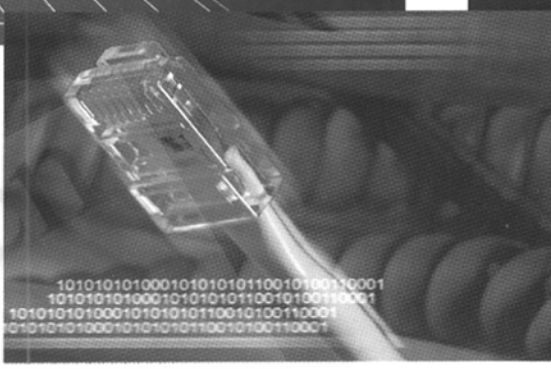
Data collection mechanisms for intrusion detection systems

IDS的数据收集机制



叶惠敏　潘正运　(郑州　解放军信息工程大学电子技术学院　450004)

摘要：本文首先阐述了数据收集在入侵检测中的重要性，然后对目前的数据收集机制进行了分类，并对每一类中的不同方法的利弊进行了比较和讨论。

关键词：入侵检测　数据收集　直接(间接)监控　探测器

1　IDS简介

自从1980年James Anderson提出入侵检测的概念以来，在过去的20年时间内，入侵检测系统(IDS)得到了飞速的发展，其相应的研究原型和商业化产品也大量涌现。

IDS的目的是要保障所监控计算机系统的安全，它可以根据配置及时发现并报告系统中未经授权或异常的现象，是一种用于检测计算机网络中违反安全策略行为的技术。

一般情况下，人们往往根据审计数据来源的不同将IDS分为两类：基于主机的IDS和基于网络的IDS。基于主机的IDS根据从所监控的主机上获取的信息进行决策，而基于网络的IDS则通过监视网络中的数据流获得决策信息，这个分类方法是从审计数据来源的角度出发的，并没有考虑数据在哪里以及如何被处理，所以，为了使读者对此分类方法有一个更清晰、准确的理解，我们将其称为基于主机数据收集的IDS和基于网络数据收集的IDS。

根据IDS从哪里以及如何处理数据，我们可以将其分成分布式和集中式两种，前者在多台主机上完成数据收集和分析工作，而后者的数据收集可能是分布的，也可能是集中的，但数据的分析处理则集中在某台主机上进行。这两种系统既可以使用基于主机的数据收集机制，也可以使用基于网络的数据收集机制，或者将两者结合使用。

数据收集机制研究的重要性是显而易见的：就准确性、可靠性和效率而言，IDS收集到的数据是它进行检测和决策的基础。如果收集数据的时延太大，系统很可能在检测到攻击的时候，造成的后果已经无法挽回了。如果数据不完整，系统的检测能力就会大打折扣。如果数据本身不正确(由于错误或入侵者的行为)，系统就无法检测某些攻击，从而给用户造成一种很虚假的安全感，后果更不堪设想。本文描述入侵检测数据收集机制的不同分类方法，并对每一种方法的利弊进行讨论。

2　基于网络的数据收集和基于主机的数据收集

入侵检测系统目前所能检测到的大部分入侵都是由主机上的活动引起的，如执行某一命令、访问某项服务并提供不正确的数据等，这些攻击活动发生在终端主机上，有时通过网络检测也可以发现它们。

针对网络本身的攻击通常都是数据洪流，即发送的数据量超出网络的接收能力，以至于合法数据包通信受阻，但在终端主机上也可以检测到这些攻击，例如，通过查看主机ICMP层是否有大量的ECHO-REQUEST分组，也可以检测到是否有Ping洪流攻击发生。

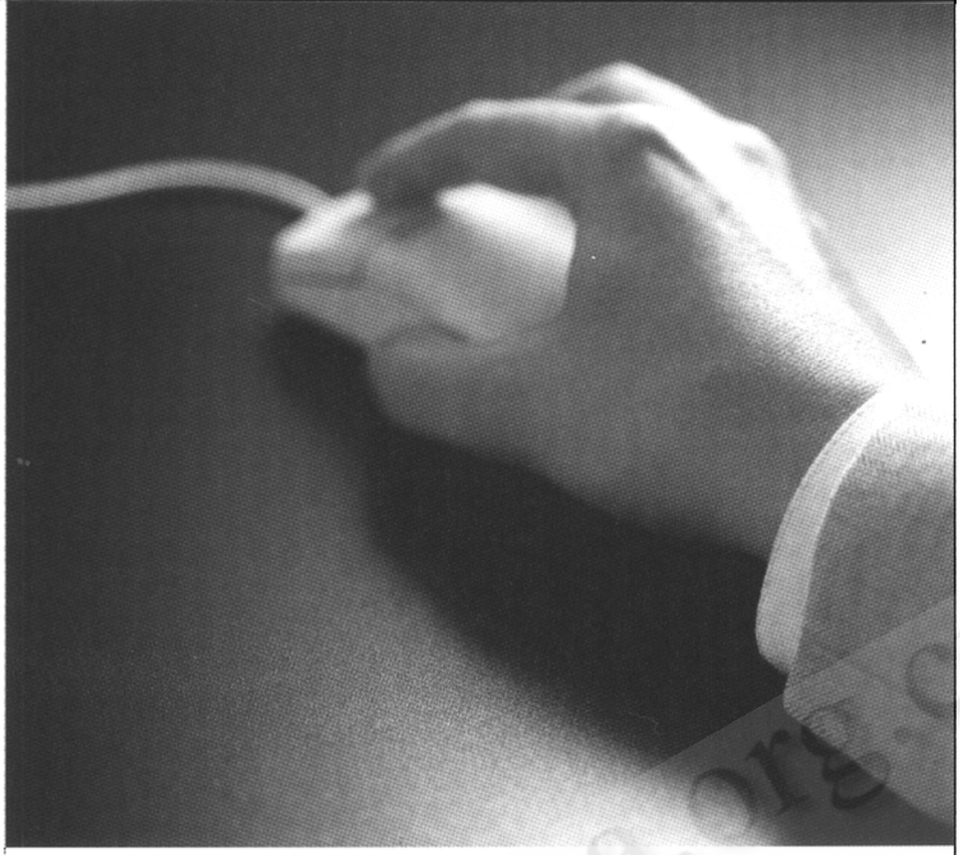
基于网络的数据收集有时还会比基于主机的数据收集效果更好，尤其是主机对攻击行为没有反应的时候(例如，分组攻击的端口是关闭的)，但即使在这种情况下，也可以通过终端主机网络栈的低层检测到这些攻击。

总体而言，基于主机的数据收集要好一些，原因如下：

(1) 基于主机收集到的数据能准确反映主机上发生的情况，它不是根据从网络上收集到的数据包去猜测发生了什么；

(2) 在数据流量很大的网络中，网络监视器经常会丢包，但主机监视器就可以报告每台主机上发生的所有事件；

(3) 基于网络的数据收集机制对插入攻击和规避攻击无能为力，但基于主机的数据收集就不存在这样的问题，它能够处理主机收到的所有数据。



这些问题也从更一般的意义上反映出了直接数据收集和非直接数据收集方法的差异。

3 直接监控和间接监控

根据下面的定义，我们将数据收集分成直接和间接两种方法：

(1) 直接监控：从数据生成地或属地直接获取数据，例如，如果要直接监控某主机的CPU负载情况，我们就需要从主机相应的内核结构中获取数据；如果要直接监控inetd后台进程所提供的网络服务的访问情况，我们就需要直接从inetd获取关于那些访问情况的数据。

(2) 间接监控：从能反映监控目标行为的数据源处获取数据。还以前面的两个例子为证，可以通过读取记录CPU负载的日志文件，完成对主机CPU负载的间接监控；通过读取inetd后台进程所产生的日志文件，或通过类似TCP-Wrappers的辅助程序，间接监控网络服务的访问情况；也可以通过监视发往主机特定端口的数据包进行间接监控。

就检测入侵行为而言，直接监控优于间接监控，原因如下：

① 从非直接数据源获取的数据（例如审计跟踪）在被IDS使用之前，有被入侵者修改的潜在可能。

② 非直接数据源可能无法记录某些事件，例如，并不是inetd后台进程的所有行为都会被记录到日志文件中的，还有一点，间接数据源可能无法访问监视对象的内部信息，例如，TCP-Wrappers不能检查inetd后台进程的内部操作，只能检查那些通过外部接口传递的数据。

③ 在间接监控中，数据一般都是通过某种机制生成的（如编写审计踪迹的代码），但那些机制并不了解IDS使用数据的具体需求，也正因如此，从间接数据源获取的数据量总是非常之大，Kumar和Spafford [3] 曾提到，一个C2生成的审计踪迹可能包含每个用户每天50K—500K的记录，Mounji [4] 也指出，对于一个中等规模的用户组来说，每天审计踪迹数据就会有好几百兆。

由于这个原因，IDS在使用间接数据源时，通常必须消耗大量的资源对数据进行过滤和精简。

直接监控方法只获取它需要的数据，所以生成的数据量就相对较小，此外，监控部件自身会对数据进行分析，只有在检测到相关事件时才产生结果，这样就减少了数据的存储量（除非要存储数据以用于事后的进一步调查）。

④ 间接监控机制的伸缩性差，因为当主机及其内部被监控要素的数目增加时，过滤数据的开销会降低被监控主机的性能。

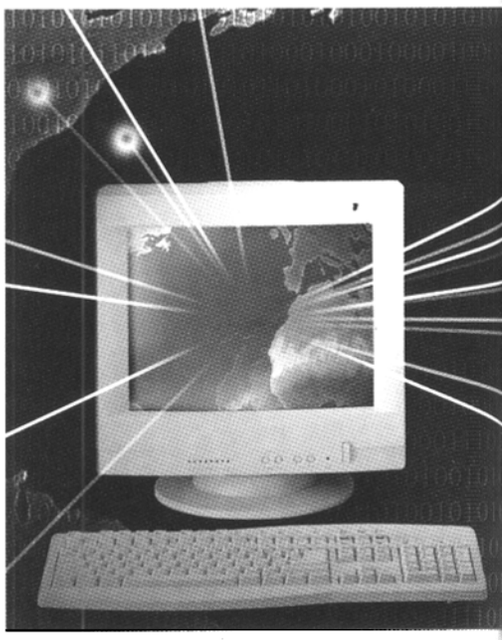
⑤ 间接数据源常常在数据产生和IDS访问这些数据之间有一个时延，而直接监控的时延就小的多，这样IDS才能据此做出更及时的反应。

4 外部探测器和内部探测器

外部探测器和内部探测器在用于直接数据收集时各有利弊，可以综合使用。表1列出了这两种类型探测器各自的优缺点。从软件工程的角度来看，内部探测器和外部探测器在以下几方面具有不同的特点：

(1) 错误引入：当使用内部探测器时，必须修改被监视程序的代码，所以程序操作中更容易引入错误。外部探测器也会引入错误（例如，代理消耗了过量资源，或库调用错误地修改了某些参数），但大部分内部探测器都是很短的代码，检查错误难度也不大。

(2) 可维护性：外部探测器与所监控的程序



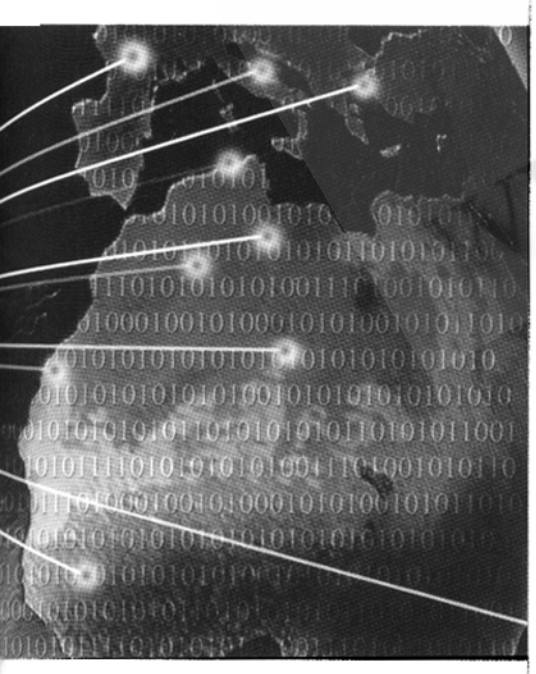
是相互分离的，所以也更容易维护。

(3) 规模：内部探测器是现成程序的一部分，所以避免了与创建进程相关的一些基本开销，因而可以比外部探测器要小。

(4) 完备性：内部探测器可以访问所监视程序中的任何信息，而外部探测器只能访问外部可获的数据，所以，内部探测器能获得关于监视程序行为更完备的信息。此外，内部探测器可以被放置在所监视程序的任何地方，而外部探测器只能"从外面"检查程序，所以内部探测器比外部探测器的观察范围也要广。

(5) 正确性：内部探测器访问的数据更完全，而外部探测器只能根据可获数据作出基于经验的猜测，所以内部探测器产生的结果也比外部探测器更正确。

通过对这两种探测器的全面比较，我们可以看到，外部探测器在易用性和可维护性方面较好，而内部探测器在监控和检测能力、以及适应性和对主机的影响方面具有明显的优势，可以根据特定的任务，综合使用这两种探测器，可扬长避短。



5 结束语

本文将IDS的数据收集技术分为以下三类：

(1) 基于主机的和基于网络的数据收集技术。我们阐述了基于主机的数据收集技术优于基于网络的数据收集技术的原因，还讨论少数几个特殊情况。

(2) 直接和间接数据收集技术。我们讨论了直接数据收集任何情况下都要优于间接数据收集技术。

(3) 可以使用外部探测器或内部探测器进行数据收集。我们对这两种探测器进行了比较，发现外部探测器易用、易实现，但内部探测器更实用、更可靠。对于IDS的设计者和实现者来说，研究所使用的数据源、权衡使用这些数据源的利弊以及是否可能有更好的方法，都是至关重要的。只有当决策数据可靠时，IDS的可靠性才能得到保证，所以我们最关心是如何为入侵检测系统提供尽可能好的数据源。

表 两种探测器比较

	外部探测器	内部探测器
优点	从主机上进行修改、添加或删除等操作很容易；可利用任何合适的编程语言实现。	在产生数据和使用数据之间的时延最小；不是分离的进程，所以不易被禁止或修改；对主机的性能开销比较小；最后实现方式是所监视程序的一部分，所以可以访问任务所必须的任何信息。
缺点	有被入侵者禁止或修改的潜在可能；在数据生成和使用之间存在时延；探测器是分离的进程或额外加载的库，所以存在性能影响；依靠系统或某些机制提供的信息，所以信息获取能力有限。	需要集成到所监视的程序中去，所以实现难度较大；实现时，需要使用与被监视程序相同的语言；如果设计或实现不当，就会严重地损坏被监视程序的性能和功能；升级或修改的难度大。

参考文献

1 Katherine E. Price. Host-based misuse detection and conventional operating systems audit data collection. Master thesis, Purdue University, December 1997.

2 Wietse Venema.TCP WRAPPER: Network monitoring, access control and booby traps. In USENIX Associaton,editor, UNIX Security III symposium, September 14-17,1992. Baltimore.

3 Sandeep Kumar and Eugene H.Spafford. A software architecture to support misuse intrusion detection. In Proceedings of the 18th Natonal Information Systems Security Conference.

4 Abdelaziz Mounji. Languages and Tools for Rule-Based Distributed Intrusion Detection. D.Sc. thesis, Facultes Universitaires, Notre-Dame de la Paix, Namur (Belgium), September 1997.