第6章 入侵检测技术

中国科学技术大学
曾凡平
billzeng@ustc.edu.cn

主要内容

6.1 入侵检测概述

- 6.1.1 入侵检测的概念及模型
- 6.1.2 IDS的任务
- 6.1.3 IDS提供的主要功能
- 6.1.4 IDS的分类

6.2 CIDF模型及入侵检测原理

- 6.2.1 CIDF模型
- 6.2.2 入侵检测原理

6.3 基于Snort部署IDS

6.4 NIDS的脆弱性及反NIDS技术

6.1 入侵检测概述

- **入侵检测**(Intrusion Detection)源于传统的**系统审计**，从1980年代初期提出的理论雏形到实现商品化的今天已经走过了近四十年的历史。

- 作为一项**主动的网络安全技术**，它能够检测未授权对象（用户或进程）针对系统（主机或网络）的入侵行为，监控授权对象对系统资源的非法使用，记录并保存相关行为的法律证据，并可根据配置的要求在特定的情况下采取必要的响应措施（警报、驱除入侵、防卫反击等）。

6.1.1 入侵检测的概念及模型

- **入侵**就是试图破坏网络及信息系统机密性、完整性和可用性的行为。入侵方式一般有：
  - (1) 未授权的用户访问系统资源；
  - (2) 已经授权的用户企图获得更高权限，或者是已经授权的用户滥用所给定的权限等。
- **入侵检测的概念：入侵检测是监测计算机网络和系统、发现违反安全策略事件的过程。**
- 美国国家安全通信委员会(NSTAC)下属的入侵检测小组(IDSG)在1997年给出的关于“入侵检测”(Intrusion Detection)的定义是：**入侵检测是对企图入侵、正在进行的入侵或已经发生的入侵行为进行识别的过程。**

“入侵检测”的3种常见的定义

- (1) 检测对计算机系统的非授权访问。
- (2) 对系统的运行状态进行监视，发现各种攻击企图、攻击行为或攻击结果，以保证系统资源的保密性、完整性和可用性。
- (3) 识别针对计算机系统和网络系统、或广义上的信息系统的非法攻击，包括检测外部非法入侵者的恶意攻击或探测，以及内部合法用户越权使用系统资源的非法行为。

入侵检测系统(IDS)

- 所有能够执行入侵检测任务和实现入侵检测功能的系统都可称为**入侵检测系统**(IDS，Intrusion Detection System)，其中包括软件系统或软/硬件结合的系统。入侵检测系统自动监视出现在计算机或网络系统中的事件，并分析这些事件，以判断是否有入侵事件的发生。

- 入侵检测系统一般位于内部网的入口处，安装在防火墙的后面，用于检测外部入侵者的入侵和内部用户的非法活动。

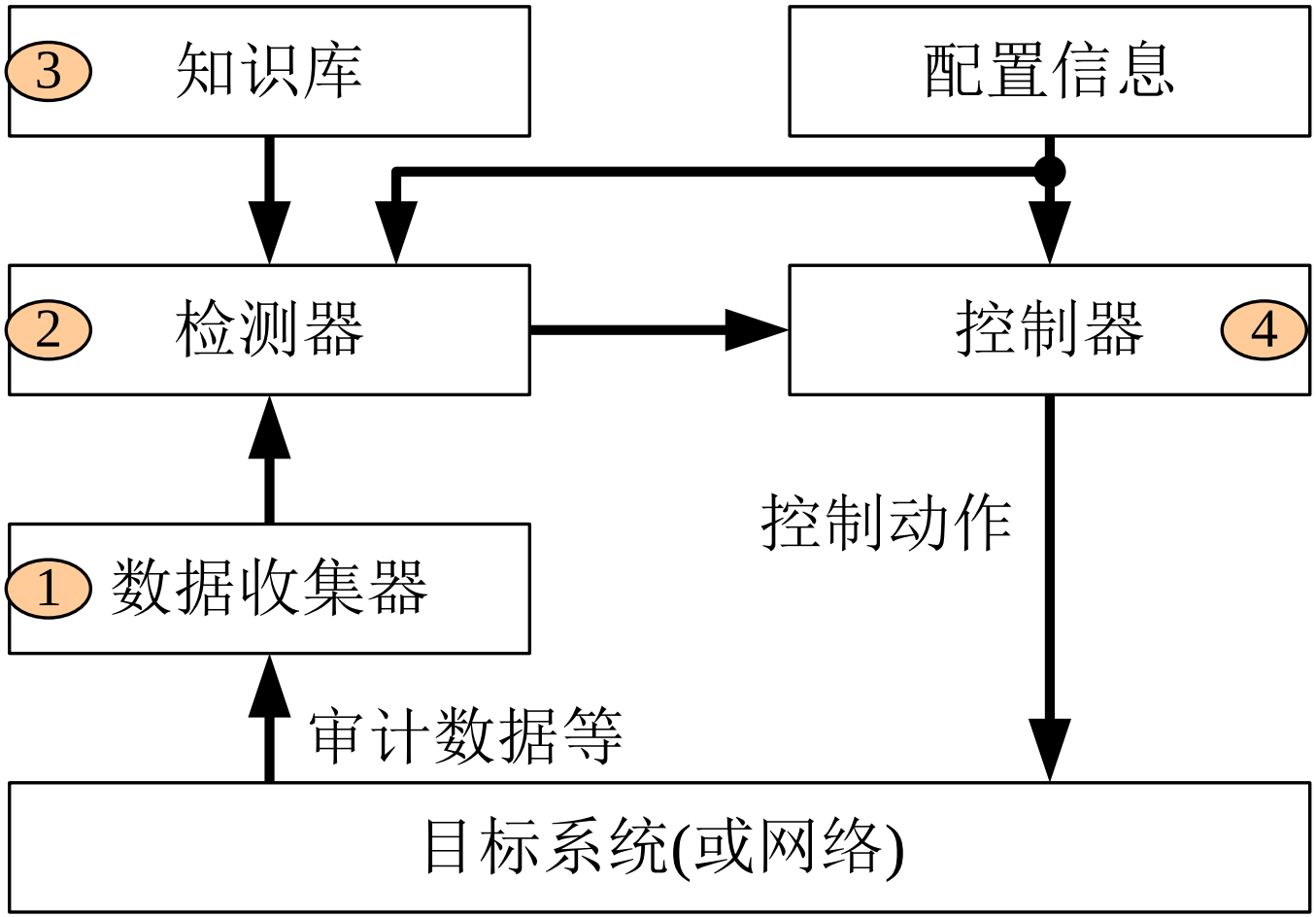


图 6-1 入侵检测系统

- (1) **数据收集器**：又称**探测器**，主要负责收集数据。收集器的输入数据包括任何可能包含入侵行为线索的数据，如各种网络协议数据包、系统日志文件和系统调用记录等。探测器将这些数据收集起来，然后再发送到检测器进行处理。
- (2) **检测器**：又称分析器或检测引擎，负责分析和检测入侵的任务，并向控制器发出警报信号。
- (3) **知识库**：为检测器和控制器提供必需的信息支持。这些信息包括：用户或系统的历史活动档案或检测规则集合等。
- (4) **控制器**：也称为响应器，根据从检测器发来的警报信号，人工或自动地对入侵行为做出响应。
- 此外，大多数入侵检测系统都会包含一个**用户接口**组件，用于观察系统的运行状态和输出信号，并对系统的行为进行控制。

6.1.2 IDS的任务

- **(1) 信息收集**
- IDS的第一项任务是信息收集。IDS所收集的信息包括用户(合法用户和非法用户)在网络、系统、数据库及应用程序活动的状态和行为。为了准确地收集用户的信息活动，需要在信息系统中的若干个关键点(包括不同网段、不同主机、不同数据库服务器、不同的应用服务器等处)设置信息探测点。

- IDS可利用的信息来源如下：

1) 系统和网络的日志文件

- 日志文件中包含发生在系统和网络上异常活动的证据，通过查看日志文件，能够发现黑客的入侵行为。

2) 目录和文件中的异常改变

- 信息系统中的目录和文件中的异常改变(包括修改、创建和删除)，特别是那些限制访问的重要文件和数据的改变，很可能就是一种入侵行为。黑客入侵目标系统后，经常替换目标系统上的文件，替换系统程序或修改系统日志文件，达到隐藏其活动痕迹的目的。

3) 程序执行中的异常行为

- 每个进程在具有不同权限的环境中执行，这种环境控制着进程可访问的系统资源、程序和数据文件等。一个进程出现了异常的行为，可能表明黑客正在入侵系统。

4) 网络活动信息

- 远程攻击主要通过网络发送异常数据包而实现，为此IDS需要收集TCP连接的状态信息以及网络上传输的实时数据。比如，如果收集到大量的TCP半开连接，则可能是拒绝服务攻击的开始。又比如，如果在短时间内有大量的到不同TCP（或UDP）端口的连接，则很可能说明有人在对己方的网络进行端口扫描。

(2) 信息分析

- 对收集到的网络、系统、数据及用户活动的状态和行为信息等进行模式匹配、统计分析和完整性分析，得到实时检测所必需的信息。

1) 模式匹配

- 将收集到的信息与已知的网络入侵模式的特征数据库进行比较，从而发现违背安全策略的行为。假定所有入侵行为和手段(及其变种)都能够表达为一种模式或特征，那么所有已知的入侵方法都可以用匹配的方法来发现。模式匹配的关键是如何表达入侵模式，把入侵行为与正常行为区分开来。模式匹配的优点是误报率小，其局限性是只能发现已知攻击，而对未知攻击无能为力。

2) 统计分析

- 统计分析是入侵检测常用的**异常发现**方法。假定所有入侵行为都与正常行为不同，如果能建立系统正常运行的行为轨迹，那么就可以把所有与正常轨迹不同的系统状态视为可疑的入侵企图。统计分析方法就是先创建系统对象(如用户、文件、目录和设备等)的统计属性(如访问次数、操作失败次数、访问地点、访问时间、访问延时等)，再将信息系统的实际行为与统计属性进行比较。当观察值在正常值范围之外时，则认为有入侵行为发生。

3) 完整性分析

- 完整性分析检测某个文件或对象是否被更改。完整性分析常利用消息杂凑函数(如 MD5和SHA)，能识别目标的微小变化。

- 该方法的优点是某个文件或对象发生的任何一点改变都能够被发现。

(3) 安全响应

- IDS在发现入侵行为后必然及时做出响应，包括终止网络服务、记录事件日志、报警和阻断等。
- 响应可分为**主动响应**和**被动响应**两种类型。**主动响应**由用户驱动或系统本身自动执行，可对入侵行为采取终止网络连接、改变系统环境(如修改防火墙的安全策略)等；**被动响应**包括发出告警信息和通知等。目前比较流行的响应方式有：记录日志、实时显示、E-mail报警、声音报警、SNMP报警、实时TCP阻断、防火墙联动、手机短信报警等。

6.1.3 IDS提供的主要功能

- 为了完成入侵监测任务，IDS需要提供以下主要功能。

(1) 网络流量的跟踪与分析功能：跟踪用户进出网络的所有活动，实时检测并分析用户在系统中的活动状态；实时统计网络流量，检测拒绝服务攻击等异常行为。

(2) 已知攻击特征的识别功能：识别特定类型的攻击，并向控制台报警，为网络防护提供依据。根据定制的条件过滤重复告警事件，减轻传输与响应的压力。

(3) 异常行为的分析、统计与响应功能：分析系统的异常行为模式，统计异常行为，并对异常行为做出响应。

(4) 特征库的在线和离线升级功能：提供入侵检测规则的在线和离线升级，实时更新入侵特征库，不断提高IDS的入侵检测能力。

(5) 数据文件的完整性检查功能：检查关键数据文件的完整性，识别并报告数据文件的改动情况。

(6) 自定义的响应功能：定制实时响应策略；根据用户定义，经过系统过滤，对告警事件及时响应。

(7) 系统漏洞的预报警功能：对未发现的系统漏洞特征进行预报警。

(8) IDS探测器集中管理功能：通过控制台收集探测器的状态和告警信息，控制各个探测器的行为。

- 一个高质量的IDS产品除了具备以上入侵检测功能外，还必须容易配置和管理，并且自身具有很高的安全性。

6.1.4 IDS的分类

(1) 基于网络的入侵检测系统(NIDS，Network Intrusion Detection System)

- 数据来自网络上的数据流。NIDS能够截获网络中的数据包，提取其特征并与知识库中已知的**攻击签名**相比较，从而达到检测目的。
- 其优点是检测速度快、隐蔽性好、不容易受到攻击、不消耗被保护主机的资源；缺点是有些攻击是从被保护的主机发出的，不经过网络，因而无法识别。

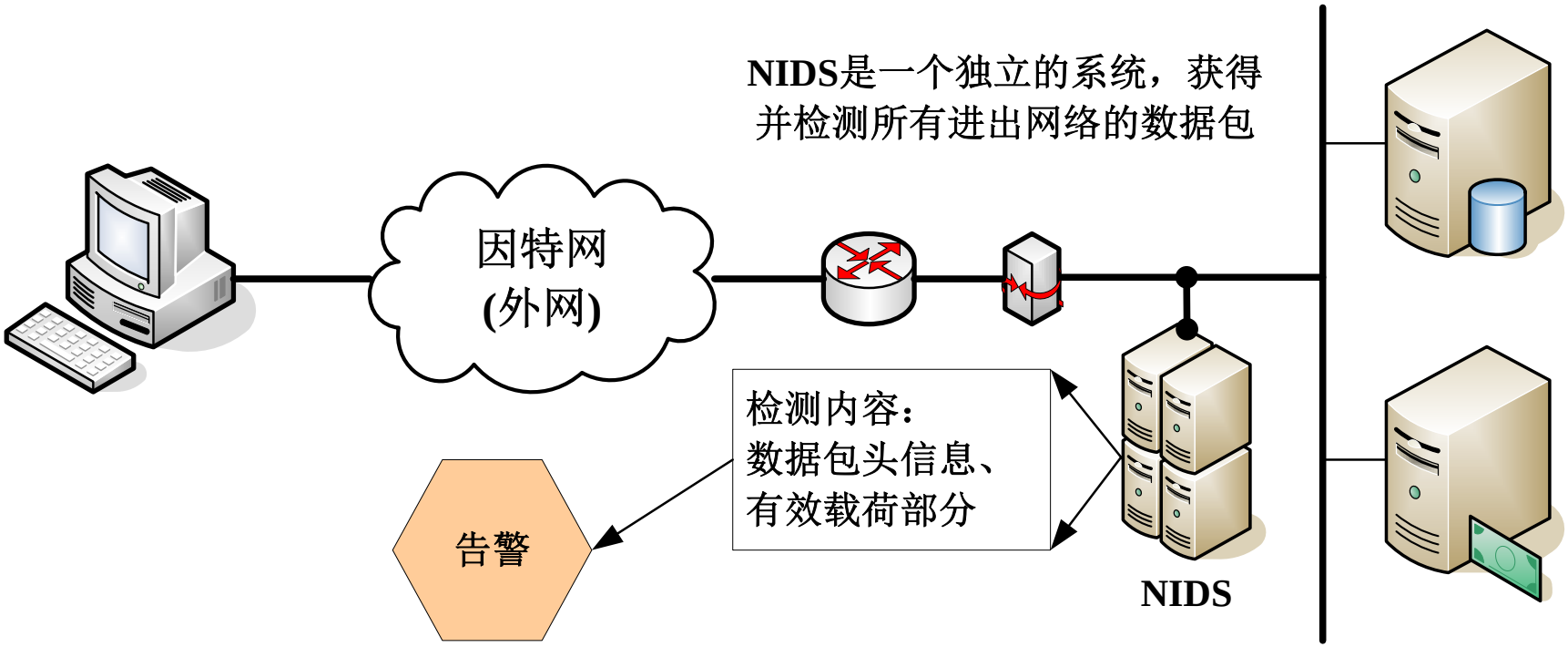


图 6-2 基于网络的入侵检测系统

(2) 基于主机的入侵检测系统(HIDS，Host Intrusion Detection System)

- 数据来源于主机系统，通常是系统日志和审计记录。HIDS通过对系统日志和审计记录的不断监控和分析来发现入侵。
- 优点是针对不同操作系统捕获应用层入侵，误报少；缺点是依赖于主机及其子系统，实时性差。HIDS通常安装在被保护的主机上，主要对该主机的网络实时连接及系统审计日志进行分析和检查，在发现可疑行为和安全违规事件时，向管理员报警，以便采取措施。

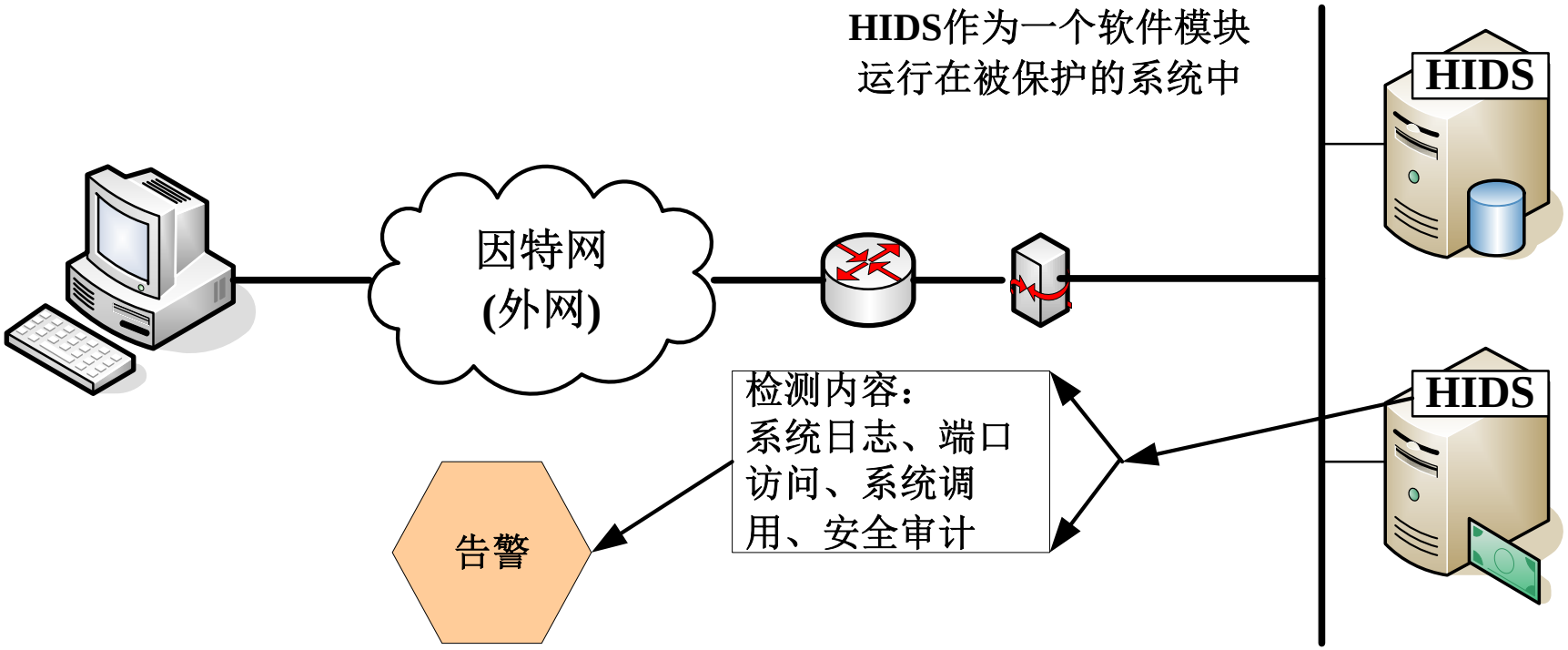


图 6-3 基于主机的入侵检测系统

LIDS：基于Linux内核的入侵检测系统。

- 这是一种基于Linux内核的入侵检测系统。它在Linux内核中实现了参考监听模式以及命令进入控制(Mandatory Access Control)模式，可以实时监视操作状态，旨在从系统核心加强其安全性。在某种程度上可以认为它的检测数据来源于操作系统的内核操作，在这一级别上检测入侵和非法活动，因此其安全特性要高于其他两类IDS。

(3) 分布式入侵检测系统 (DIDS， Distributed Intrusion Detection System)

- 采用上述两种数据来源。这种系统能够同时分析来自主机系统的审计日志和来自网络的数据流，一般为分布式结构，由多个部件组成。DIDS可以从多个主机获取数据，也可以从网络取得数据，克服了单一的HIDS和NIDS的不足。
- 典型的DIDS采用控制台/探测器结构。NIDS和HIDS作为探测器放置在网络的关键节点，并向中央控制台汇报情况。攻击日志定时传送到控制台，并保存到中央数据库中，新的攻击特征能及时发送到各个探测器上。每个探测器能够根据所在网络的实际需要配置不同的规则集。

6.2 CIDF模型及入侵检测原理

6.2.1 CIDF模型

- 由于大多数的入侵检测系统都是独立开发的，不同系统之间缺乏互操作性和互用性，这对入侵检测系统的发展造成了障碍，因此，DARPA（the Defense Advanced Research Prouects Agency，美国国防部高级研究计划局）在1997年3月开始着手通用入侵检测架构（CIDF, Common Intrusion Detection Framework）标准的制定。
- CIDF 是一种推荐的入侵检测标准架构。

- CIDF由S.Staniford等人提出，主要有三个目的：

1. IDS构件共享，即一个IDS系统的构件可被另一个系统使用；
2. 数据共享，即通过提供标准的数据格式，使得IDS中的各类数据可以在不同的系统之间传递并共享；
3. 完善互用性标准，并建立一套开发接口和支持工具，以提供独立开发部分构件的能力。

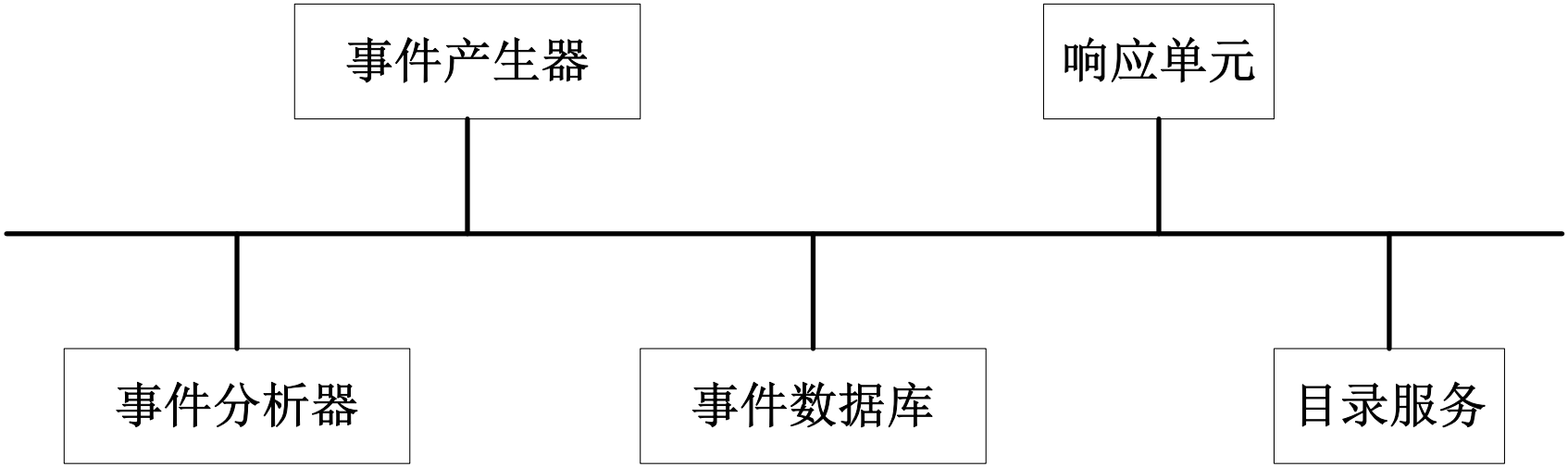


图 6-4 CIDF框架结构图

CIDF模型将入侵检测需要分析的数据称作事件（Event），它可以是基于网络的入侵检测系统的数据包，也可以是基于主机的入侵检测系统从系统日志等其它途径得到的信息。模型也对各个部件之间的信息传递格式、通信方法和API进行了标准化。

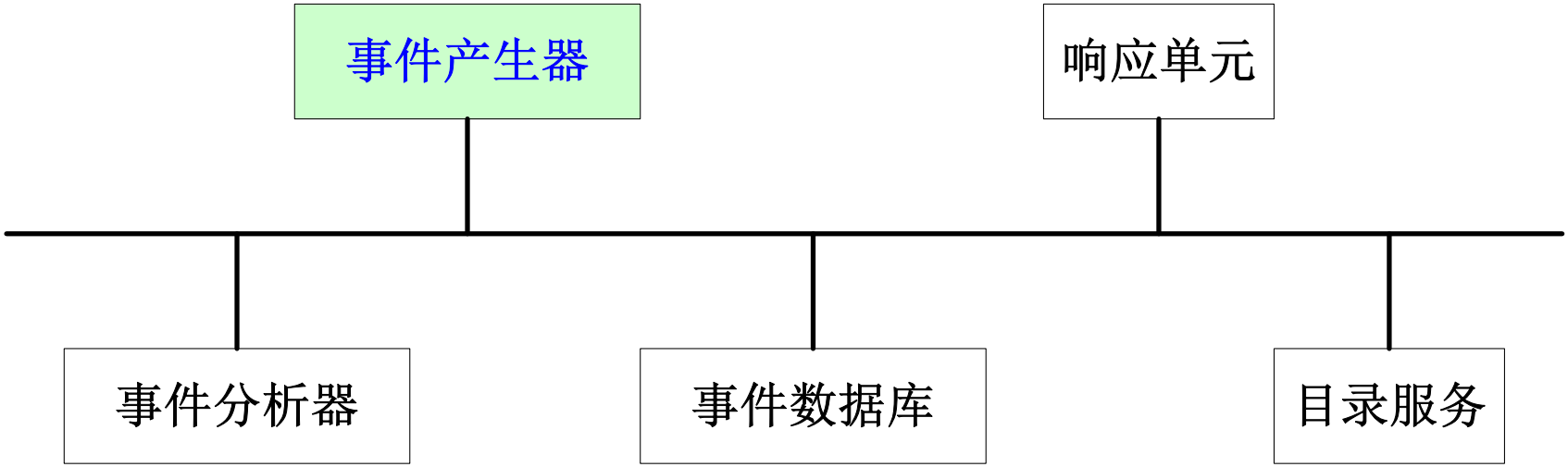


图 6-4 CIDF框架结构图

事件产生器的目的是从整个的计算机环境（也称为信息源）中获得事件，并向系统的其他部分提供该事件，这些数据源可以是网络、主机或应用系统中的信息。

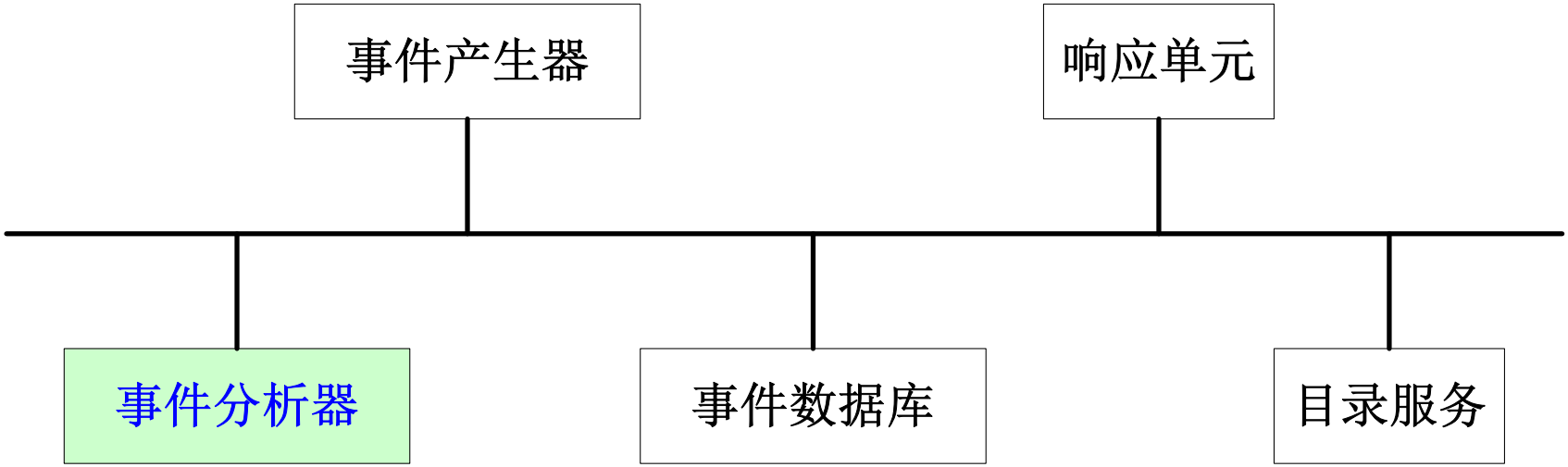


图 6-4 CIDF框架结构图

事件分析器从事件产生器中获得数据，通过各种分析方法——一般为误用检测和异常检测方法——来分析数据，决定入侵是否已经发生或者正在发生，在这里分析方法的选择是一项非常重要的工作。

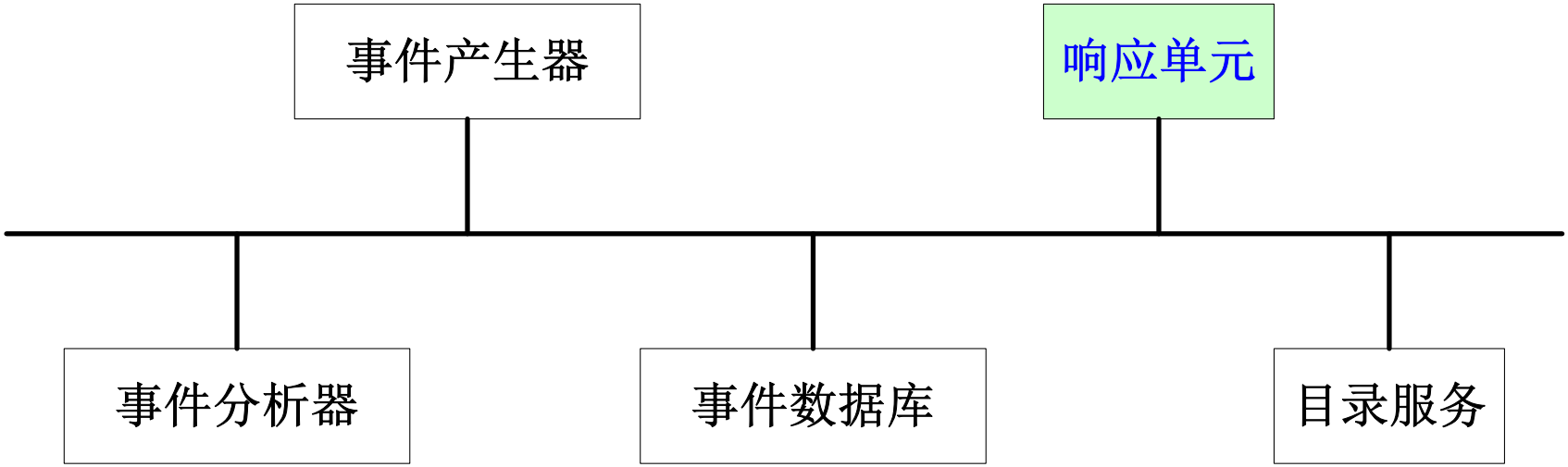


图 6-4 CIDF框架结构图

响应单元则是对分析结果作出反应的功能单元。最简单的响应是报警，通知管理者入侵事件的发生，由管理者决定采取应对的措施。

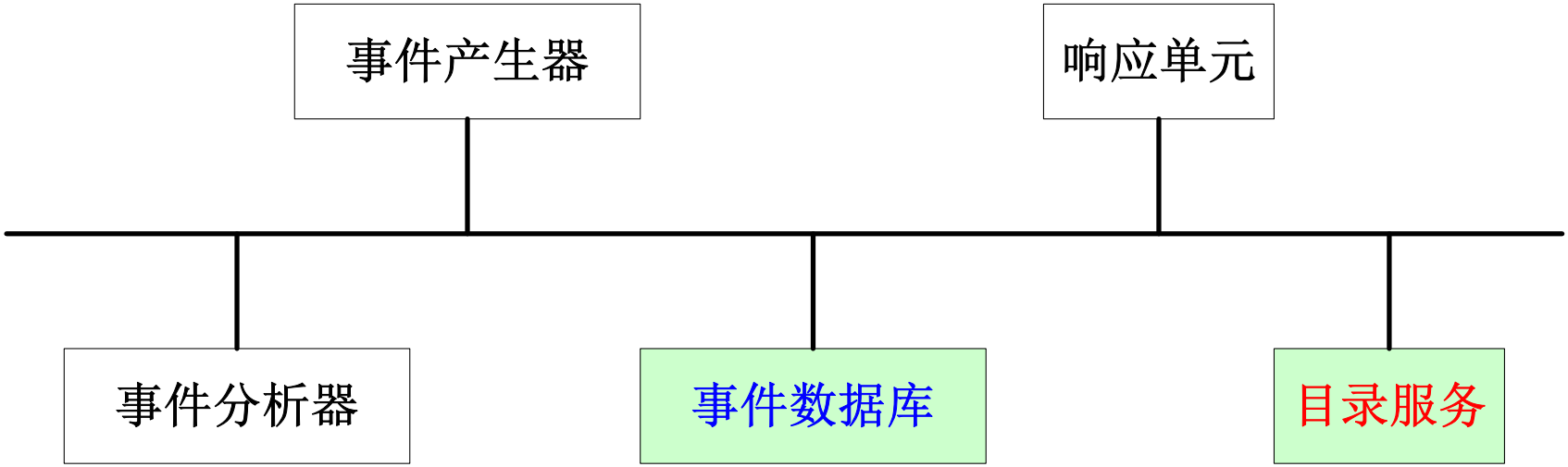


图 6-4 CIDF框架结构图

事件数据库是存放各种中间和最终数据的地方的总称，它可以是复杂的数据库，也可以是简单的文本文件。

目录服务构件用于各构件定位其他的构件，以及控制其他构件传递的数据并认证其他构件的使用，以防止IDS系统本身受到攻击。它可以管理和发布密钥，提供构件信息和告诉用户构件的功能接口。

入侵检测系统的处理模式

- 在目前的入侵检测系统中，经常用信息源、分析部件和响应部件来分别代替事件产生器、事件分析器和响应单元等术语。因此，人们往往将**信息源、分析和响应**称作**入侵检测系统的处理模式**。
- 虽然CIDF具有明显的优点，但实际上由于目前数据交换标准还在制定之中，因此它还没有得到广泛地应用，也没有一个入侵检测系统产品完全使用该标准，但未来的IDS系统将可能遵循CIDF标准。

6.2.2 入侵检测原理

- **事件分析器**也称为**分析引擎**，是入侵检测系统中最重要的核心部件，其性能直接决定IDS的优劣。

- IDS的**分析引擎**通常使用两种基本的分析方法来分析事件、检测入侵行为，即**误用检测**(MD，Misuse Detection)和**异常检测**(AD，Anomaly Detection)。

(1) 误用检测

- 误用检测技术又称基于知识的检测技术。它假定所有入侵行为和手段(及其变种)都能够表达为一种模式或特征，并对已知的入侵行为和手段进行分析，提取入侵特征，构建攻击模式或攻击签名，通过系统当前状态与攻击模式或攻击签名的匹配判断入侵行为。误用检测是最成熟、应用最广泛的技术。其工作模型如图6-5所示。
- 误用检测技术的优点在于可以准确地检测已知的入侵行为，缺点是不能检测未知的入侵行为。误用检测的关键在于如何表达入侵行为，即攻击模型的构建，把真正的入侵与正常行为区分开来。

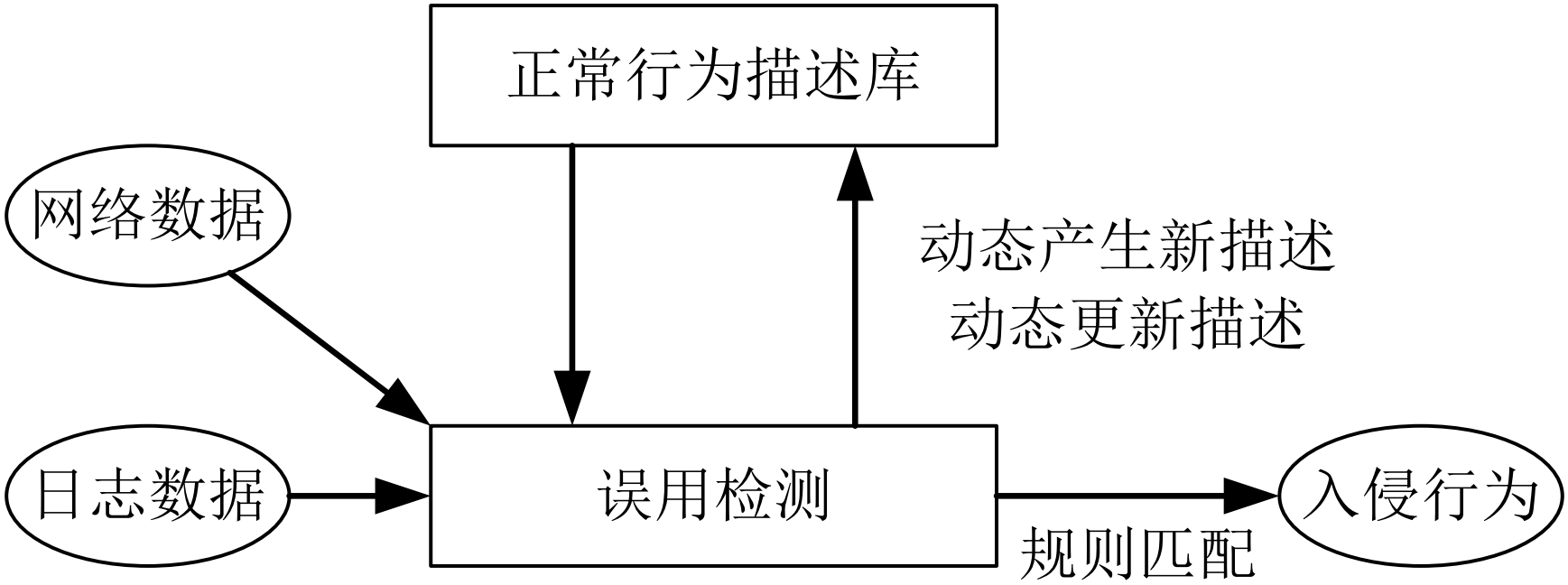


图 6-5 误用检测模型

(2) 异常检测

- 异常检测技术又称为**基于行为**的入侵检测技术，用来检测系统（主机或网络）中的异常行为。其基本设想是入侵行为与正常的(合法的)活动有明显的差异，即正常行为与异常行为有明显的差异。

- **异常检测的工作原理：**首先收集一段时间系统活动的历史数据，再建立代表主机、用户或网络连接的正常行为描述，然后收集事件数据并使用一些不同的方法来决定所检测到的事件活动是否偏离了正常行为模式，从而判断是否发生了入侵。

异常检测方法

- 基于异常检测原理的入侵检测方法有以下几种：
  - **(1) 统计异常检测方法；**
  - **(2) 特征选择异常检测方法：**
  - (3) 基于贝叶斯推理异常检测方法：
  - (4) 基于贝叶斯网络异常检测方法；
  - (5) 基于模式预测异常检测方法。
- 其中，比较成熟的方法是**统计异常检测方法和特征选择异常检测方法**。目前，已经有根据这两种方法开发而成的软件产品面市，其他方法目前还停留在理论研究阶段。

6.3 基于Snort部署IDS

- 在网络中部署IDS时，可以使用多个NIDS和HIDS，这要根据网络的实际情况和自己的需求而定。图6-6是一个典型的IDS的部署图。各机构可以根据自身特点选用其中的一部分IDS。对于实验室，一般只需在服务器上部署HIDS就可以了。
- Snort是一个免费的网络入侵检测系统，它是用C语言编写的开源软件。其作者Martin Roesch在设计之初，只打算实现一个数据包嗅探器，之后又在其中加入了基于特征分析的功能，从此Snort开始向入侵检测系统演变。现在的Snort已经发展得非常强大，拥有核心开发团队和官方站点(www.snort.org)。

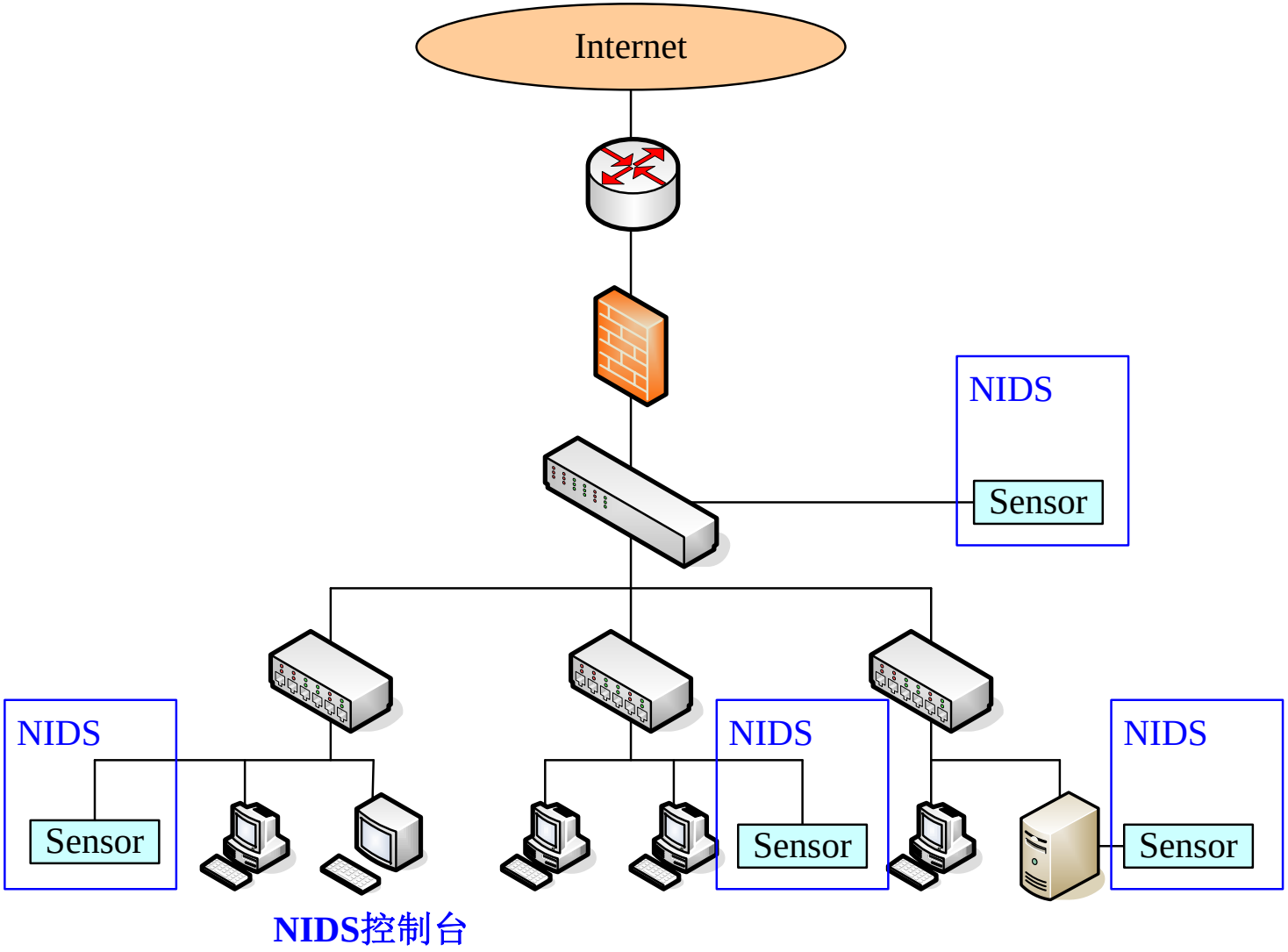


图 6-6 典型的IDS的部署图

- Snort是一个基于libpcap的轻量级网络入侵检测系统。所谓轻量级入侵检测系统，是指它能够方便地安装和配置在网络中任何一个节点上，而且不会对网络产生太大的影响。它对系统的配置要求比较低，可支持多种操作平台，包括Linux、Windows、Solaris和FreeBSD等。在各种NIDS产品中，Snort是其中最好的之一。不仅因为它是免费的，还因为它本身提供了如下各种强大的功能：
- (1) 基于规则的检测引擎。
- (2) 良好的可扩展性。可以使用预处理器和输出插件来对Snort的功能进行扩展。
- (3) 灵活简单的规则描述语言。只要用户掌握了基本的TCP、IP知识，就可以编写自己的规则。
- (4) 除了用作入侵检测系统，还可以用作嗅探器和包记录器。

- 一个基于Snort的网络入侵检测系统由以下5个部分组成：
- 解码器；预处理器；检测引擎；输出插件；日志/警报子系统

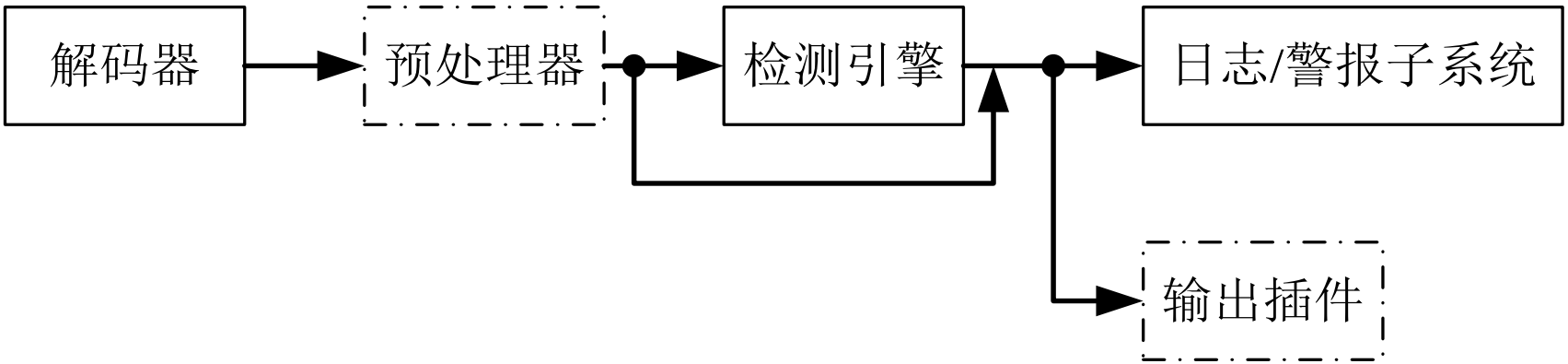
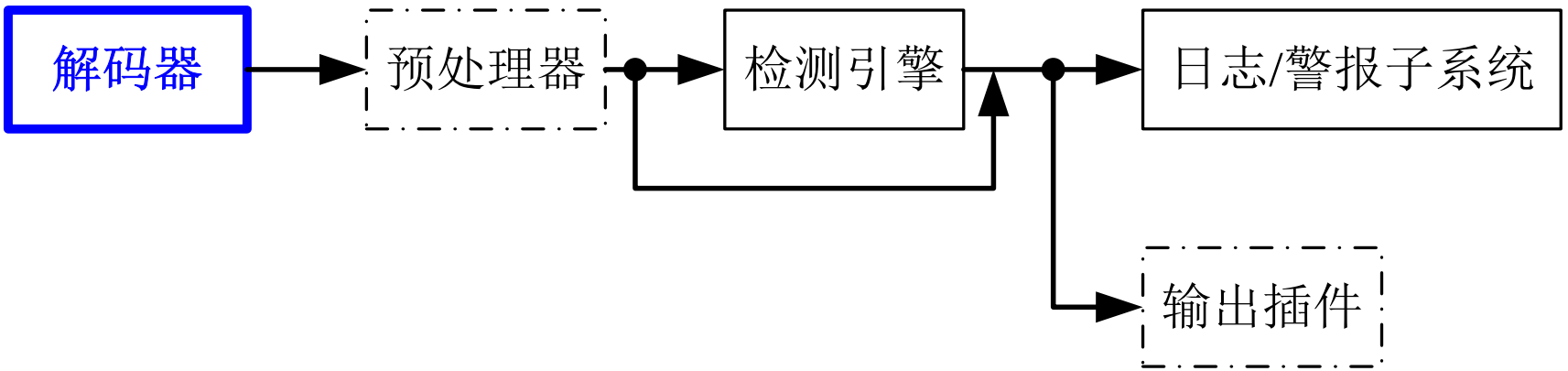
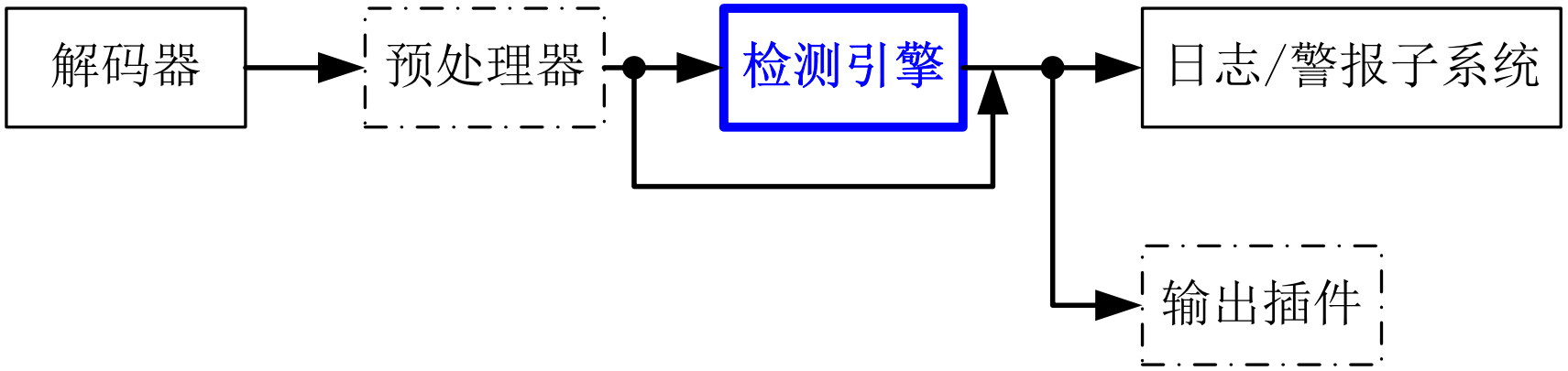


图6-7 Snort的结构



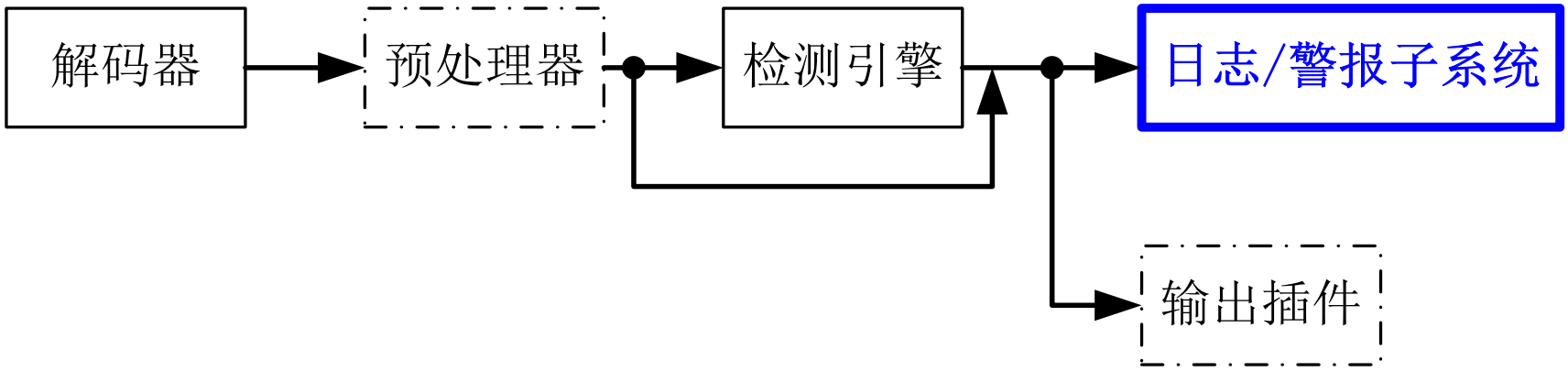
1.解码器

- 解码器负责从网络接口上获取数据包。在编程实现上，解码器用一个结构体来表示单个数据包，该结构记录了与各层协议有关的信息和其他检测引擎需要用到的信息。获取的信息将被送往检测引擎或者预处理器中。解码器支持多种类型的网络接口，包括Ethernet、SLIP、PPP等。



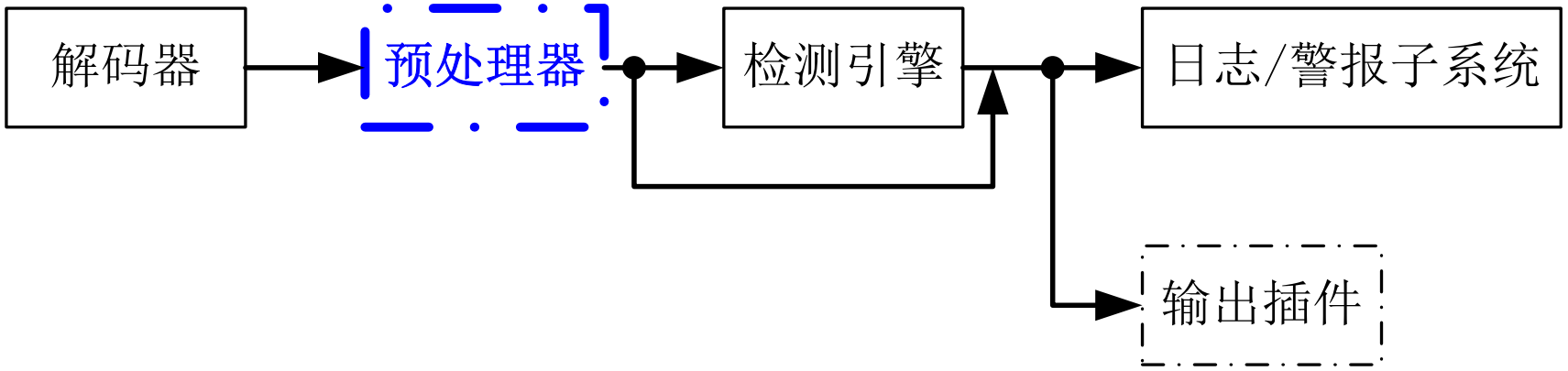
2.检测引擎

- 该子系统是Snort工作在入侵检测模式下的核心部分，它使用基于规则匹配的方式来检测每个数据包。一旦发现数据包的特征符合某个规则定义，则触发相应的处理操作。



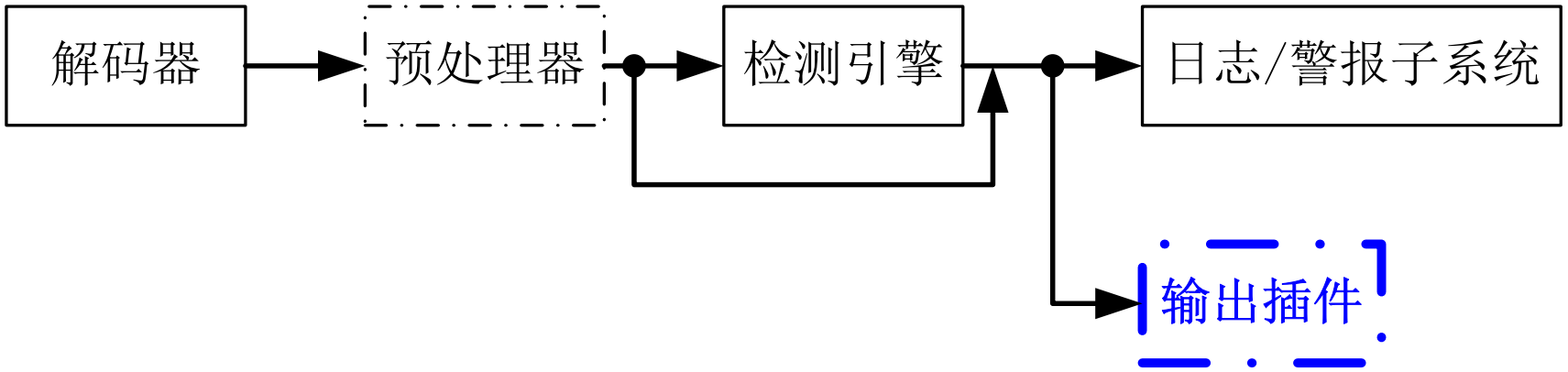
3.日志/警报子系统

- 规则中定义了数据包的处理方式，包括alter(报警)、log(日志记录)和pass(忽略)等，但具体的alter和log操作则是由日志/警报子系统完成的。日志子系统将解码得到的信息以ASCII码的格式或以tcpdump的格式记录下来，警报子系统将报警信息发送到syslog、socket或数据库中。



- Snort主要采用基于规则的方式对数据包进行检测，这种方式因匹配速度快而受到欢迎。
- 但对于Snort来说，超越基于规则匹配的检测机制是必要的。比如说，仅依赖规则匹配无法检测出协议异常。这些额外的检测机制在Snort中是通过预处理器来实现的，它工作在检测引擎之前，解码器之后。

- Snort中包含了三类预处理器，分别实现不同的功能：
  - **包重组**。这类预处理器的代表有stream4和frag2。它们将多个数据包中的数据进行组合，构成一个新的待检测包，然后将这个包交给检测引擎或其他预处理器。
  - **协议解码**。为了方便检测引擎方便地处理数据，这类预处理器对Telnet，HTTP和RPC等协议进行解析，并使用统一规范的格式对其进行表述。
  - **异常检测**。用来检测无法用一般规则发现的攻击和协议异常。与前面两种预处理器相比，异常检测预处理器更侧重于报警功能。



5.输出插件

- 输出插件用来格式化警报信息，使得管理员可以按照公司环境来配置容易理解、使用和查看的报警和日志方法。例如，某公司使用MySQL来存储公司和客户的信息，他们的报表系统是基于MySQL之上的，那么，对于该公司来说，把入侵检测的日志和报警信息保存在MySQL中就显得非常有用。Snort有大量的插件来支持不同的格式，包括数据库、XML、Syslog等格式，从而允许以更加灵活的格式和表现形式将报警及日志信息呈现给管理员。

Snort的工作流程

- 首先，Snort利用libpcap进行抓包。之后，由解码器将捕获的数据包信息填入包结构体，并将其送到各式各样的预处理器中。对于那些用于检测入侵的预处理器来说，一旦发现了入侵行为，将直接调用输出插件或者日志、警报子系统进行输出；对于那些用于包重组和协议解码的预处理器来说，它们会将处理后的信息送往检测引擎，由检测引擎对数据包的特征及内容进行检查，一旦检测到与已知规则匹配的数据包，或者利用输出插件进行输出，或者利用日志、警报子系统进行报警和记录。

Snort的安装、配置与使用

- 请从官方网站www.snort.org下载用户手册。
  - 截至2017年10月25日，最新版本为2.9.11

6.4 IDS的发展方向

- 随着网络技术和网络规模的不断发展，人们对计算机网络的依赖也不断增强。与此同时，针对网络系统的攻击也越来越普遍，攻击手法日趋复杂。

- 为了应对日益复杂的网络入侵，IDS技术也在不断进步。大致地说，IDS的发展趋势主要表现在以下方面：

(1) 宽带高速实时检测技术

- 大量高速网络技术(如千兆以太网等)在近年相继出现。在此背景下，各种宽带接入手段层出不穷。如何实现高速网络下的实时入侵检测已经成为现实面临的问题。

- 目前的千兆IDS产品的性能指标与实际要求相差很远。要提高其性能主要需考虑以下两个方面：
  - 首先，IDS的软件结构和算法需要重新设计，以适应高速网的环境，提高运行速度和效率；
  - 其次，随着高速网络技术的不断发展与成熟，新的高速网络协议的设计也必将成为未来发展的趋势，那么，现有IDS如何适应和利用未来的新网络协议，将是一个全新的问题。

(2) 大规模分布式的检测技术

- 传统的集中式IDS的基本模型是在网络的不同网段放置多个探测器，收集当前网络状态信息，然后将这些信息传送到中央控制台进行处理。这种方式存在明显的缺陷：
  - 首先，对于大规模分布式攻击，中央控制台的负荷将会超过其处理极限，这种情况会造成大量信息处理的遗漏，导致漏警率增高；
  - 其次，多个探测器收集到的数据在网络上传输会在一定程度上增加网络负担，导致网络系统性能降低；
  - 再者，由于网络传输的时延问题，中央控制台处理的网络数据包所包含的信息只反映探测器接收它时的网络状态，不能实时反映当前网络状态。

- 面对以上问题，新的解决方法也随之产生，如Purdue大学开发的AAFID系统。该系统是Purdue大学设计的一种采用树形分层构造的代理群体，其根部是监视器代理，提供全局的控制、管理及分析由上一层节点提供的信息。树叶部分的代理专门负责收集信息。处在中间层的代理被称为收发器。这些收发器一方面实现对底层代理的控制，一方面可以对信息进行预处理，把精练的信息反馈给上层的监视器。这种结构采用了本地代理处理本地事件、中央代理负责整体分析的模式。与集中式不同，它强调通过全体智能代理的协同工作来分析入侵策略。这种方法明显优于前者，但同时带来了一些新的问题，如代理间的协作、代理间的通信等。这些问题仍在进一步研究之中。

(3) 数据挖掘技术

- 操作系统的日益复杂和网络数据流量的急剧增加导致审计数据以惊人的速度增加。如何在海量的审计数据中提取具有代表性的系统特征模式，对程序和用户行为做出更精确的描述，是实现入侵检测的关键。
- 数据挖掘技术是一项通用的知识发现技术，其目的是从海量数据中提取对用户有用的数据。将该技术用于入侵检测领域，利用数据挖掘中的关联分析、序列模式分析等算法提取相关的用户行为特征，并根据这些特征生成安全事件的分类模型，应用于安全事件的自动认证。

- 一个完整的基于数据挖掘的入侵检测模型包括对审计数据的采集、数据预处理、特征变量选取、算法比较、挖掘结果处理等一系列过程。这项技术的难点在于如何根据具体应用要求，从用于安全的先验知识出发，提取出可以有效反映系统特性的特征属性，应用适合的算法进行数据挖掘。另一个技术难点在于如何将挖掘结果自动地应用到实际IDS中。
- 目前，国际上对这个方向的研究很活跃，这些研究多数得到美国国防部高级计划署、国家自然科学基金的支持。但我们也应看到，数据挖掘技术用于入侵检测的研究从总体上来说还处于理论探讨阶段，离实际应用还有距离。

(4) 更先进的检测算法

- 在入侵检测技术的发展过程中，新算法的出现可以有效提高检测效率。下述三种机器学习算法为当前检测算法的改进注入了新的活力。它们分别是计算机免疫技术、神经网络技术和遗传算法。
- 1)计算机免疫技术是直接受到生物免疫机制的启发而提出的。在生物系统中，脆弱性因素由免疫系统来处理，而这种免疫机制在处理外来异体时呈现出分布、多样性、自治及自修复等特征，免疫系统通过识别异常或以前未出现的特征来确定入侵。计算机免疫技术为入侵检测提供了一个思路，即通过正常行为的学习来识别不符合常态的行为序列。这方面的研究工作已经开展很久，但仍有待于进一步深入。

2)神经网络技术在入侵检测中的应用研究时间较长，并在不断发展。早期的研究通过训练后向传播神经网络来识别已知的网络入侵，进一步研究识别未知的网络入侵行为。今天的神经网络技术已经具备相当强的攻击模式分析能力，能够较好地处理带噪声的数据，而且分析速度很快，可以用于实时分析。现在提出了各种其他神经网络架构，诸如自组织特征映射网络等，以期克服后向传播网络的若干限制性缺陷。

3)遗传算法在入侵检测中的应用研究时间不长，在一些研究试验中，利用若干字符串序列来定义用于分析检测的命令组，用以识别正常或异常行为。这些命令在初始训练阶段不断进化，分析能力明显提高。该算法的应用还有待于进一步的研究。

(5) 入侵响应技术

- 当IDS检测出入侵行为或可疑现象后，系统需要采取相应手段，将入侵造成的损失降至最小。系统一般可以通过生成事件告警、E-mail或短信息来通知管理员。
- 随着网络变得日益复杂和安全要求的不断提高，更加实时的系统自动入侵响应方法正逐渐得到研究和应用。这类入侵响应大致分为三类：系统保护、动态策略和攻击对抗。这三方面都属于网络对抗的范畴，系统保护以减少入侵损失为目的；动态策略以提高系统安全性为职责；而攻击对抗则不仅可以实时保护系统，还可实现**入侵跟踪和反入侵的主动防御策略**。

6.5 NIDS的脆弱性及攻击方法

- 反NIDS的目标是：使*NIDS*检测不到入侵行为的发生，或无法对入侵行为做出响应，或无法证明入侵行为的责任。

- 其策略主要有三种：
  - 规避NIDS的检测；
  - 针对NIDS自身发起攻击，使其无法正常运行；
  - 借助NIDS的某些响应功能达到入侵或攻击目的。

6.5.1 NIDS所面临的几个问题

(1) **检测的工作量很大**

– NIDS**需要高效的检测方法和大量的系统资源。**

- 通常NIDS检测保护的是一个局域网络，其数据流量通常会比单机高出一到两个数量级，且由于协议的层次封装特性，使得很多信息要逐层地从网络数据包中提取并分析，NIDS的检测分析工作因此而变得十分繁杂。NIDS必须尽快地处理网络数据包，以保持与网络同步，避免丢包。

– NIDS**的检测是资源密集型的**，这在某种程度上使NIDS更加容易遭受DoS攻击。

(2) 检测方法的局限性

- 复杂的、智能化方法的作用十分有限，而AD方法(**误用检测方法**)受限于某些资源的请求使用在数据传输过程中的模糊性与隐含性，也难以在NIDS中发挥另人满意的功效。特征匹配成为NIDS分析引擎的一个不可或缺的模块功能。
- 特征匹配作为一种轻量级的检测方法有其固有的缺陷，缺乏弹性（尤其是字符串匹配），**如何完备定义匹配特征（也即匹配特征库的完备性）是决定检测性能的一个关键问题。**

(3) 网络协议的多样性与复杂性

- TCP/IP协议族本身十分庞杂，各种协议不下几十种，呈现横向跨越和纵向深入的两维分布。为了适应网络检测的需要，NIDS须对其中的大部分协议进行模拟分析检测工作，这会使得分析引擎变得臃肿而效率低下。
- 更为重要的是部分协议（如IP协议、TCP协议等）非常复杂，使精确地模拟分析十分困难，其难度随着协议层次的上升而增加。到了应用层，这种模拟分析工作几乎无法继续，由于缺少主机信息，*NIDS将难于理解应用层的意图，更无法模拟或理解某些应用提供的功能（如*bash*提供的*tab*键命令补齐功能）作用于具体环境下所产生的效果。*

(4) 系统实现的差异

- 具体实现时，各种系统不完全按RFC，对那些建议值和可选功能，会有自己的偏好。NIDS为了逼近各种系统的实现就必须尽可能多地了解每一种系统对这些不一致情况的处理方式，然后根据实际应用中检测保护的对象再决定分析动作。但这种想法在实际中并不完全可行，**有些问题不仅仅是系统的实现问题，还包含了用户的配置选择（如是否计算UDP数据报的校验和）**，因此很难做到与目标系统的一致性处理。
- 另外，某些系统（如Unix）出于操作的自由性和应用的方便性，允许用户对网络底层进行直接操作，致使入侵者几乎可以随心所欲地构造各种奇特的数据包。

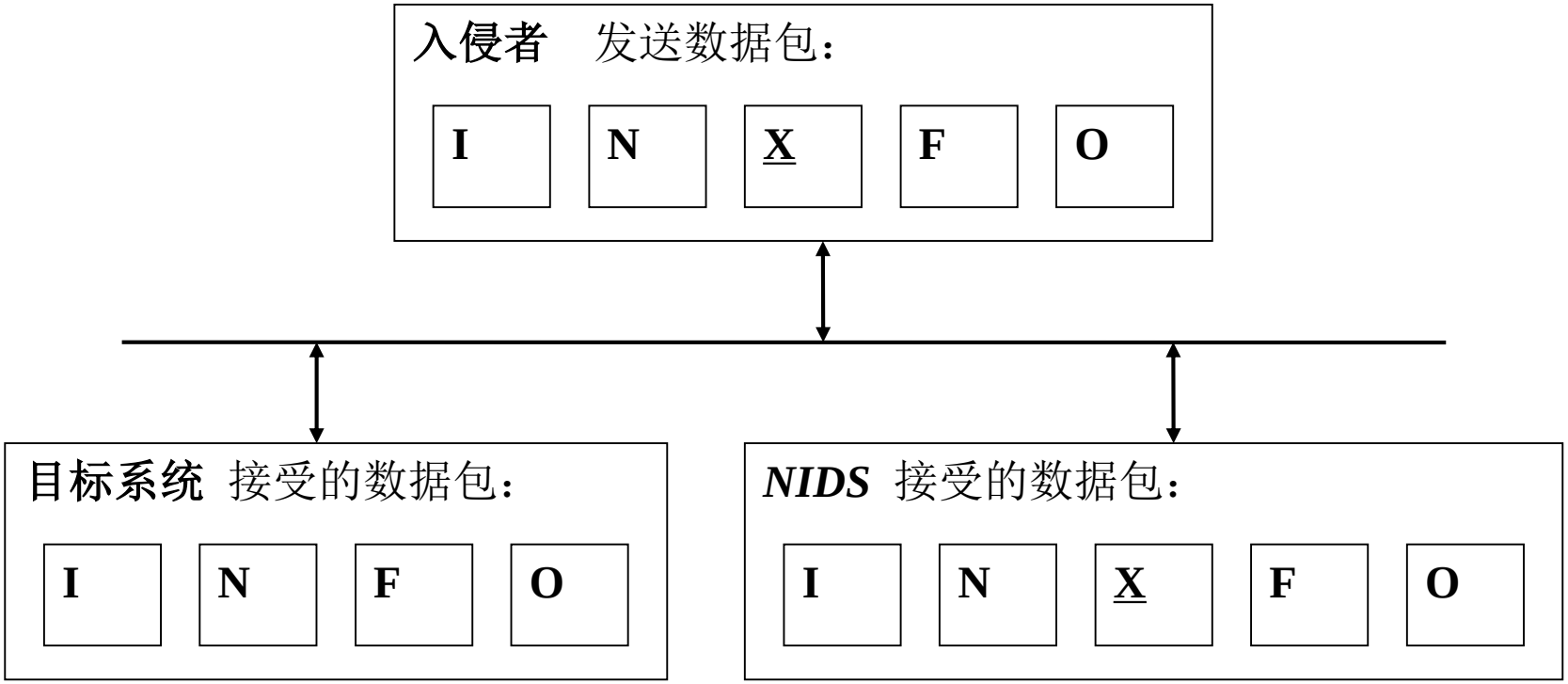
6.5.2 规避NIDS

- 基于特征匹配检测固有的脆弱性，入侵者可以利用上面提到的几个问题，对普通的入侵行为加以掩饰，或使NIDS无法接收准确完整的入侵特征，或使入侵特征变得模糊而难以识别，NIDS在检测不到入侵特征的情况下将很难识别入侵者的行为意图，从而无法检测到入侵的发生。这种攻击手法称为规避NIDS。
- 分析引擎是IDS的检测主体，规避NIDS的检测就是要**隐藏或消除NIDS分析引擎检测规则中所对应的行为模式或入侵特征**，使入侵行为淹没在正常的网络流量“噪声”中。

6.5.2.1 Insertion技术

- *Insertion*是指诱使*NIDS*接受一些目标系统无法接收或是拒绝接受的数据包。
- 通常情况下*NIDS*与目标系统应该接受相同的数据包，并按照相同的方式分析处理，这样才能保证*NIDS*对那些针对目标系统的入侵进行有效地检测。但是由于*NIDS*不可能完全了解目标系统的处理方式（即使是相同的系统也可能会因为配置问题而进行不同的处理），因此，如果*NIDS*接受了那些目标系统无法接收或是拒绝接受的数据包就有可能导致问题的发生。
- *Insertion*是一种比较常见的规避*NIDS*检测的技术，它主要对下述两个方面造成影响。

其一，针对非单特征点的入侵特征进行插入操作，将导致特征失效。



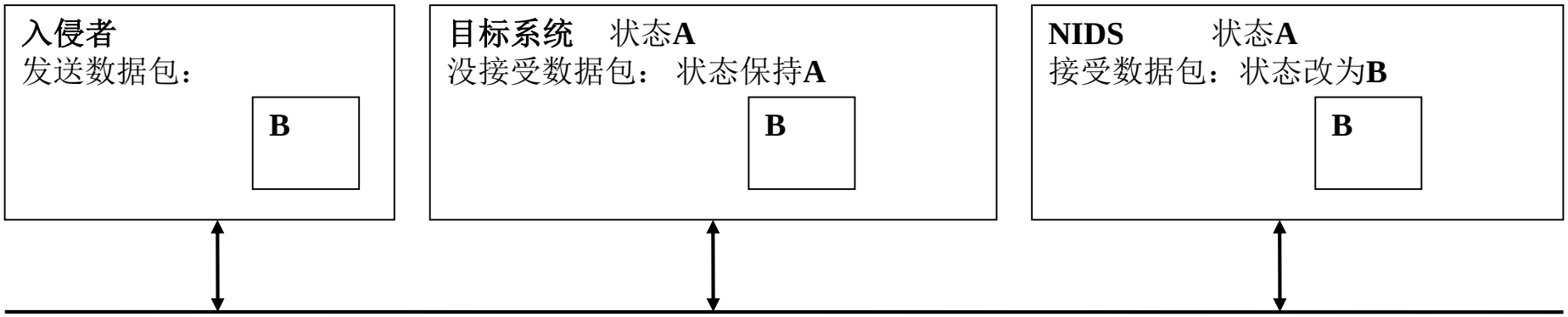
Insertion技术（续2）

- 假设“**INFO**”作为入侵的一个特征串，入侵者分四个数据包发出该特征串，并在发送第二个特征点“**N**”之后插入一个迷惑数据包“**X**”，该数据包对于目标系统来说是无法接收或拒绝接受的，但NIDS却错误地接受了，于是目标系统与NIDS得到不同的“特征串”。从目标系统来看入侵显然发生了，但对于NIDS来说它将无法检测到“**INFO**”，因此也不会据此判断入侵的发生。
- 对于这种情况而言，入侵特征应该是非单特征点的，因为单特征点是无法Insertion的，而且需要特征的分割传输，并在传输中间插入可以导致NIDS错误接受的迷惑性数据包。

Insertion技术（续3）

失去状态同步

- 其二，对于面向连接的协议，状态是一个十分重要的信息，传输的每一个数据包都携带状态信息，并且会对连接双方的状态造成影响。
- 为了对这样的协议进行分析，NIDS将不得不保持同样的或是类似的状态信息以跟踪连接。一旦NIDS错误地接受了目标系统无法接收或是拒绝接受的数据包，那么两者之间的状态就会产生不一致，我们称之为“**失去状态同步**”。
- 失去状态同步的NIDS在对该连接的后继数据包的处理上将遇到困难。如图(Next)所示。



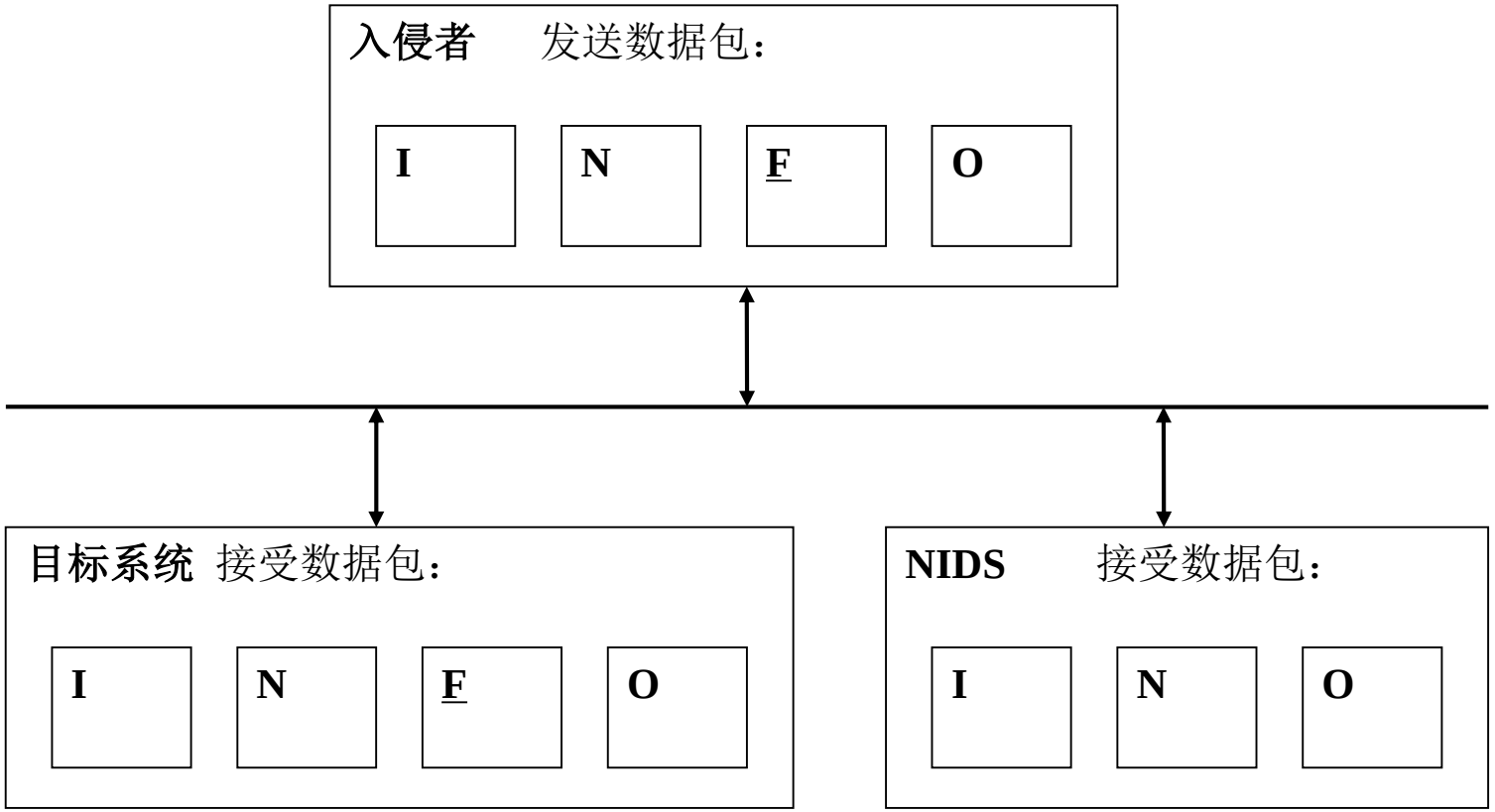
- 假设目标系统处于状态A，NIDS跟踪连接也保持为这一状态。现在入侵者发送一个携带状态B的数据包，因为某种原因目标系统没有接收(或接受)数据包，而NIDS却接受了该数据包并更新状态为B，于是与目标系统失去状态同步。在NIDS找回状态同步之前，对后继数据包接受与处理上可能会与目标系统产生不一致。更为严重的是，NIDS可能会在错误接受的数据包的误导下复位或终止相应的状态跟踪，以至放弃或失去对目标系统连接的跟踪。
- 这种影响并不基于特征匹配，但它只适用于需要维护状态信息的或面向连接的情况。

6.5.2.2 Evasion技术

- 这种技术与Insertion有些相似，这一次是NIDS**错误地拒绝了本应接受的数据包**，从而导致NIDS比目标系统“少”看到了东西，这种非一致性同样可能产生问题。

- 其影响也是两个方面。

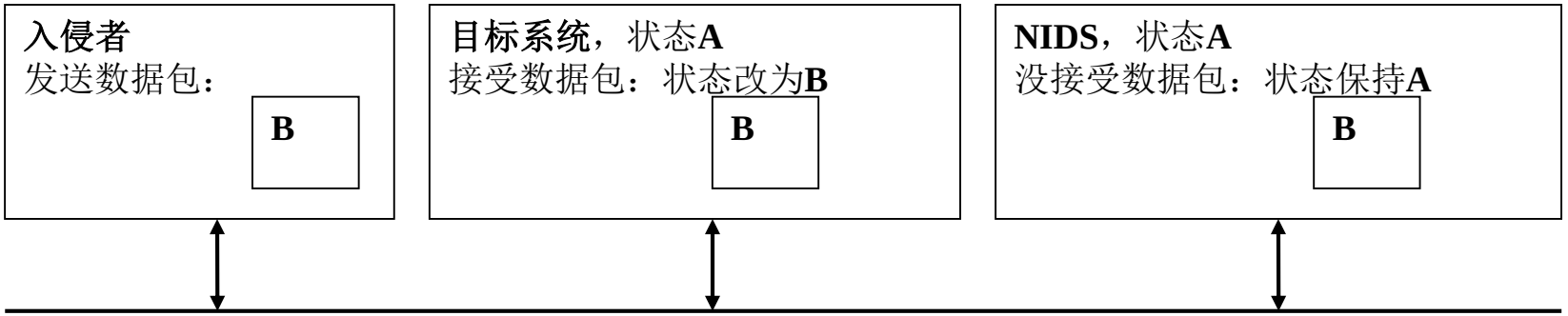
(1) 特征匹配问题



(1) 特征匹配问题（续1）

- 入侵者分四个数据包发送入侵特征串“**INFO**”，并且对第三个特征点数据包“**F**”进行一些特殊处理，使之看起来像一个不合理数据包，NIDS于是错误地将其拒绝，而目标系统本身却认为其是合法的而加以接受。显然NIDS无法匹配到“**INFO**”，但入侵确实发生了。
- 更为极端的是，入侵特征不必分割传输（即不限制特征点数目），而是只包含在一个数据包里，但该数据包却被NIDS错误地拒绝接受，那么入侵不被检测。
- 当然这里讨论的Evasion只限于软件原因引起的拒绝接受，而通过其他传播途径（如拨号连接）导致的NIDS接收不到数据包也能产生同样的效果。

(2)与*Insertion*中的第二点很相似，*NIDS*会因为少接受了数据包而错过状态的改变，从而导致失去状态同步

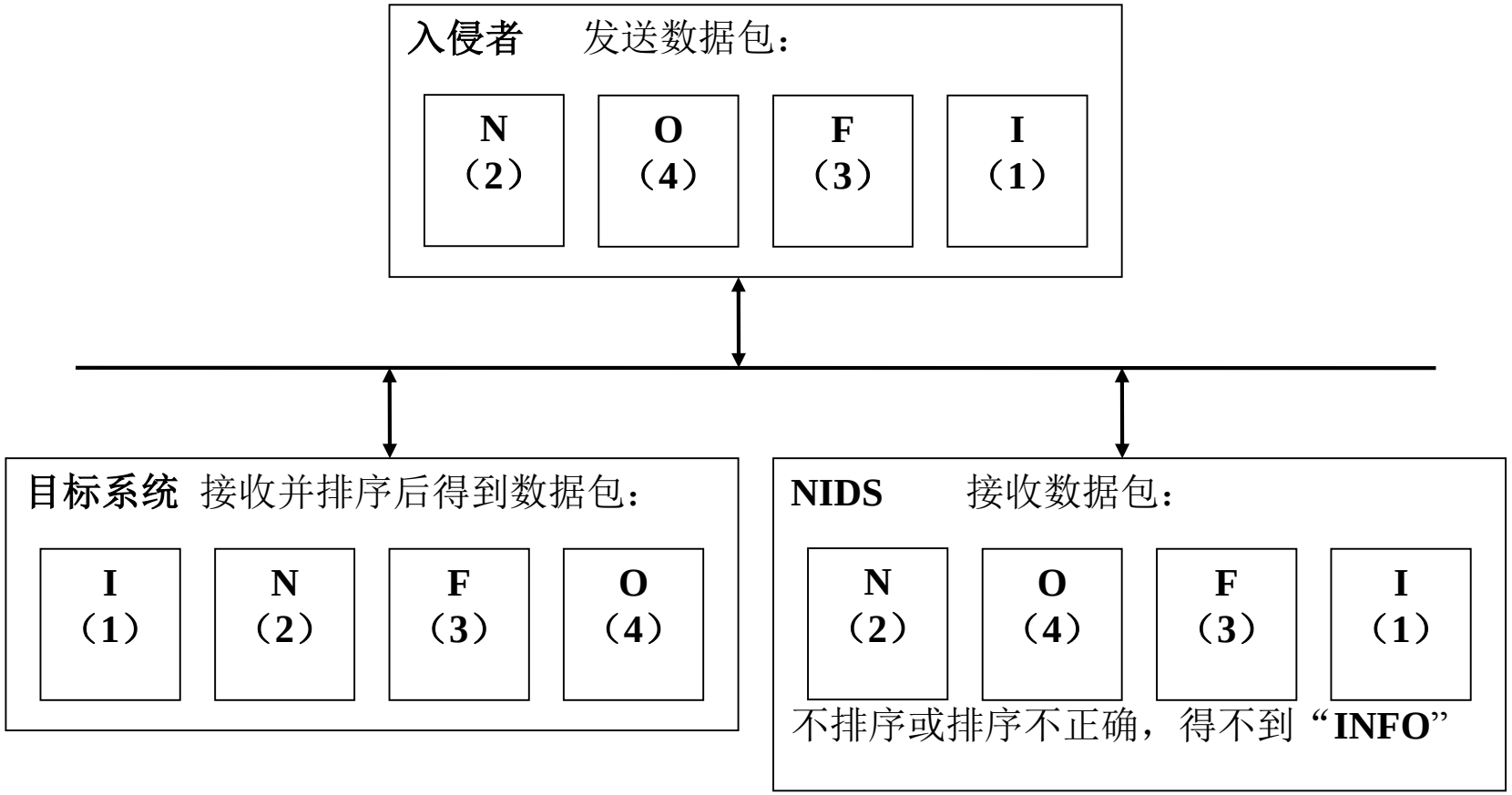


- 不同的地方在于：目标系统的状态更新可能对应着连接重置或连接终止，NIDS因为错过了状态更新而无法了解到这一情况，从而继续进行无意义地跟踪并占用资源，留下遭受DoS攻击的潜在危险。

6.5.2.3 Tousle技术

- **该技术打乱特征点的次序**，即将特征数据分割发送，并设法使包含各个特征点数据包不能按照特征点的原始次序到达目标系统，由目标系统根据相关信息恢复数据包的原始次序，而由于某些因素NIDS不能同样地对数据包排序。于是即使NIDS与目标系统接受了同样的数据包，但由于接受次序的不同，两者还是面对不同的“特征”。
- 这种技术的影响主要集中于那些特征（点）具有次序概念的系统和应用中，尤其是针对特征串匹配。

Tousle技术(续1)


入侵者 发送数据包：
N（2）
O（4）
F（3）
I（1）
目标系统 接收并排序后得到数据包：
I（1）
N（2）
F（3）
O（4）
NIDS 接收数据包：
N（2）
O（4）
F（3）
I（1）
不排序或排序不正确，得不到“INFO”

Tousle技术(续2)

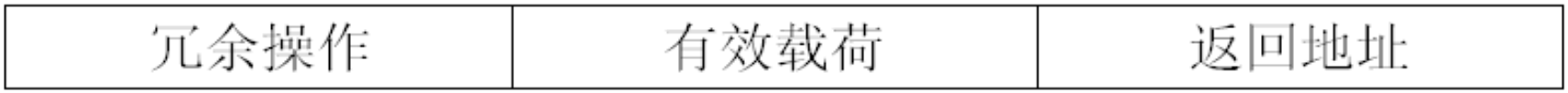
- 入侵者将特征串“**INFO**”分割为四个数据包进行传输，每个数据包携带的数字代表其在原始数据中的位置。现入侵者以(2)(4)(3)(1)的顺序发送各个数据包，不考虑网络拥塞和路由引起的传输问题，则数据包将很可能按发送的顺序到达目标系统和NIDS。目标系统能够根据每个数据包中的位置信息重构原始数据，得到特征串“**INFO**”；而NIDS由于某种原因没能够排序数据包或是不正确地排序数据包，于是得到了与原始数据不同的“特征串”，错过对入侵的检测。

6.5.2.4 *Anamorphosis-Polymorph*技术

- 该技术受病毒变种技术的启发，针对特征匹配缺乏弹性的固有缺陷，适当改变入侵模式，或通过某种手段（如加密）隐藏入侵特征，或使**入侵特征在一定程度上发生变化**（但对入侵不造成实质影响），以规避NIDS的检测。

- 在某种程度上类似于入侵变种，一个很好的例子就是用于缓冲区溢出攻击的**多态**shellcode**技术**。

修改*shellcode*规避入侵检测



*Anamorphosis-Polymorph*技术（续）

- 对于“冗余”操作来说，其目的只是为了增加跳转成功的几率，因此任何不影响有效载荷执行的操作都可以用来代替nop指令序列。这种“变种”操作很多，依目标系统的指令集而定，就IA32体系而言，粗略估计有不下五十种的操作可以选择。

- 如果以之代替shellcode中的nop序列，则那些检测特征系列nop的分析引擎将失败；如果分析引擎还能检测其他的替代操作，那么仅是为此目的而增加的特征就多达几十个；而且即使是这样，因为“冗余”操作是一系列的操作，攻击者还可以将各种替代操作以一种随机的方式组合，只要不超过shellcode的允许长度，则将产生成千上万种组合操作方式，这里不妨保守地计算一下：所有可以替代的操作选定为50种，“冗余”操作长度为5条指令，那么所有可能的组合共有50*50*50*50*50种之多，这是简单匹配难以对付的。

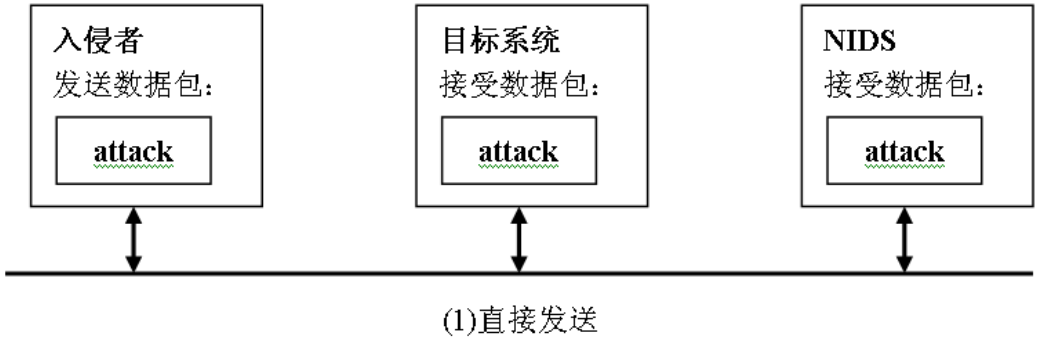
6.5.2.5 Substitution技术

- 该技术是一类规避技术的统称，其想法是：

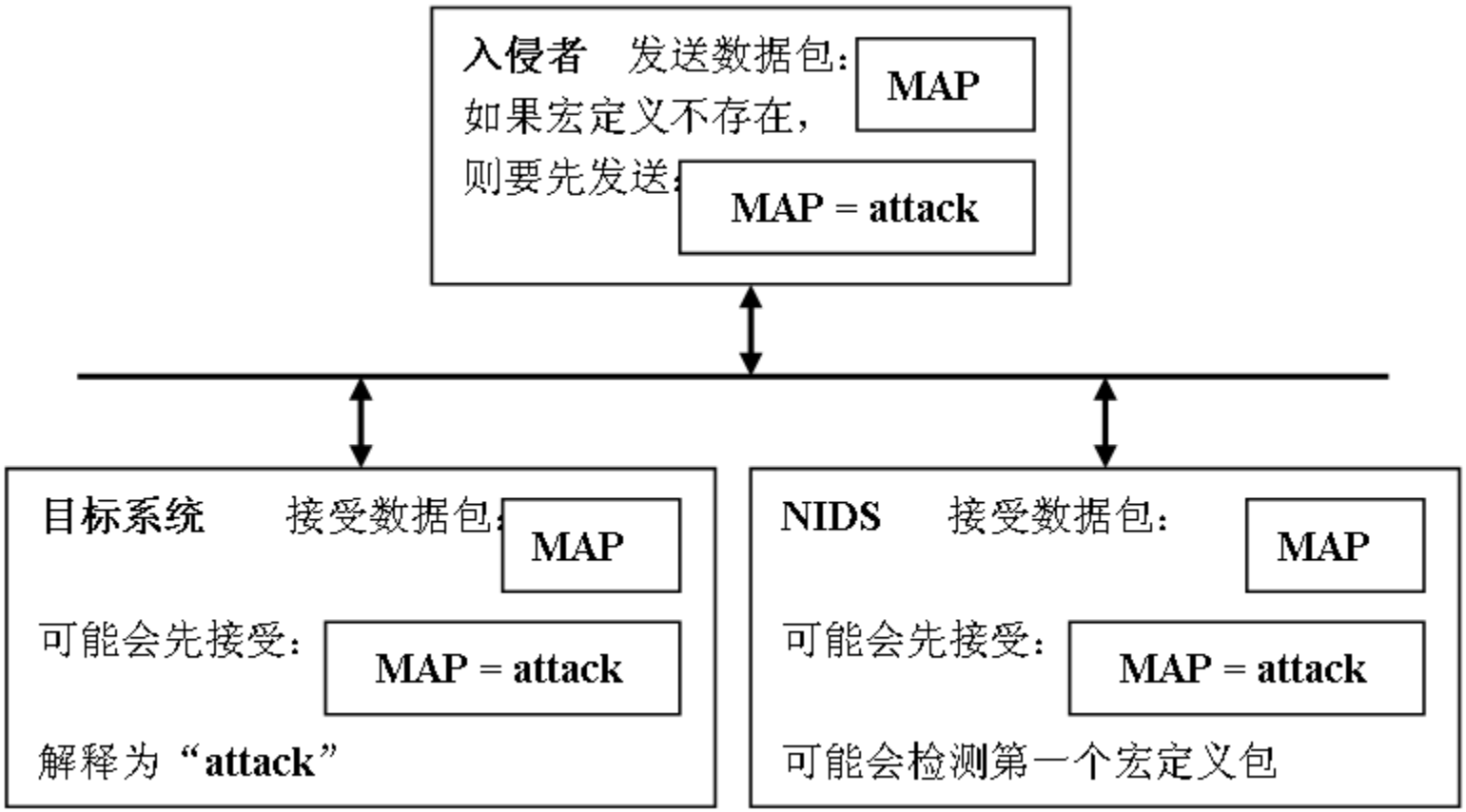
  不直接传输入侵特征，退而选择其他迂回的手段，使输入依靠目标系统的解释来完成，以达到同样的目的。

- 这种技术的好处在于传输过程不必出现特征数据，因此不易被基于特征匹配的分析引擎所检测，缺点是受限于目标系统的支持。

宏替换


入侵者
发送数据包：
attack
目标系统
接受数据包：
attack
NIDS
接受数据包：
attack

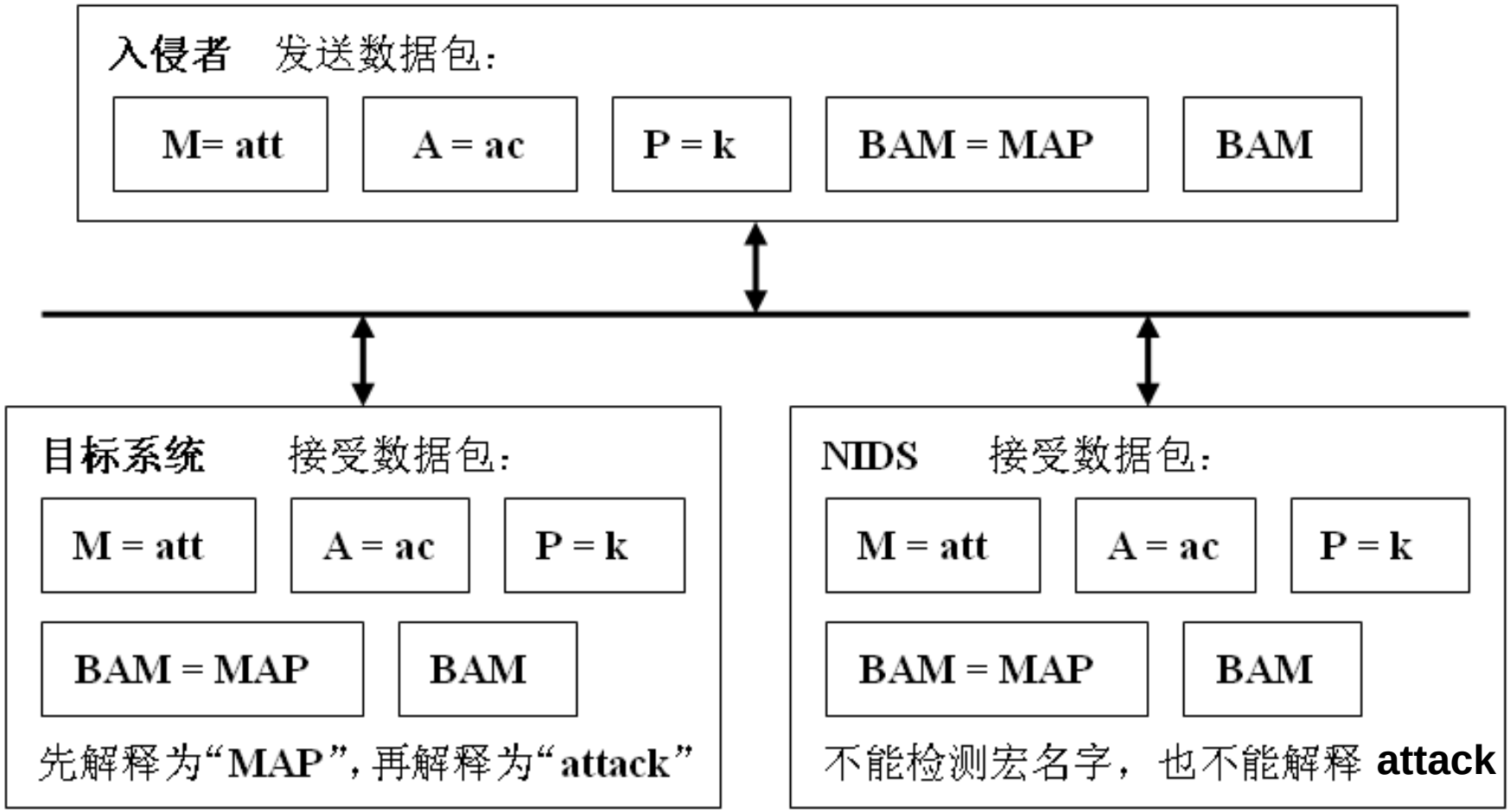
(1)直接发送


入侵者 发送数据包：
MAP
如果宏定义不存在，
则要先发送：
MAP = attack
目标系统 接受数据包：
MAP
可能会先接受：
MAP = attack
解释为“attack”
NIDS 接受数据包：
MAP
可能会先接受：
MAP = attack
可能会检测第一个宏定义包

宏替换

- 假定分析引擎匹配检测“**attack**”作为某入侵特征。入侵者若直接传输“**attack**”，则入侵将很容易被检测到；现已知目标系统（可以是操作系统支持，也可以是相关应用程序支持）支持宏扩展，入侵者设定宏“**MAP = attack**”，则入侵者可以以输入宏“**MAP**”来代替输入“**attack**”，然后由目标系统将其翻译为“**attack**”，再据此实现入侵者的意图。由于宏名字是由入侵者随意选择的，NIDS无法检测宏名字，因此可能错过对此次入侵的检测。
- 有时迂回可能不会这么简单，比如宏定义不是以前就存在的，那么定义宏“**MAP = attack**”的数据包可能会被分析引擎检测到，尽管以这种方式检测到的“**attack**”可能会增加误报率，但于此次入侵是无益的，因此需要更加迂回的手段。

二次宏技术


入侵者 发送数据包：
M= att
A = ac
P = k
BAM = MAP
BAM
目标系统 接受数据包：
M = att
A = ac
P = k
BAM = MAP
BAM
先解释为“MAP”，再解释为“attack”
NIDS 接受数据包：
M = att
A = ac
P = k
BAM = MAP
BAM
不能检测宏名字，也不能解释 attack

(3)二次宏

二次宏技术

- 这里可以采用**二次宏技术**，简单来说，入侵者可以定义“**M = att，A = ac，P = k**”，再定义“**BAM = MAP**”，之后输入“**BAM**”，目标系统经过两次宏扩展后同样会得到“**attack**”，而在整个传输的过程中将不会出现“**attack**”，加之宏名字选择的随意性和宏定义的嵌套性，NIDS除非支持同样的扩展机制，否则将很难检测到这样的入侵特征。
- 实际上Substitution技术所描述的决不仅仅是类似的宏扩展技术，其他诸如某些shell所支持的命令别名技术、环境变量技术、命令历史记录功能以及tab键命令补齐功能等，都可以看作是Substitution技术的不同应用实例。

6.5.2.6 Unicode-Related技术

- 如果入侵者只是简单地以ASCII码形式传输攻击序列，那么其中的特征数据将很容易被NIDS所识别。现在入侵者把特征数据以Unicode标准的UTF-8编码形式进行传输（假设目标系统支持Unicode），情况将会发生改变。

(1) NIDS不支持Unicode(这种情况现在已经不多了)，这样的分析引擎无法通过特征匹配来检测以Unicode方式传输的特征数据。

Unicode-Related技术（续1）

(2) NIDS支持Unicode，检测特征数据之前先进行编码转换，即分析引擎先把数据包的有效内容转化为Unicode，再对之进行特征匹配。但由于新旧标准的覆盖问题，分析引擎只有按照目标系统所支持的标准进行转换才能确保“看”到的内容与目标系统保持一致。潜在的问题是目标系统的多样性，为此NIDS需要了解每一个被检测保护系统所支持的标准，甚至是目标系统上每一个支持Unicode服务的所支持标准，这是一个烦琐而繁重的工作。如果对Unicode的支持是可选的，那么这种试图了解几乎不可能的。

Unicode-Related技术（续2）

(3) NIDS支持Unicode，检测特征数据之前不进行编码转换，即直接匹配UTF-8格式的编码。这种方法节省了转换时间，也避免了因支持标准不一致而导致的误解释，但是一个必须面对的新问题是多重表示。

- 随特征点的增多（绝大多数字符都有多重表示），这将产生同前一节中的“冗余”替换一样的组合爆炸问题。曾经有人做过统计，在微软Windows2K Advance Server的IIS上，Unicode的字符“A”有近30种的表示方法，“E”对应34种，“I”对应36种，“O”对应39种，“U”对应58种，仅一个字符串“AEIOU”就对应30*34*36*39*58种表示方法，以现有的匹配算法和机器的处理能力，对付如此巨量的特征匹配并非易事。

6.5.3 攻击NIDS

- 攻击NIDS**主要利用了其资源密集型检测的特点**，对其实施DoS攻击，并且通常结合欺骗技术，使得NIDS既不能找到攻击的真正来源，又不能执行正常的检测功能。入侵者利用这种空隙可以进行其他入侵活动。
- 另外入侵者还可以利用NIDS软件中可能存在的缺陷（或漏洞）对NIDS本身进行攻击，并可能导致NIDS的部分功能失效（取得与上面针对资源DoS攻击类似的效果）或是获得某些额外权限，继而为其他的非法活动打开方便之门。

6.5.4 利用*NIDS*

- **这种策略主要利用了NIDS的响应机制**。
- 作为NIDS必不可少的一个模块，响应部件除了警报功能之外，通常还配置了其他一些防护性功能。尽管这些功能的攻击性可能不是很强，但其针对性通常很强。
- 如果这些功能行使得不够谨慎，入侵者能够误导响应部件执行这些功能来对付目标系统，则它将成为破坏系统的工具，而NIDS本身却不知道真正发生了什么，因而也无法对这种入侵做出检测。

(1)“狼来了”效应

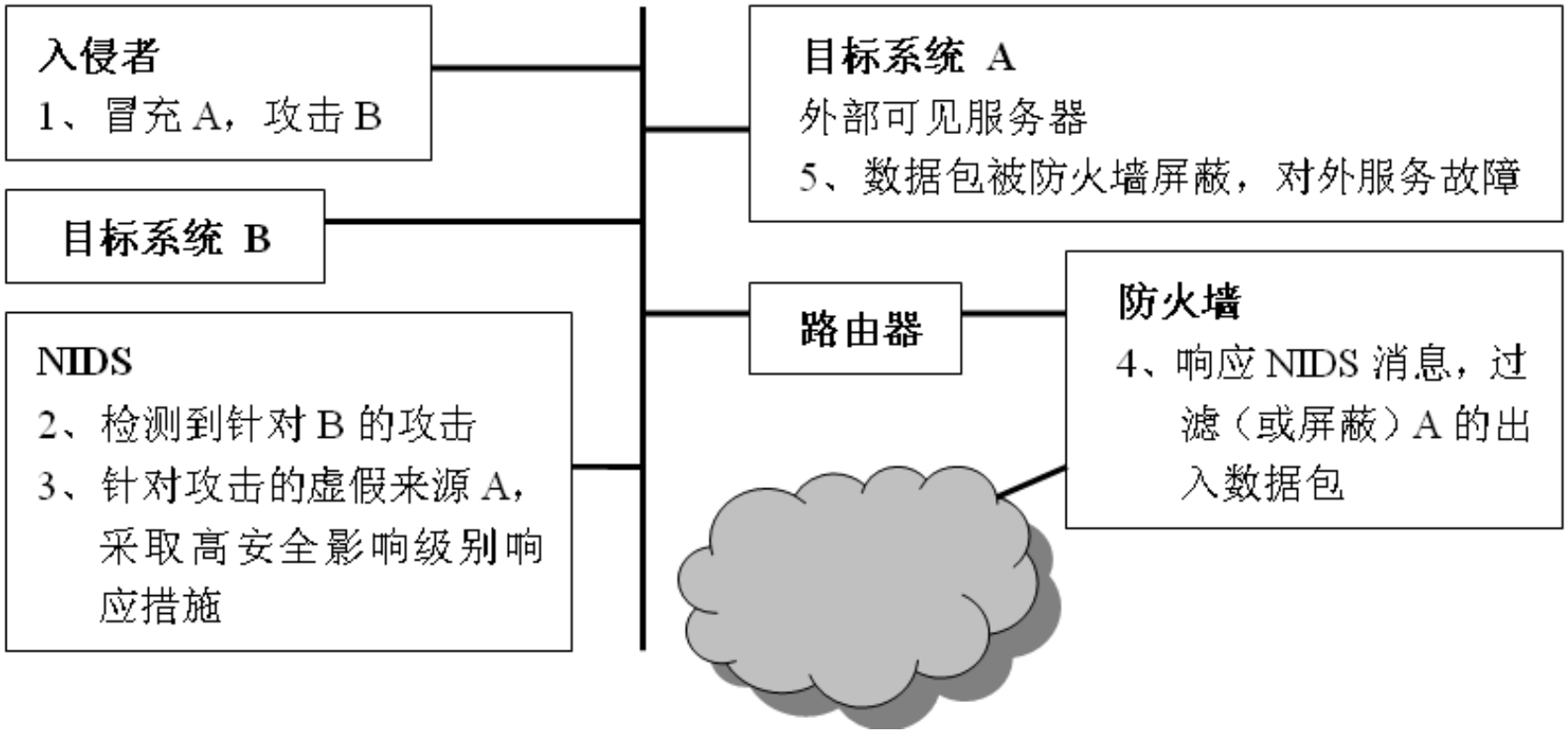
- 警报级别的响应措施包括屏幕文字、声音警报、电子邮件、日志记录、电话通知等。
- 由于是低影响级别的，因此不会对系统安全造成什么危害，利用的意义并不大。不过频繁的警报会影响用户的正常使用，尤其是虚假攻击的警报更会使人们怀疑NIDS的可信度，时间久了，可能会导致“狼来了”的效应，使得管理者忽略或干脆关闭（或部分关闭）警报功能。

(2) 利用防卫反击

- 这是安全影响级别最高的响应措施，也是安全隐患最大的级别，通常由它所造成的影响是持久的，有些甚至是难以恢复的。在这个级别上，如果NIDS检测到入侵发生，那么它采取的措施（有些也许是在被入侵之后而采取的补救措施）可能包括封锁用户、禁止地址、与其他安全工具交互配合调整、使用其他工具回击等。

- **下图给出了利用*NIDS*高安全影响级别响应措施对付目标系统的示意。**

利用防卫反击（续1）


入侵者
1、冒充A，攻击B
目标系统 B
NIDS
2、检测到针对B的攻击
3、针对攻击的虚假来源A，采取高安全影响级别响应措施
目标系统 A
外部可见服务器
5、数据包被防火墙屏蔽，对外服务故障
路由器
防火墙
4、响应NIDS消息，过滤（或屏蔽）A的出入数据包

利用防卫反击（续2）

- 现在从外部看来系统A与遭受DOS攻击没有多大的区别，入侵者的目的基本达到。这里隐含了一点假设，即入侵者知道什么样的攻击行为可以触发NIDS响应以防火墙配置更新的措施。
- 响应措施的不同造成的安全危害也不一样，但其共同点是这种危害由被误导的NIDS的响应直接所为，因此不在其检测之列。
- 上述划分并不意味着响应部件能够按照先低级再中高级的顺序选择不同影响级别的响应措施。入侵者需要了解那些高影响级别响应措施的触发机制，再施以适当的欺骗攻击，诱导响应部件对目标系统采取高安全影响级别的响应措施，以达到其借盾以击的目的。

作业和实践

- 作业
  - **总结NIDS的脆弱性**

- **上机实践(自己练习，不考核)**
  - 从www.snort.org 下载snort，部署在Linux系统下。
  - New：参考SEED的实验，设计一个网络嗅探系统：
    - http://www.cis.syr.edu/~wedu/seed/Labs_12.04/Networking/Sniffing_Spoofing/