些数据包是合理的。然而许多操作系统可以被设置为自动拒绝源路由的包，所以IDS在不知道目的主机如何处理源路由包的情况下是无法采取适当的措施的。

IP选项中另一个可能产生问题的是"时间戳"，它要求一些特定的接收者在数据包中的时间戳上作记录。目的主机可能根据时间戳上的记录来决定是否丢弃该数据包，如果IDS对"时间戳"的处理和目的主机不同，就有可能出现插入攻击。

3) IP分片　IP包在传输过程中被分为更小的包，到达目的地后被重新组装，这被称为IP分片，是IP协议的一部分。由于目的主机要将IP分片重新组装，那么负责网络监控的IDS是否能正确地组装这些分片就显得很重要了。在组装IP分片时可能出现的问题：

①IDS在IP分片时可能采用的方法是等待所有的分片到达后再组装，这将使IDS必须保存每个IP分片直到全部到达，攻击者可以发许多不完整的IP分片使IDS最后资源耗尽。目的主机也必须面对这一问题，许多系统依据分片的TTL来丢弃它们，以保证系统资源不会耗尽。如果IDS也采用某种策略来丢弃旧的、不完整的分片，那么它必须和目的主机采用的策略一致，否则就容易受到攻击。

②重叠的分片(Overlapping Fragments)

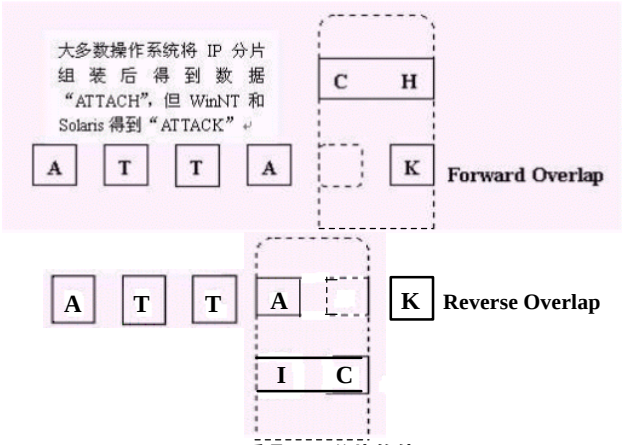

图4 “重叠”IP分片的处理

分片的重叠和IDS是密切相关的，指不同大小的分片无序地到达目的地并且存在重叠的部分。如果一个到达目的主机的分片所含的数据已经由另一个分片送到了，就可能出现新到的数据将旧的数据覆盖。于是IDS面临一个问题，如果它对重叠分片的处理与被监控系统的处理不同，就可能在重新组装后得到一个不同的数据包，尽管收到的分片是相同的。攻击者如果知道这一个不同点，便可以

利用重叠分片中的数据进行攻击。重叠数据的处理是比较复杂的，跟相互冲突的分片的位置有关。在某些情况下，会优先考虑新的数据，有时候又会优先考虑旧的数据而将新数据丢弃，图3说明了IP分片重新组装的几种情况。

不同的操作系统处理也有所不同，如表1所示：

表1 不同操作系统对重叠IP分片的处理

操作系统	对重叠的IP分片的操作
Irix 5.3	Favors New Data for Forward Overlap
HP-UX 9.01	Favors New Data for Forward Overlap
Linux	Favors New Data for Forward Overlap
Solaris 2.6	Always Favors Old data
4.4 BSD	Favors New Data for Forward Overlap
Windows NT 4.0	Always Favors Old data

⑶ TCP层的问题

和IP协议一样，有几种不同的方法可以对IDS进行“插入”攻击。TCP的处理比较复杂，可能有多种因素导致一个包被丢弃。正如前面讨论过的，如果IDS处理TCP包的过程与被监控的系统不同，就容易受到攻击。

① TCP包头　如果在设计TCP会话监视器时没有考虑TCP包头中的某些域，就容易受到“插入”攻击。例如TCP包头中有一些二进制的标志位(URG、ACK、PSH、RST、SYN、FIN)，这些标志位的某些特定的组合是无效的，会导致接收方丢弃该数据包，而且许多系统在实现TCP时不接收标志位ACK没有设置的数据包。通常系统会接收SYN包中的数据，但有些系统不能正确处理这些数据，这将会导致针对IDS的“插入”或“逃避”攻击。另一个容易忽视的问题是TCP的校验和，所有的TCP实现都要求对其进行检查，但许多IDS却没有这么做，因此攻击者可以用TCP校验和错误的包来实现“插入”攻击。

② TCB的建立　要监视一个TCP会话必须先建立一个TCB(TCP Control Block)。IDS建立TCB的策略决定从哪一点开始记录连接的数据，以及会话的初始状态。IDS可以根据TCP连接的3次握手来创建一个TCB，即“3次握手同步”(synch on 3WH)。但这么做有一些明显的不足：第一，IDS如果没有发现一个连接的3次握手，就会错过整个TCP连接，即IDS无法监控启动以前已经存在的TCP连接，如果攻击者能避免3次握手被发现，就可以使整个连接不被IDS发觉；第二，如果IDS以3次握手来确定连接的序列号，以后将每个包的序列号和初始的序列号进行比较以确定它属于哪个连接，那么攻击者很可能先伪造一次连接(3次握手)，这时IDS会将此连接序列号记录下来，并建立一个TCB，当客户机和服务器用不同的序列号建立真正的连接时，IDS就不会跟踪这次连接了(因为在某一时刻，一个TCB唯一对应客户端的IP、端口以及服务器的IP、端口这4个参数)。另一种建立TCB的方法是通过观察网络上的数据包来推断连接的初始状态，即"数据同步"(synch on data)。这种方法的主要优点是：可以获取更多的数据包。即使没有发现3次握手，也能跟踪该连接。但这种方法也存在问题，因为依靠序列号来判断连接的存在，攻击者可以在攻击以前发许多伪造的数据包来干扰IDS，让IDS建立一个无效的TCB，然后再进行攻击。

另一种改进方法是：允许IDS进行“数据同步”，同时注意3次握手的数据包。最初通过观察到的数据包来确定连接的初始状态，一旦看见3次握手就将此连接重新初始化。为了防止攻击者伪造源地址为服务器(被攻击方)的数据包，必须在恰当的地方加上包过滤器，使得至少服务器端的应答(SYN+ACK)是可信的(攻击者仍可伪造SYN包)，这样IDS的会话监视才有更高的可信度。

⑷ 拒绝服务攻击

针对IDS的拒绝服务攻击是严重的，因为IDS是“失败开放”(fail open)的，即IDS如果停止工作以后，攻击者仍可以访问被IDS监控的网络和主机，而防火墙则不同，一旦防火墙被攻破，它可以切断攻击者和内部网络的连接。因此IDS攻击者的目的就是在能够访问被攻击主机的同时使IDS失效。

① 耗尽资源

对IDS有许多不同的方法来实现拒绝服务攻击，首先讨论的与耗尽资源有关，攻击者可能耗尽的资源有：CPU、内存、磁盘空间和网络带宽。

IDS读包、分析其内容，把它们放进相应的队列，以及将IP分片或TCP分段重新组装，这些都要消耗CPU，攻击者可以知道IDS的哪些操作是最消耗CPU的，从而对其进行攻击。例如IDS对IP分片的处理，由于IDS要处理许多主机的IP包，每个IP包可能有若干个分片，IDS对分片的分类将消耗CPU资源，而这种分类查找算法通常是线性表的，当线性表的长度相当大时，查找效率就很低了，攻击者可以通过发送许多微小的IP分片来耗尽CPU资源。

IDS组装TCP分段、IP分片，跟踪TCP连接的状态，分析数据包的内容，这些都要消耗内存，攻击者同样可以发出许多特定的数据包，让IDS为处理这些包而耗尽内存。

IDS通常要在磁盘上保留事件的日志，每个事件都会消耗一定的磁盘空间，攻击者可以产生许多无意义的事件来消耗磁盘空间，以致IDS没有足够的空间来记录真正有效的事件。

使IDS耗尽资源最简单的办法可能是消耗网络带宽了。因为IDS要跟踪网络上所有的活动，不像一般的主机只关心目的地址是自己的数据包，所以IDS很可能来不及处理接收到的许多包，特别是网络很繁忙的时候。攻击者可以向网络发大量的无意义的数据包，以致IDS跟不上网络上的数据流，从而无法监视网络。

② 利用IDS的响应措施

在某些情况下，IDS自身可能被利用来进行拒绝服务攻击。IDS如果设置了针对攻击的一些响应措施，(例如一旦发现某个IP在进行攻击就自动改变路由器的设置，过滤掉该IP)就可能被攻击者利用。攻击者假冒一个合法的用户的IP进行攻击，IDS响应这次攻击后实际上关闭了合法用户对网络的访问，即产生了拒绝服务攻击。所以，对IDS的设置必须谨慎，以避免攻击者有机可乘。

5 防范策略与IDS的改进

从以上分析看出基于网络的IDS因为是以sniffer为基础，完全依赖网络型的IDS来保护网络和系统的安全是不足取的，必须以综合的防范措施来实现网络和系统的安全。

防火墙作为网络安全的第一道屏障是必需的。如果把网络型的IDS比作一幢大楼里许多隐蔽的监视器，那么防火墙就是大楼的门卫，虽然门卫不能保证进入大楼的人都是安全的，但毕竟可以挡住一部分可疑的人进入大楼。防火墙的作用不仅可以过滤掉一部分可能破坏网络安全的数据包，还可以对访问内部网络的用户进行身份验证，并作好日志记录。防火墙可以只对外开放几个需要使用的端口，把一些不需要对外开放的服务隐藏起来，使攻击者不能获取足够的内部系统的信息，从而阻止攻击的发生。防火墙可以进行地址转换(NAT)以便隐藏真实的内部地址，上面讨论发现许多针对IDS的攻击都要伪造数据包的IP地址来欺骗IDS，如果攻击者无法确定目的主机及IDS的地址，就不能实现地址伪装。

正确进行网络分段也是维护安全所必需，利用防火墙可以将网络分为外部网、内部网和DMZ区，对不同网段之间的数据流加以限制可以防止入侵发生的可能。一个网络型IDS只对同一网段的系统进行监视，这样IDS要处理的数据流就会减少，也可以避免攻击者使用TTL的"插入"攻击。

(下转第194页)

踪。

```
App_Framework *appFramePtr=new App_Framework
appFramePtr->ESAppInitialize((ESParam)NULL)
if ( !traceManager.init()) return(1);
traceManager.setTraceLevel("app:0");//trace by default
traceManager.setTraceLevel("api:0");//trace by default
if( !traceManager.processArgument(argc,argv))return(1)
//初始化应用框架
if(!appFramePtr->initialize()){
   appTrace<<VTMA_trace::info<<"Couldn't initialize the Agent
" <<endI;
  return(1);}
//激活应用框架
if(!appFramePtr->activate()){
    appTrace<<VTMA_trace::info<<"Fatal Error:-couldn't bring
up agent"<<endI;
   return(1);}
//以DMI的system对象初始化包含树
VTMA::M_CreateArgument Sysarg;
VTMA::InvokeId ref;
VTMA::M_CreateResult* sysRst+new VTMA::M_CreateResult;
createSystemArg(1,sysarg)
ref=createInvokeLd();//create invokeId
insertLocalinvokeId(ref,sysRst);
appframePtr->localMOCreate(sysarg,ref, *sysRst);
//create system object locally
//进入事件循环，处理输入事件
appFramePtr->ESMainLoop();
```

实现代理的主循环后，代理应用程序就可以接收并处理MIB中定义的 CMISE命令，这些命令由 App_Framework和App_ManagedObject对象具体处理。当然，由于我们的代理应用是基于内存的，没有实现 MIB与具体网络设备的后端接口，相应的操作只能对内存中的 MIB产生影响。通过进一步编写 MIB与设备的后端接口，我们的代理应用就可以管理具体的网络设备了。

3.2.4 生成代理应用

完成上述各步骤后，已经具备了生成代理所需的各种源代码，通过使用适当的C++编译器(目前的 Vertel平台使用Microsoft VC++5.0)，编译实现的代理应用类和主循环，并连接已经生成的类库 unixlib.lib，即可得到代理应用程序。

3.3 测试

通过对代理应用进行测试，检验它是否完成我们所要求的功能。测试一般使用管理者模拟器进行。使用工具命令语言(Tool command language, TCL) 编写测试用例在管理者模拟器上运行，即可检验信息模型及相应的操作和事件的正确性。限于篇幅，这里不再赘述。

4 结束语

本文简介了 Vertel的TMN平台，分析了基于ADE平台的代理开发过程，并以一个实例进行了具体说明。ADE平台提供的API接口和对象编译器、管理者模拟器等工具为我们的开发提供了良好的支持，大大提高了网管软件的开发速度和质量。

另一方面，上述的平台也存在一定的不足之处：(1) 对软件开发的前期，如需求获取、系统分析阶段缺乏支持，许多工作依赖于经验；(2) 现有的信息建模采用GDMO/ASN.1,所产生的信息模型为文本形式，可读性差；(3) 信息模型与具体的网管协议(CMIP)相关，难以为其它的管理协议如SNMP、CORBA等所重用。为此，我们在基于TMN平台的网管软件开发中，引入可视化面向对象建模语言UML及相应的支持工具Rose98，通过UML的用例图、交互图支持软件开发的需求获取和系统分析；通过UML的类图,构造独立于网管协议、可视化的信息模型。具体细节将另文说明。

参考文献

1 孔令萍.电信管理网 . 北京：人民邮电出版社，1997-12

2 Vertel Corp.TMN C++Agent Development Environment Getting Started . 1998-06

3 Vertel Corp .TMN C++ Agent for Microsoft Windows NT Application Developer's Guide . 1998-06

4 Vertel Corp. TMN Manager Simulator User Guide . 1998-06

☆☆

(上接第174页)

使用基于主机的IDS以弥补网络型IDS的不足。网络型IDS的数据源是网络上的数据包。主机型IDS往往以系统日志、应用程序日志等作为数据源，从所在的主机收集信息进行分析。网络型IDS的优点主要是简便，一个网段上只需安装一个或几个这样的系统，便可以监测整个网段的情况。由于可以分出单独的计算机作这种应用，不会给运行关键业务的主机带来负载上的增加。但由于现在网络的日趋复杂和高速网络的普及，这种结构正受到越来越大的挑战。面对高速网络IDS可能来不及处理数据包，在交换式网络中可能根本看不到其它主机收到的数据包。从前面针对网络型IDS的攻击的分析可以发现，许多攻击所利用的一点是：IDS处理数据包的具体做法和目的主机的做法不一定相同。通过主机型的IDS可以知道目的主机对某一些特定的数据是如何处理的，可以利用操作系统本身提供的功能并结合异常分析，更准确地报告攻击行为。

正确设置IDS的响应措施。有些IDS可以被设置为当发现攻击时能够自动改变路由器的设置以过滤掉攻击方的IP，从以上的分析得知，这样做实际上是弊大于利。

6 结束语

通过对网络型IDS存在的一些问题的讨论，可见有些是其本身无法克服的，实际上任何一种安全工具都有其不足的一面，我们只有综合应用各种安全措施，才能不断弥补已经存在的缺陷，使攻击发生的概率降到最低。

参考文献

1 Atkins D.Internet网络安全专业参考手册.北京机械工业出版社, 1999

2 Ptacek T H,Newsham T N.Insertion, Evasion,and Denial of Service: Eluding Network Intrusion Detection.Secure Networks,Inc.,1998-01

3 Next Generation Intrusion Detection in High Speed Network.from www.nai.com,2000

4 (美) Stevens W R.TCP/IP详解(第1 卷).北京: 机械工业出版社,2000

5 Network-vs.Host-based Intrusion Detection.from www.iss.net,2000