

# 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 IDS的发展方向

## 6.5 NIDS的脆弱性及反NIDS技术

# 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

## 6.1.1 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、完整性和可用性等安全属性的行为。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”的另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络的入口处，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。

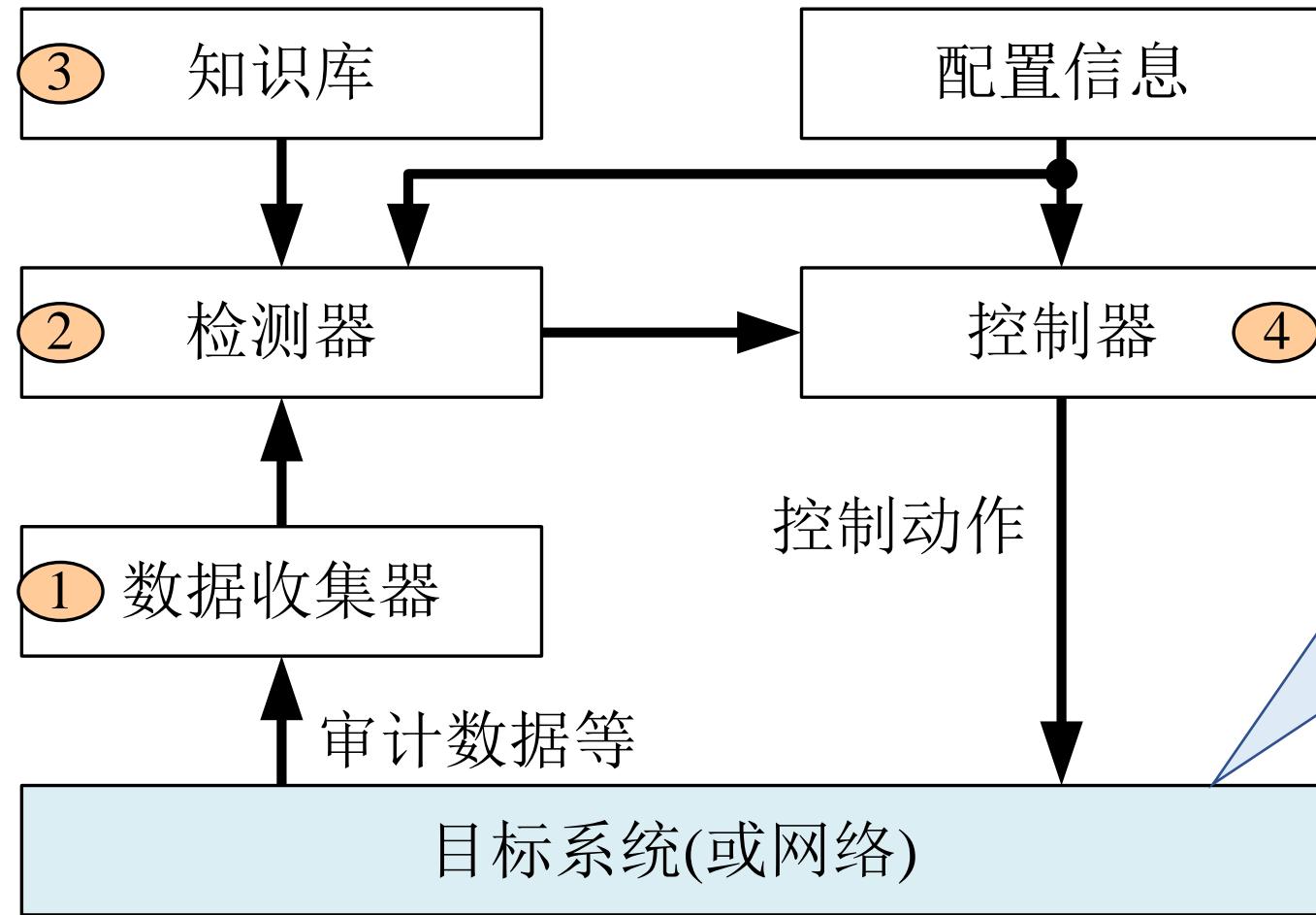
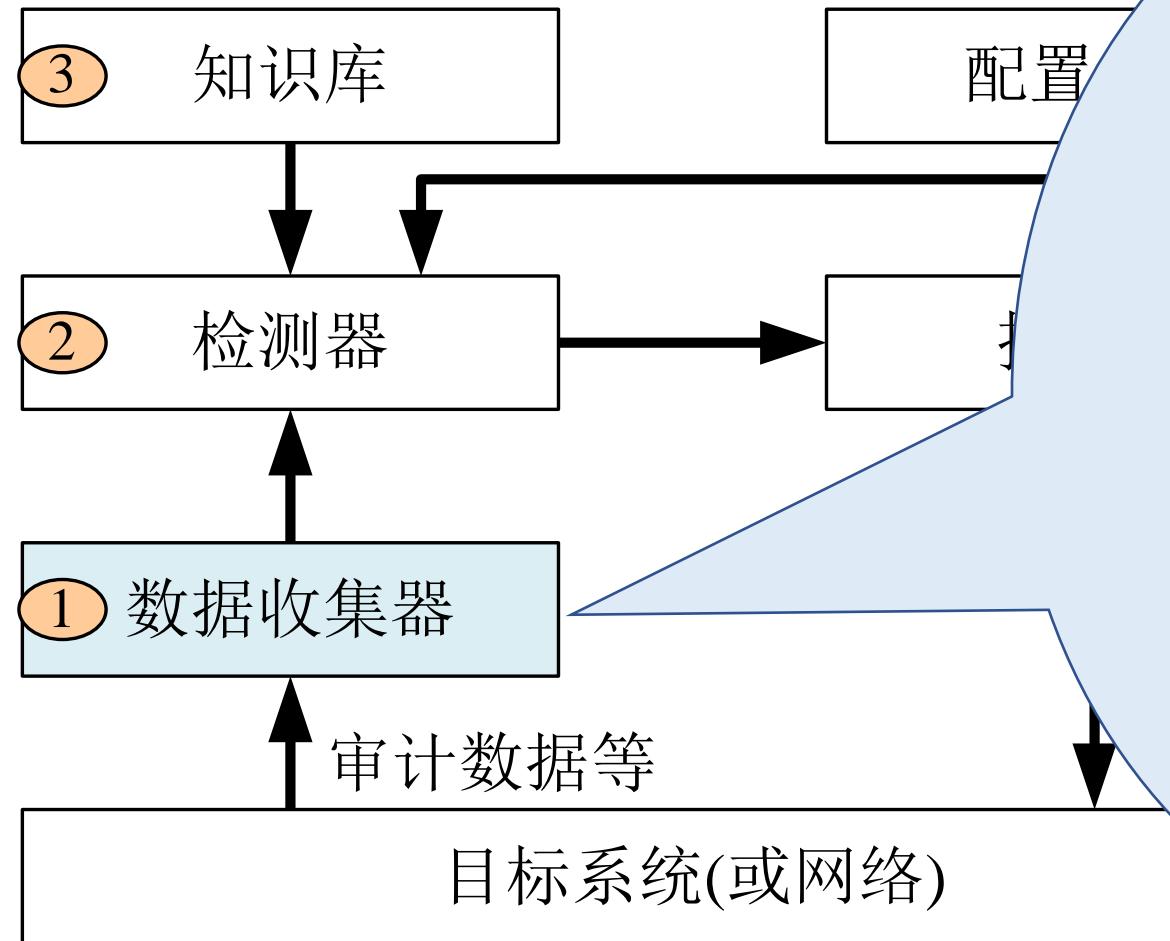


图 6-1 入侵检测系统



**(1)数据收集器：**  
又称 **探测器** 或 **传感器** (**sensor**)，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。

图 6-1 入侵检测系统

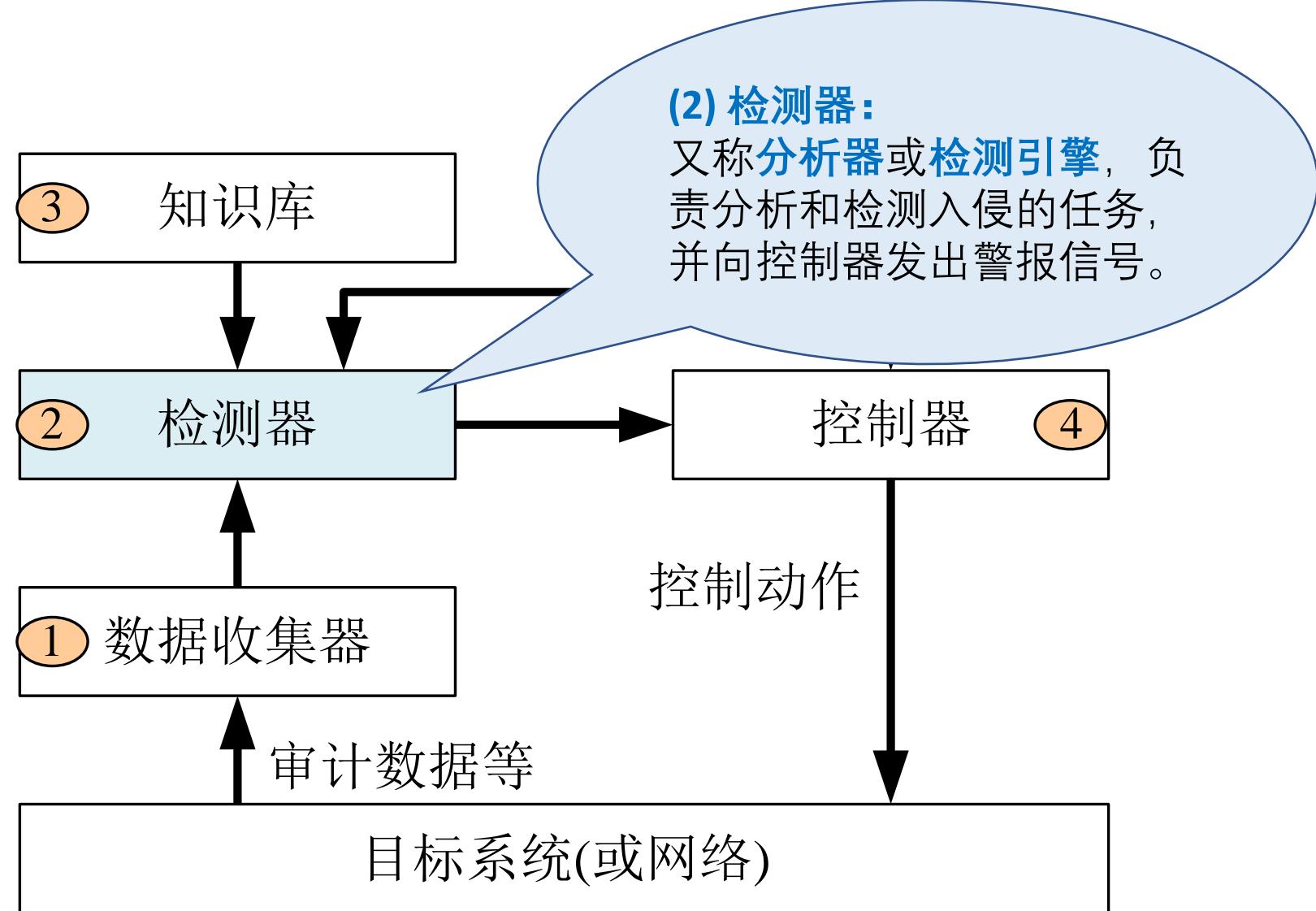


图 6-1 入侵检测系统

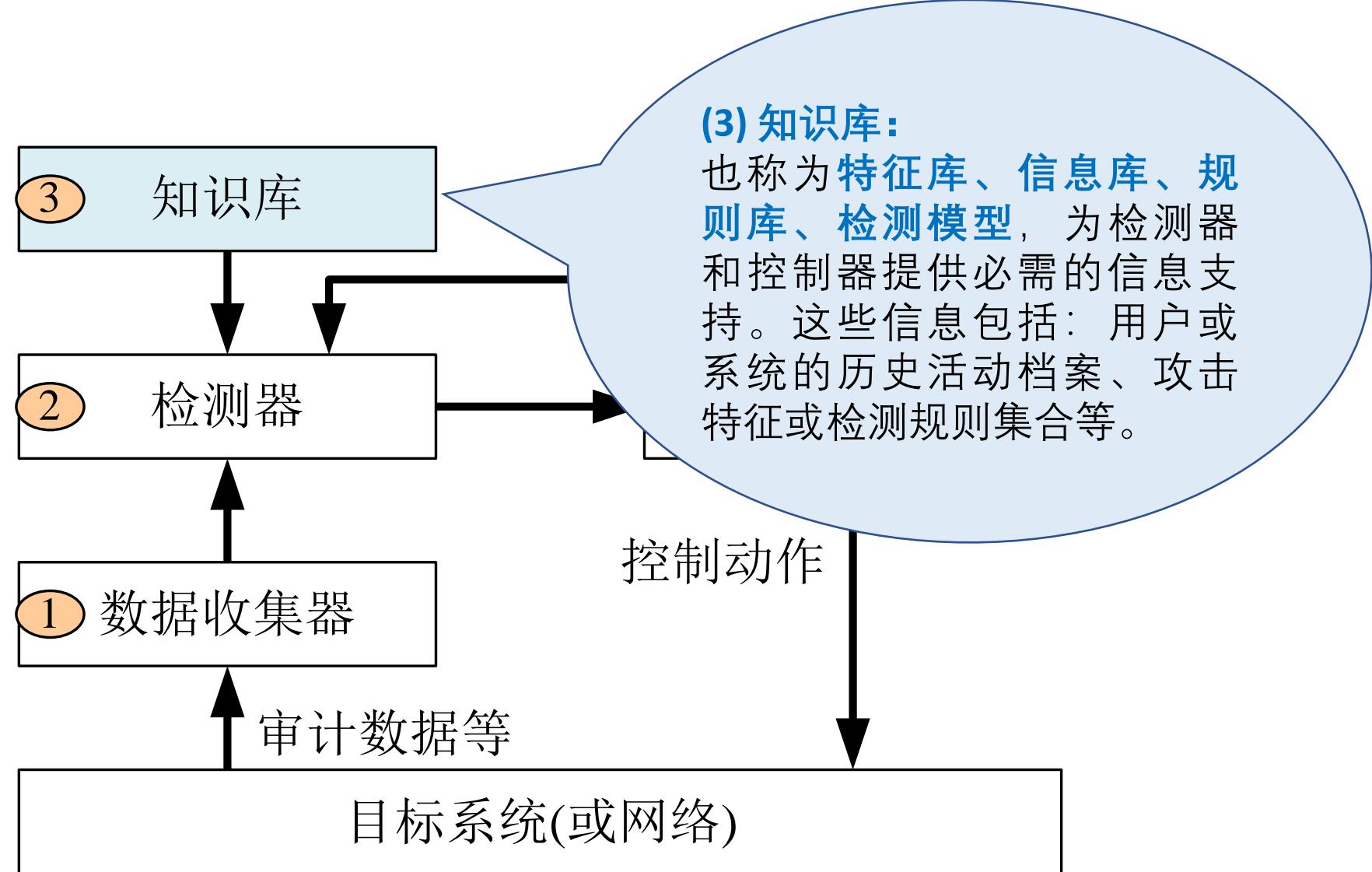


图 6-1 入侵检测系统

#### (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。

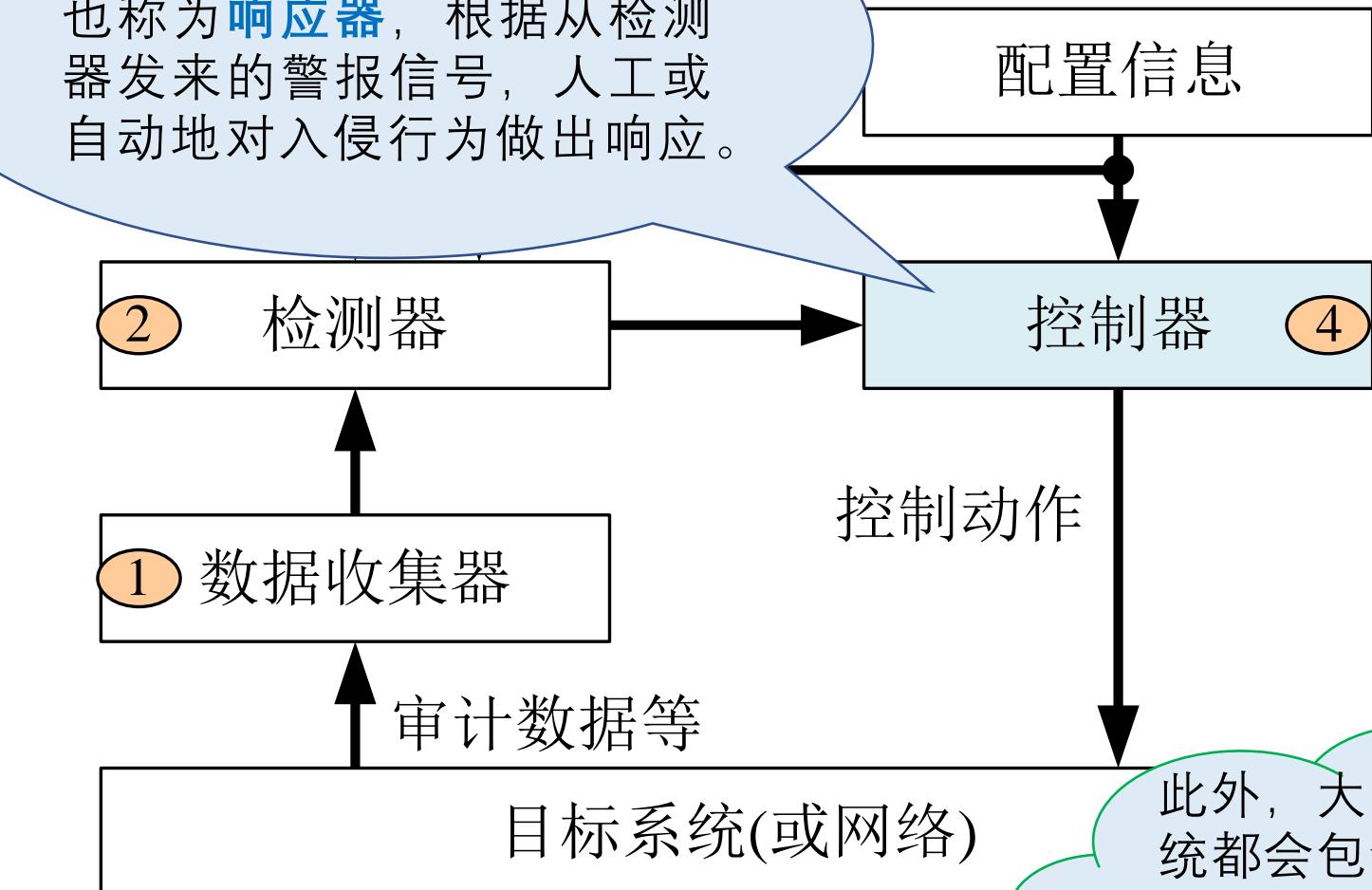


图 6-1 入侵检测系统

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**在网络与信息系统中的若干个关键点设置信息探测点**。
- IDS可利用的信息来源：
  - 1) 系统和网络的日志文件
  - 2) 目录和文件中的异常改变
  - 3) 程序执行中的异常行为
  - 4) 网络活动信息

## (2) 信息分析

- 对收集到的信息**进行分析，判断系统是否被入侵。**

### 1) 模式匹配

- 将收集到的信息与已知的网络或系统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) 信息分析—统计分析

### 2) 统计分析

- 一种入侵检测常用的**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。

## (2) 信息分析—完整性分析

### 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(哈希)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。缺点是当完整性分析未开启时，不能主动发现入侵行为。
- 进程的完整性分析是分析入侵的一种重要方法，其难点在于定义进程的完整性。在进程的完整性度量方面目前还没有好的解决方案。

### (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。
- 安全响应可分为**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、手机短信报警等。

### 6.1.3 IDS提供的主要功能

- (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。

## IDS提供的主要功能

- (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：**对新发现或新公布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。

## 6.1.4 IDS的分类

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

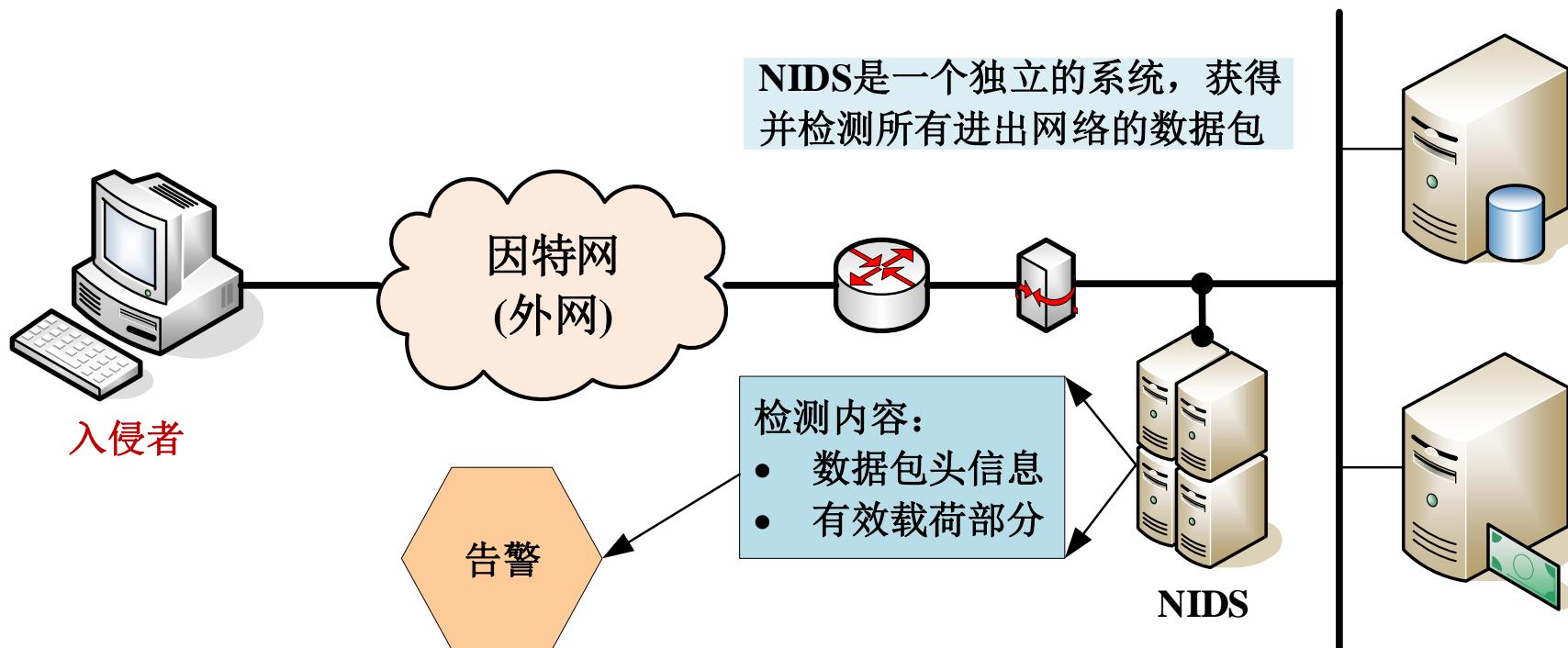


图 6-2 基于网络的入侵检测系统

✓ **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。

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

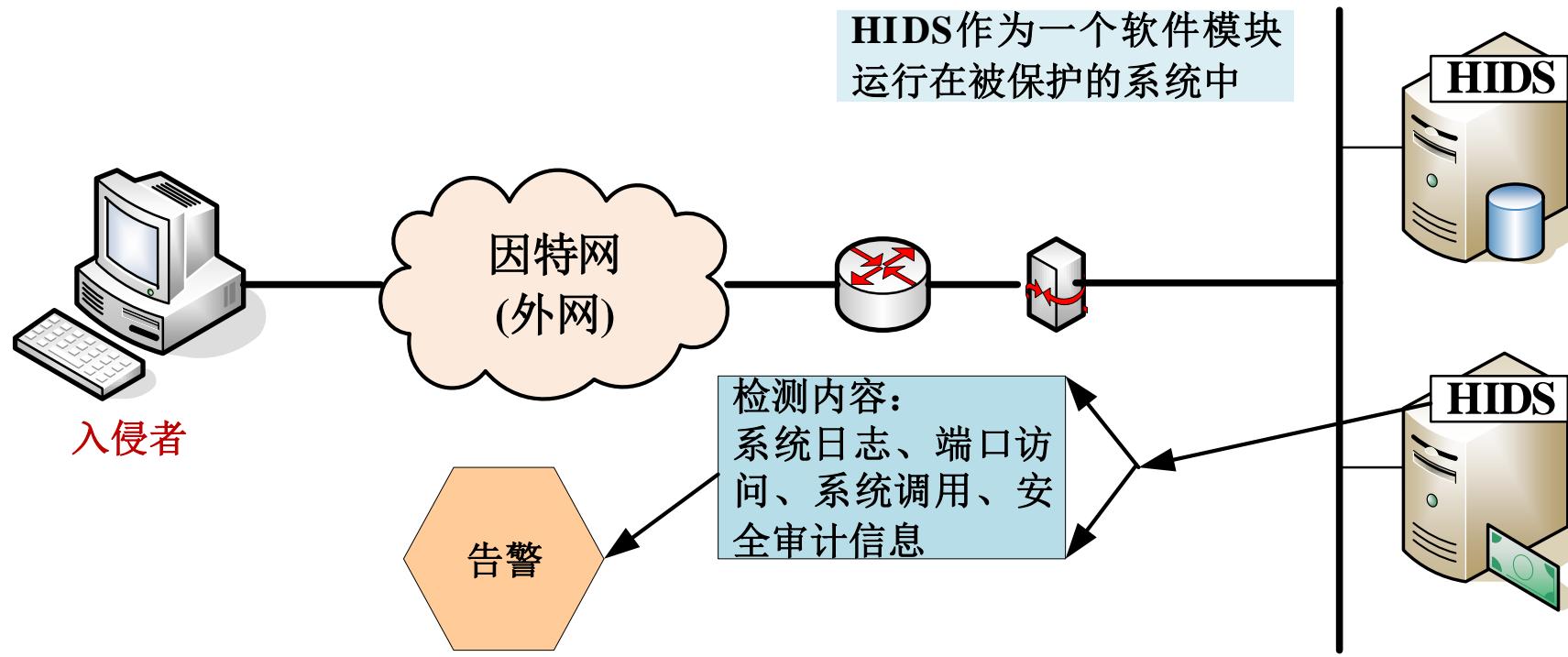


图 6-3 基于主机的入侵检测系统

- ✓ **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。

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

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

CIDF (Common Intrusion Detection Framework, 通用入侵检测框架) 模型由 S.Staniford 等人提出，主要目的有三个：

- 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。

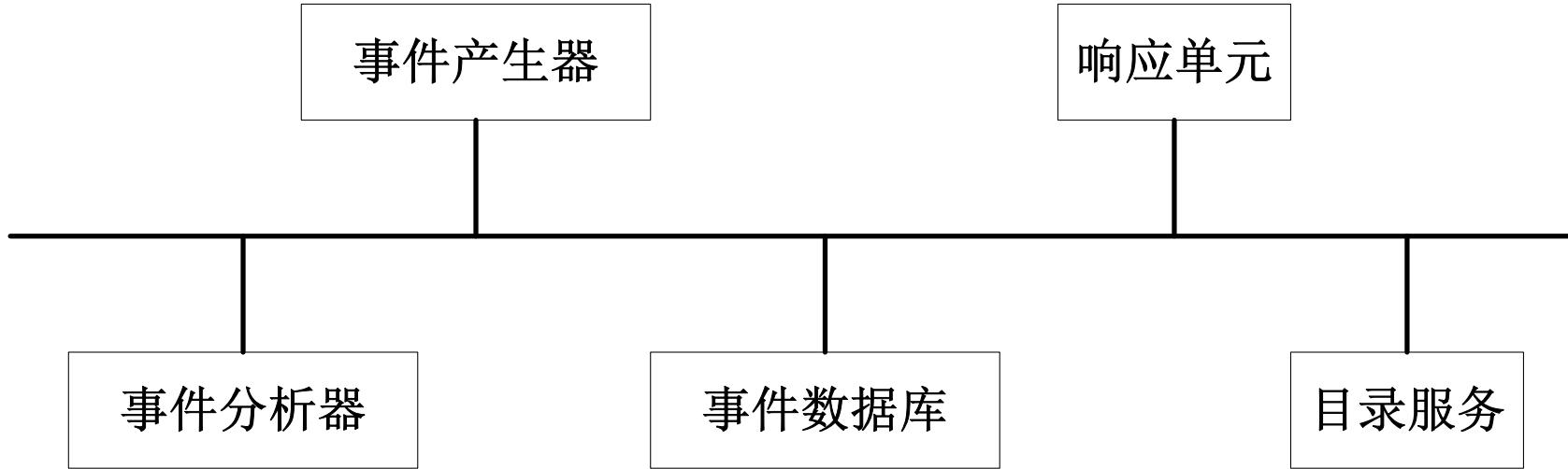


图 6-4 CIDF结构图

- CIDF模型将入侵检测需要分析的数据称作事件（Event）。
- 事件（Event）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。
- CIDF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。

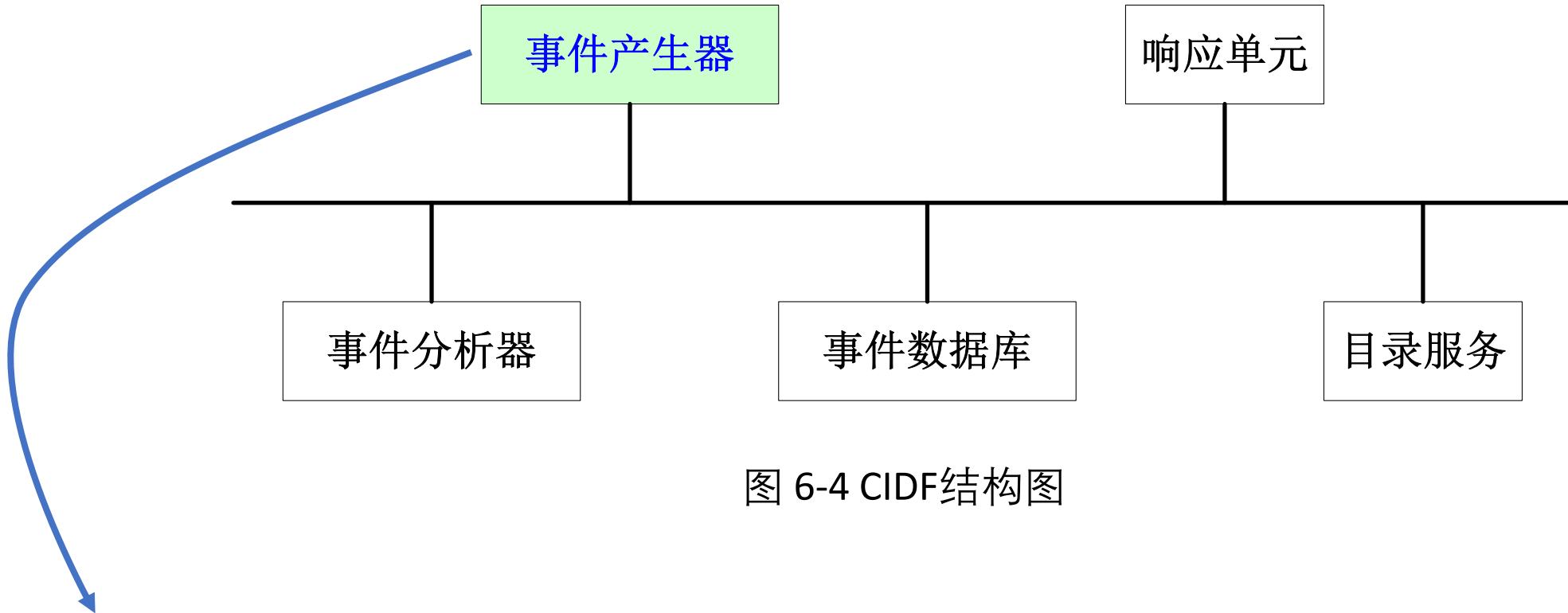


图 6-4 CIDF结构图

- **事件产生器**的任务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。

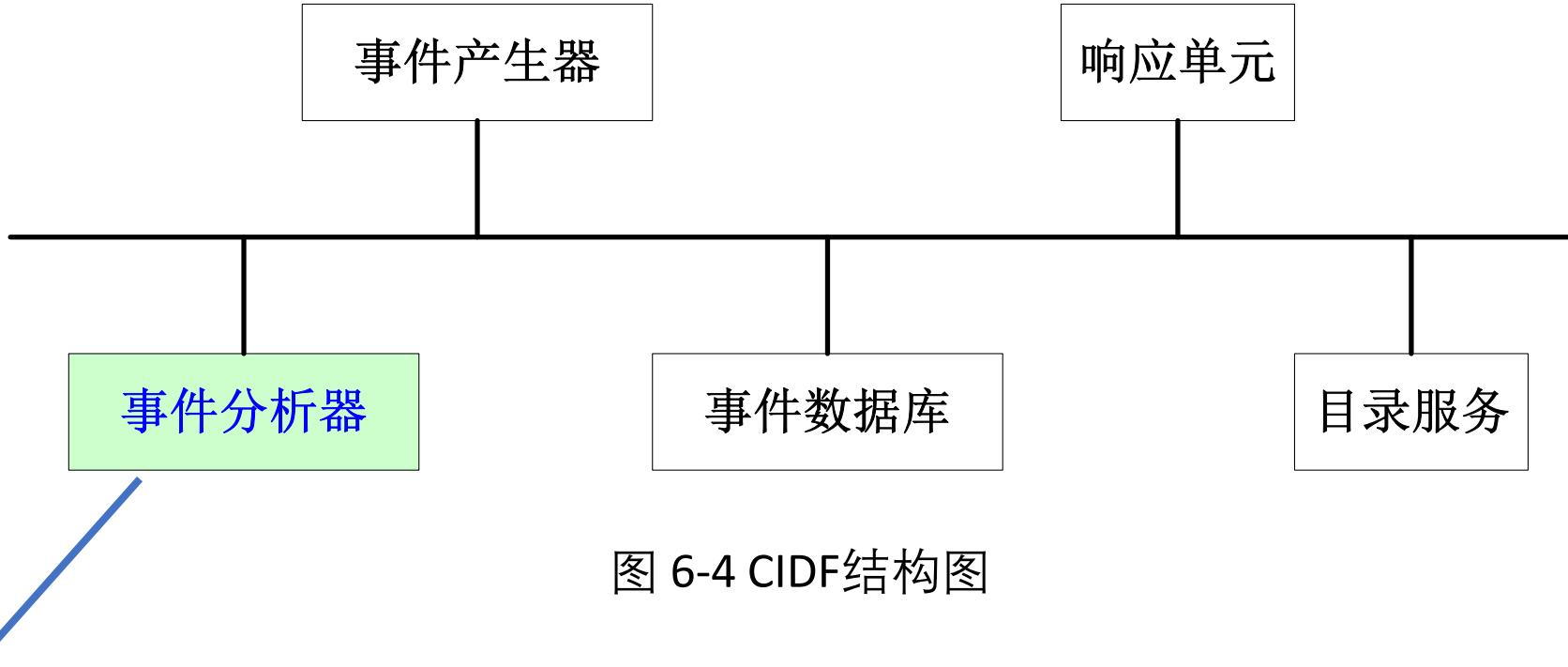


图 6-4 CIDF结构图

- **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，判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。

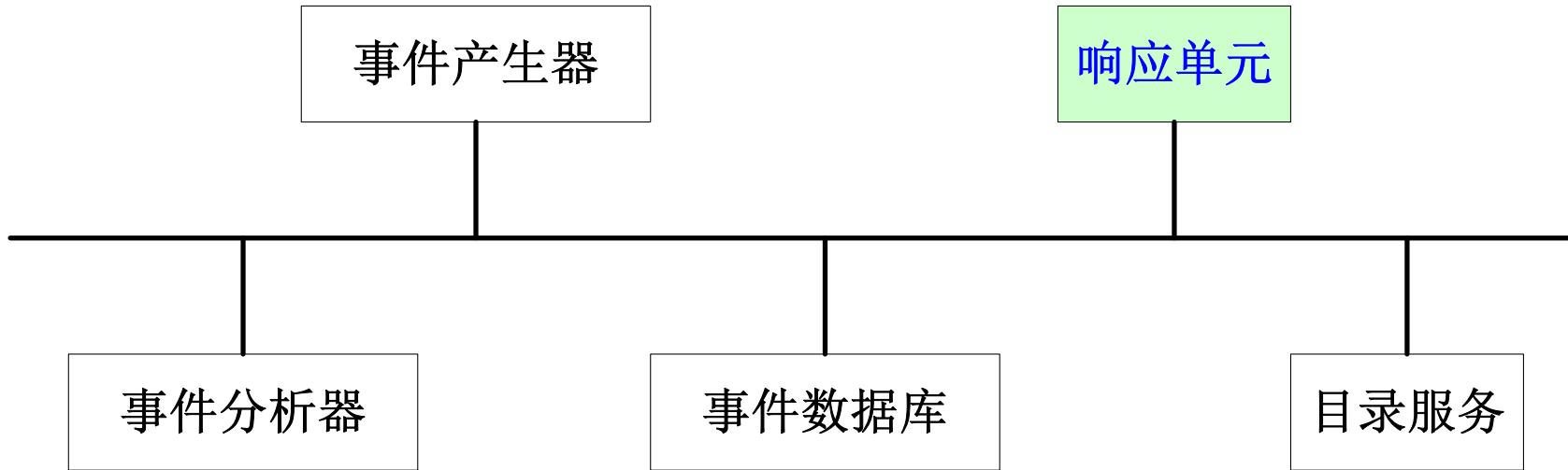


图 6-4 CIDF结构图

- **响应单元**: 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，由管理者决定采取的应对措施。

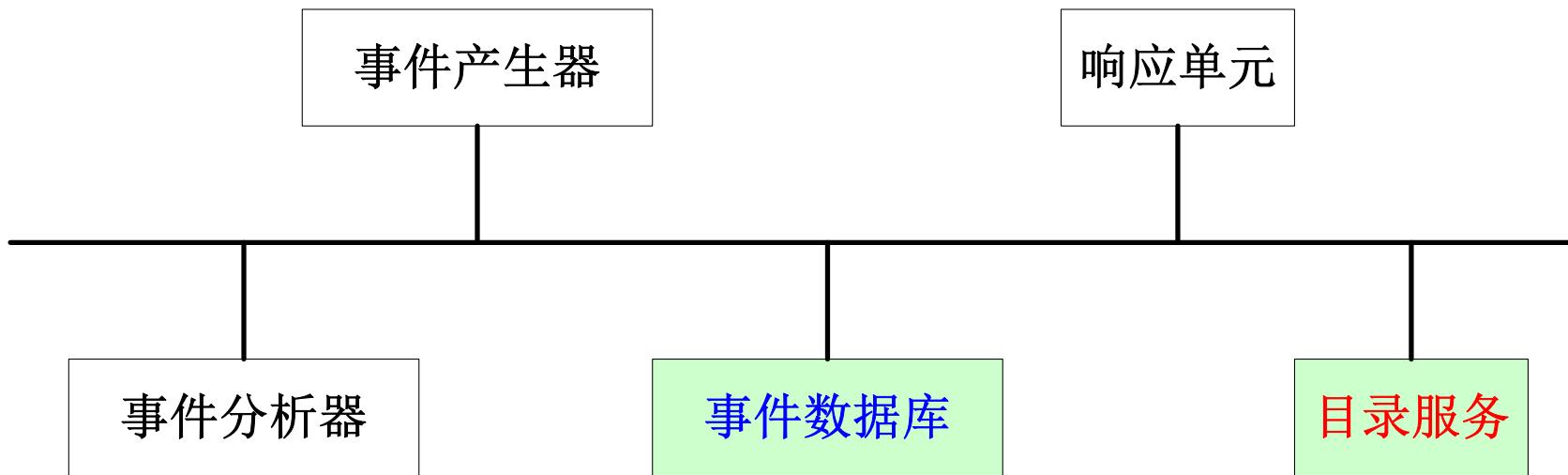


图 6-4 CIDF结构图

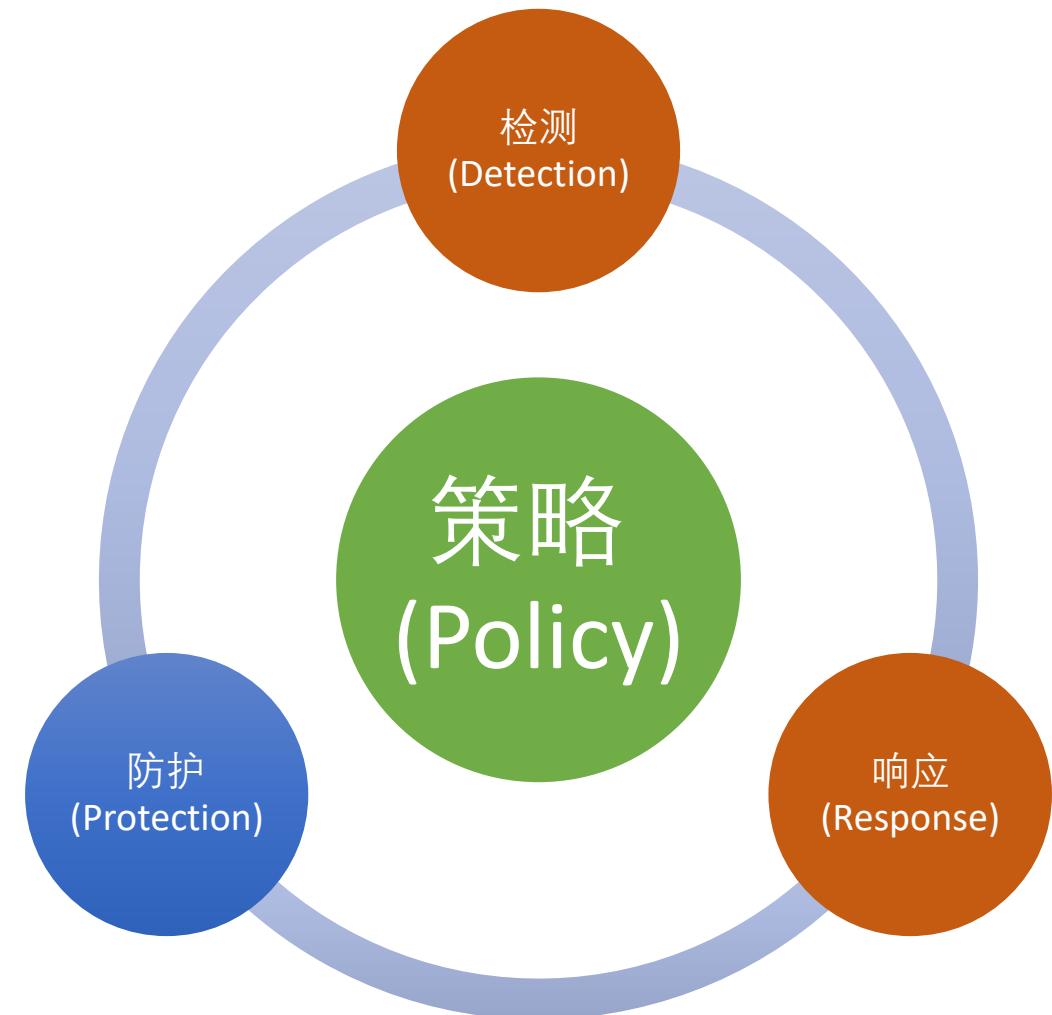
- **事件数据库**是存放各种中间和最终数据的地方的总称，它可以是复杂的数据库，也可以是简单的文本文件。
- **目录服务构件**用于各构件定位其他的构件，以及控制其他构件传递的数据并认证其他构件的使用，以防止IDS系统本身受到攻击。它可以管理和发布密钥，提供构件信息和告诉用户构件的功能接口。

# 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(IDS的三大要素)**称作**入侵检测系统的处理模式**。
- 虽然CIDF具有明显的优点，但它还没有得到广泛的应用，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。

# PPDR模型：一种以安全策略为核心的动力防御模型

- PPDR 是 策 略 (Policy) 、 防 护 (Protection)、 检 测(Detection)和 响 应 (Response)的缩写。
- PPDR模型由于具有动态、自适应的特性，符合计算机安全运行和发展的特点，被越来越多的人所接受。
- 其中，**策略是整个模型的核心**，规定了系统的安全目标及具体安全措施和实施强度等内容。
- **防护**指具体的安全规则、安全配置和安全设备；**检测**是对整个系统动态的监控；**响应**是对各种入侵为及其后果的及时反应和处理。



## 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**基于知识或特征的检测**技术。
- **工作原理**: 假定所有入侵行为和手段(及其变种)都能够表达为模式或特征(签名), 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。误用检测工作模型如图6-5所示。
- 误用检测技术的优点: 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**。

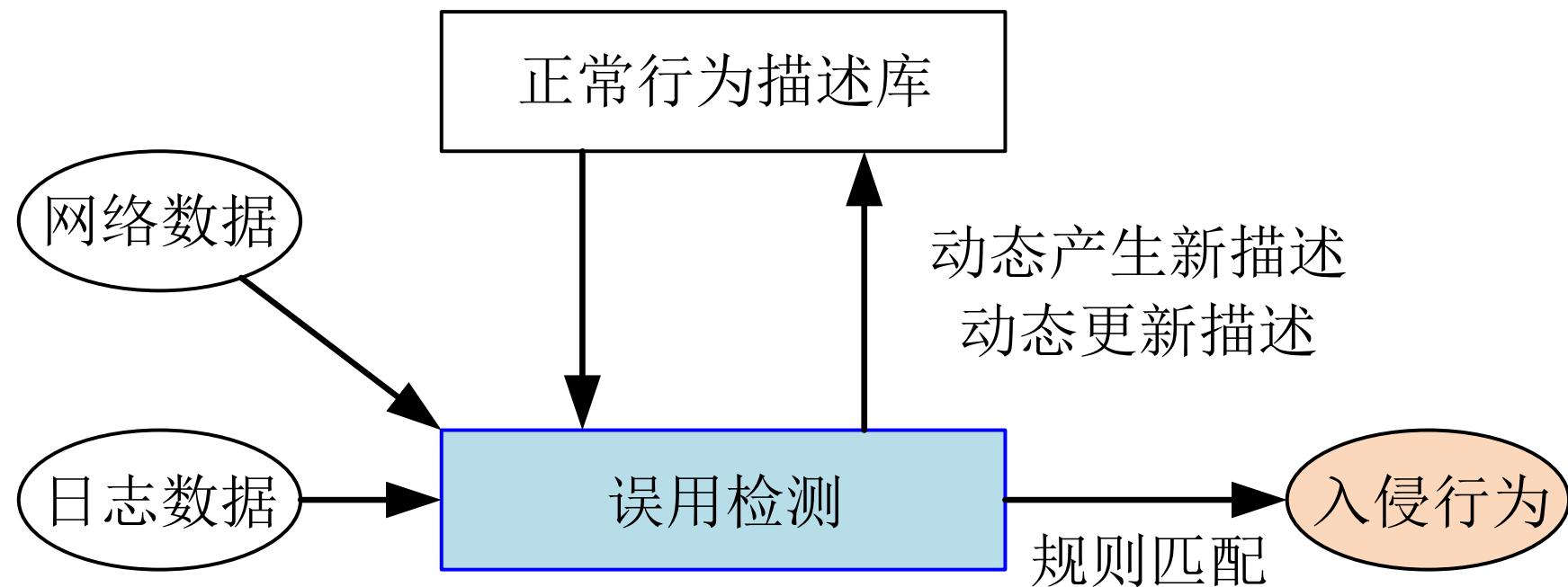


图 6-5 误用检测模型

## (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。
- 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。

## 6.3 基于Snort部署IDS

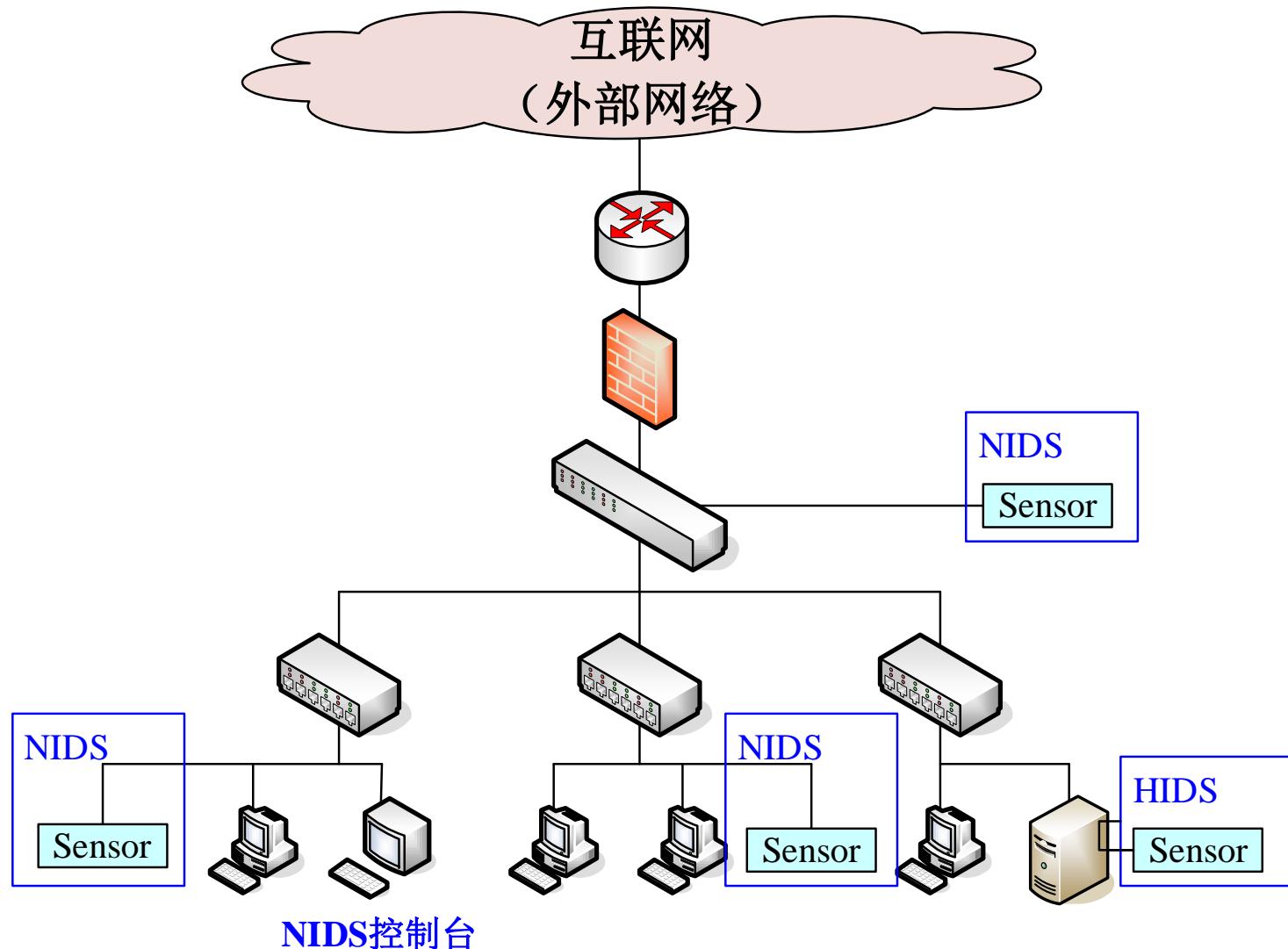


图 6-6 典型的IDS的部署图

# 基于Snort部署IDS

- 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(<https://www.snort.org/>)。
  - ① Copyright c 1998-2003 **Martin Roesch**
  - ② Copyright c 2001-2003 Chris Green
  - ③ Copyright c 2003-2013 Sourcefire, Inc.
  - ④ Copyright c 2014-2020 **Cisco and/or its affiliates**. All rights reserved.

# Snort

- Snort是一个基于libpcap的轻量级网络入侵检测系统，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

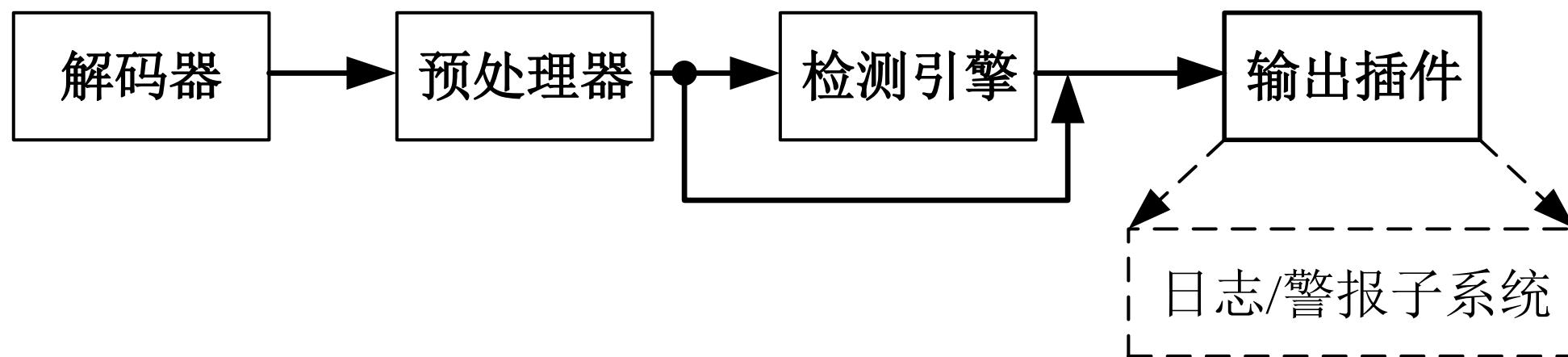
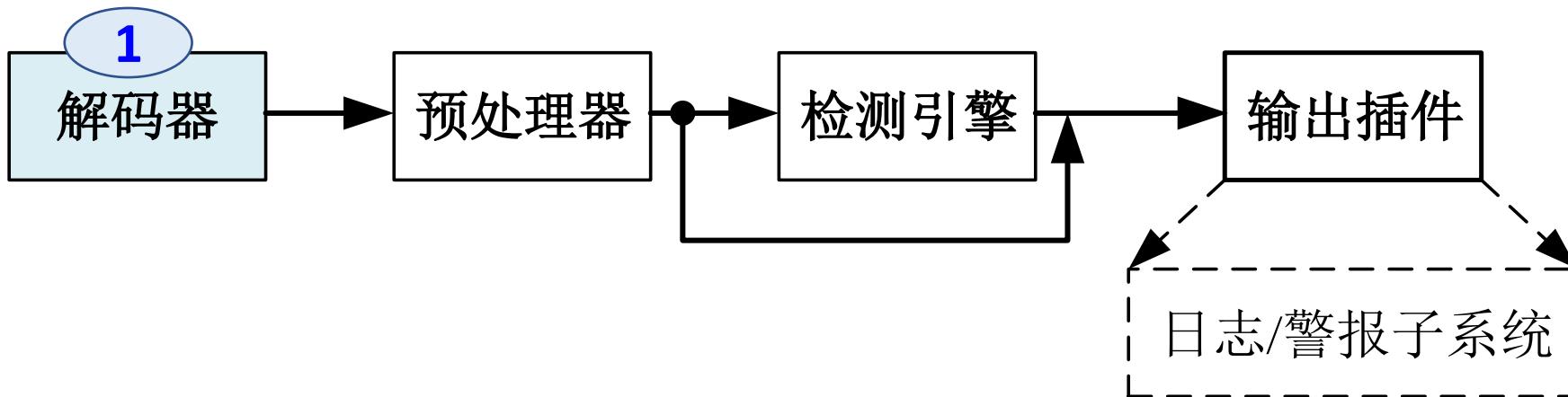
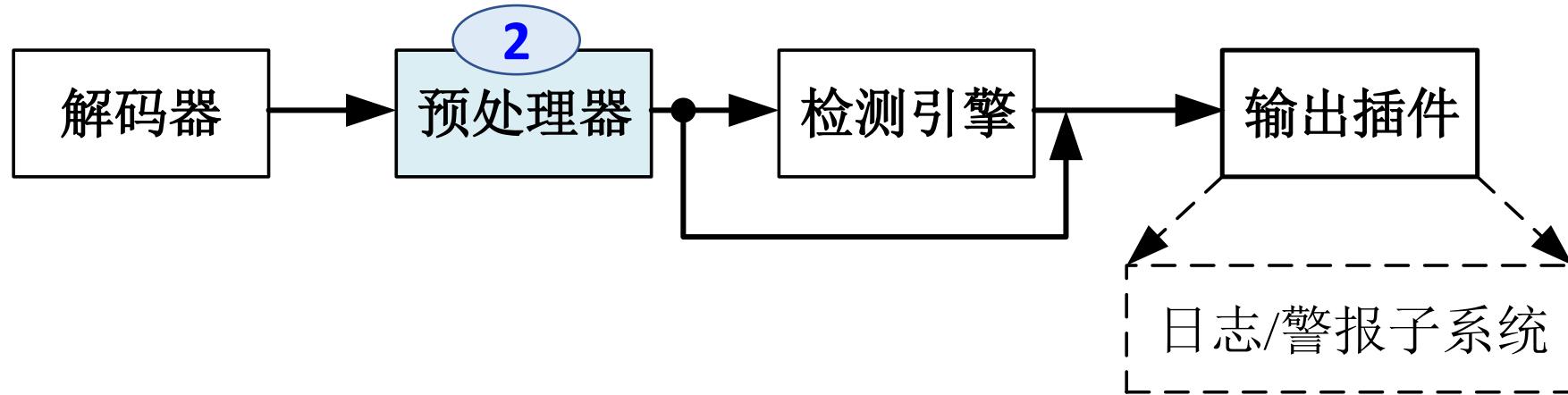


图6-7 Snort的结构



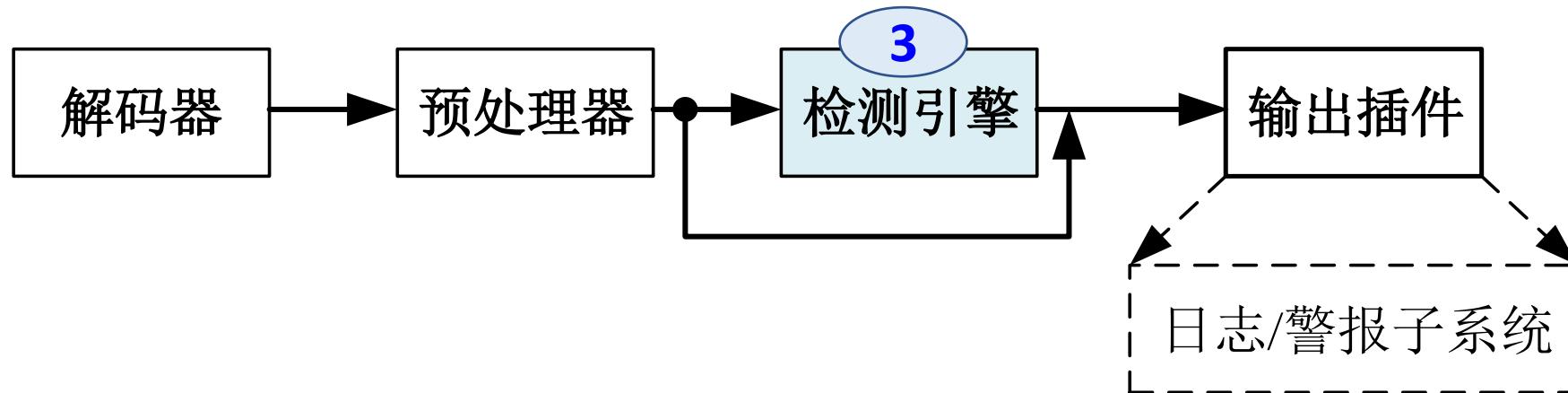
## 1.解码器

- 通过libpcap获得网络数据包之后，数据将通过一序列的解码器。
- 首先填写链路级协议的包结构，然后解码为后续处理所需的信息，如TCP或UDP端口之类的信息。获取的信息将被送往预处理器。
- 解码器支持多种类型的网络接口，包括Ethernet、SLIP、PPP等。



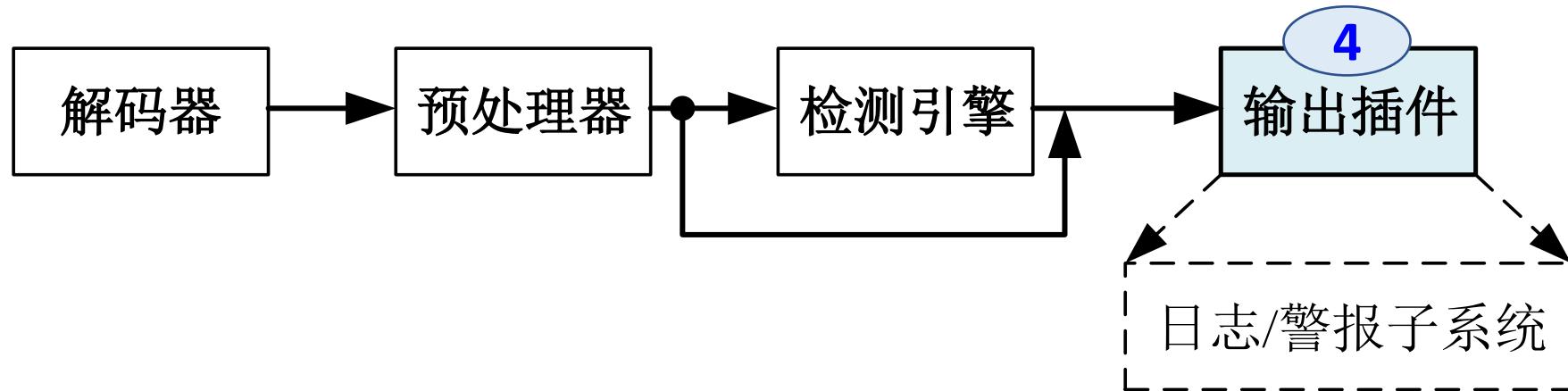
## 2. 预处理器

- 数据包被依次发送到注册过的一组预处理器。每个预处理器都会检查这个数据包，决定是否应该查看。
- Snort中包含了多种**预处理器**，如包重组、协议解码、**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)等，分别实现不同的功能。
- Snort 2.9.16版本内置了24个预处理器，这些预处理器的功能和配置方法详见用户手册(snort manual)的2.2节。
- 用户可以设计自己的预处理器（比如：基于人工智能的入侵检测，可以作为一个预处理器），以扩展入侵检测的功能。



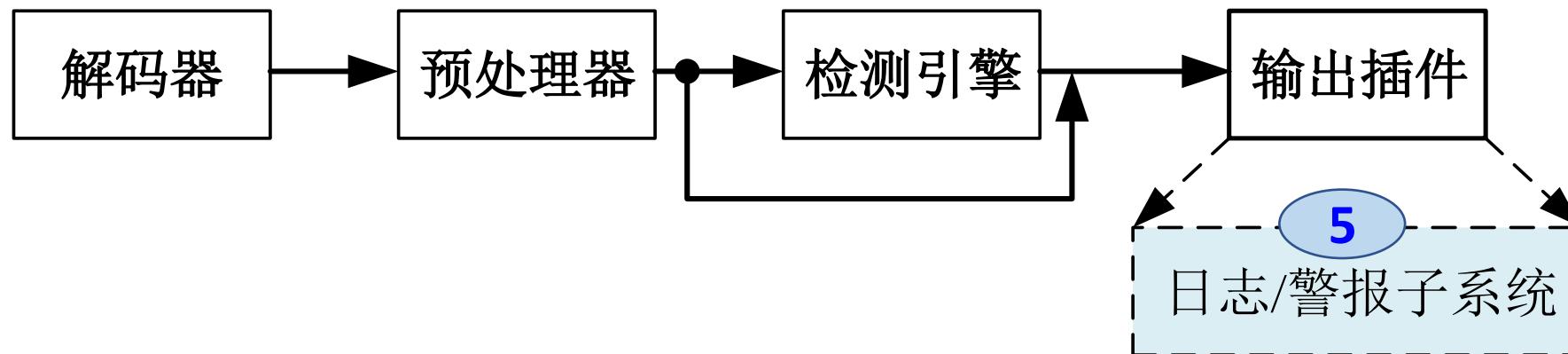
### 3. 检测引擎

- 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(误用检测)**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。



## 4. 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。
- 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。



## 5. 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、tcpdump等格式记录下来；警报子系统将报警信息发送到syslog、socket或数据库中。

# Snort的工作流程

1. 首先，Snort利用libpcap捕获网络数据包。
2. 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。
3. 检测和响应：
  - ① 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；
  - ② 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](http://www.snort.org)下载用户手册。  
截至2025年9月2日，最新的稳定版本为Snort v3.9.3.0
- snort\_manual.pdf从5个方面对Snort进行了详细介绍。  
详见：<https://www.snort.org/documents>

# IDS实例：Snort在ubuntu 24.04系统的使用

- 用ifconfig获取网卡信息，确定需要监视的网络接口名称：

**enp0s3**: 55.55.55.233/8

**enp0s8**: 166.66.66.233/16

**enp0s9**: 192.168.11.24/24

- 安装snort：

**sudo apt install snort**

系统要求输入需要监视的本地网络的IP地址范围，本例输入如下内容：

**192.168.0.0/16,55.0.0.0/8,166.66.0.0/16**

安装之后查看版本信息：

**snort -V**

Version 2.9.20 GRE (Build 82)

# 运行snort

- snort安装完成后将新增snort的相关目录

## whereis snort

/usr/sbin/snort : 可执行文件

/usr/lib/snort : 相关模块所在的目录

/etc/snort : 配置文件和规则库所在的目录

/usr/share/man/man8/snort.8.gz : snort帮助文件

## ll /var/log/snort/

-rw-r----- snort.log : 默认的运行日志文件

- 以NIDS模式运行snort：

在网络接口（网卡）enp0s8监听数据包，使用预定义规则/etc/snort/snort.conf，日志信息存放在./log目录中：

```
sudo snort -l ./log -A fast -c /etc/snort/snort.conf -i enp0s8
```

**snort.conf** is the name of your **snort configuration file**. This will apply the **rules** configured in the snort.conf file to each packet to decide if an action based upon the rule type in the file should be taken.

# 运行中的snort (演示)

Reload thread started, thread 0x7f0502bdf640 (4736)

Decoding Ethernet

--== Initialization Complete ==--

„\_ -\*> Snort! <\*-

o" )~ Version 2.9.15.1 GRE (Build 15125)

"" By Martin Roesch & The Snort Team:  
<http://www.snort.org/contact#team>

Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.

Copyright (C) 1998-2013 Sourcefire, Inc., et al.

Using libpcap version 1.10.1 (with TPACKET\_V3)

Using PCRE version: 8.39 2016-06-14

Using ZLIB version: 1.2.11

Rules Engine: SF\_SNORT\_DETECTION\_ENGINE Version 3.1 <Build 1>

Preprocessor Object: SF\_DNP3 Version 1.1 <Build 1>

Preprocessor Object: SF\_MODBUS Version 1.1 <Build 1>

Preprocessor Object: SF\_REPUTATION Version 1.1 <Build 1>

Preprocessor Object: SF\_POP Version 1.0 <Build 1>

Preprocessor Object: SF\_SSH Version 1.1 <Build 3>

Preprocessor Object: SF\_IMAP Version 1.0 <Build 1>

Preprocessor Object: SF\_DNS Version 1.1 <Build 4>

Preprocessor Object: SF\_SMTP Version 1.1 <Build 9>

Preprocessor Object: SF\_GTP Version 1.1 <Build 1>

Preprocessor Object: SF\_DCERPC2 Version 1.0 <Build 3>

Preprocessor Object: SF\_FTPTELNET Version 1.2 <Build 13>

Commencing packet processing (pid=2656)

# Snort检测入侵并告警的实例

1. 从Windows发动对Windows 2000 RPC漏洞的攻击。
2. 在snort所在的虚拟机（另一个终端）中查看入侵告警信息

```
ls -l
```

```
total 8  
-rw-r--r-- 1 root root 412 Oct 28 15:13 alert  
-rw----- 1 root root 1328 Oct 28 15:13 snort.log.1761635479
```

告警信息存放在文件alert中

```
cat ./alert
```

```
10/28-15:13:42.906612 [**] [1:3276:2] NETBIOS DCERPC IActivation little endian bind  
attempt [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP}  
166.66.0.128:1035 -> 166.66.0.140:135  
  
10/28-15:13:42.906612 [**] [1:2251:14] NETBIOS DCERPC Remote Activation bind  
attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP}  
166.66.0.128:1035 -> 166.66.0.140:135
```

尝试获得管理员权限

# 6.4 IDS的发展方向

## (1) 宽带高速实时检测技术

- 大量高速网络技术(如千兆、万兆以太网等)在近年相继出现并普及。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。
- 目前主流的千兆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(安全关键点)放置多个IDS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，需要研究大规模分布式的入侵检测技术。

### (3) 面向IDS的数据挖掘技术

- 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。
- 将数据挖掘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。

# 数据挖掘技术应用于IDS

- 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，国际上在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。

## (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。如计算机免疫技术、神经网络技术和遗传算法。

### 1) 计算机免疫技术在入侵检测中的应用

- 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。

# 更先进的检测算法

## 2) 神经网络技术（深度学习）在入侵检测中的应用

- 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。
- 现在提出了多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。

- 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。
  - **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；
  - **攻击对抗** 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。
- 攻击对抗的主动防御能力是综合的网络空间安全能力的体现，需要多种技术的有机集成才可以完成。

## (6) 与其他安全技术的结合

- 随着黑客入侵手段的提高，尤其是分布式、协同式、复杂模式攻击的出现和发展，传统的缺乏协作的单一IDS已经不能满足需求，需要有充分的协作机制。
- 所谓协作，主要包括两个方面：事件检测、分析和响应能力的协作，各部件所拥有的安全相关信息的共享。协作的层次主要有以下几种：
  - 1) 同一系统中不同入侵检测部件之间的协作，尤其是主机型和网络型入侵检测部件之间的协作，以及异构平台部件的协作。
  - 2) 不同安全工具之间的协作。
  - 3) 不同厂商的安全产品之间的协作。
  - 4) 不同组织之间预警能力和信息的协作。

# 与其他安全技术的结合(安全技术联动)

- 此外，单一的入侵检测系统并非万能，因此，需要结合身份认证、访问控制、数据加密、防火墙、安全扫描、PKI技术、病毒防护等众多网络安全技术，来提供完整的网络安全保障。
- 总之，入侵检测系统作为一种主动的安全防护技术，提供了对内部攻击、外部攻击和误操作的实时保护。随着网络通信技术对安全性的要求越来越高，为给电子商务等网络应用提供可靠服务，入侵检测系统的发展，必将进一步受到人们的高度重视。
- 未来的入侵检测系统将会结合其他网络管理软件，形成**入侵检测**、**网络管理**、**网络监控**三位一体的工具。
- 网络**安全态势感知**系统：
  - 可以看成是基于分布式入侵检测系统的综合安全监控系统，具有入侵检测、安全状态可视化展示、安全状态理解及趋势分析预测，以及网络监视和网络控制等功能。

# 安全联动技术实例：本课题组的三维融合网络入侵自动阻断系统

The screenshot shows the main interface of the 'Three-dimensional integrated network intrusion automatic blocking system'. The top navigation bar includes tabs for 'System Management', 'Attack Alert and Firewall Rule Management', 'Vulnerability Information', 'SMS Management', 'Device Management', 'Basic Data', 'Statistics and Analysis', and 'Audit Information'. The main dashboard displays real-time statistics: 'Today's Attack Total: 12772' (Low Risk: 2753, Medium Risk: 2762, High Risk: 6791), a flowchart showing the data flow from an IDS to a security platform and then to a firewall, and 'Firewall Policy Total: 1924' (Pending Execution: 1924, Executed: 0, Abandoned: 0). Below the dashboard is a 'Query Conditions' section with dropdowns for 'All' or 'Selected', a search button, and two radio buttons for 'Show all alert information' or 'Only show associated vulnerability alert information'. A progress bar indicates 'Refreshing cycle (seconds): 600' and a green status bar says 'Updating current alert information'. A large table titled 'Alert Rolling List' shows 12772 records, each with columns for event time, source IP, source port, destination IP, destination port, event description, risk level, and related ID. At the bottom of this section is a pagination bar with links for 'First', 'Previous', '1/128', 'Next', and 'Last'. The bottom half of the interface is a 'Firewall Policy Generation' section with similar query conditions and a table showing 1924 policy rules, each with columns for selection, firewall identifier, interface IP, command action, rule sequence, and command content. A pagination bar at the bottom of this section also shows '1/20'.

通过漏洞扫描、入侵检测、资产审计发现高风险入侵，并自动生成防火墙命令（防火墙规则）。

## 6.5 NIDS的脆弱性及攻击方法

- IDS虽然在一定的程度上提高了系统的安全性，但是由于其固有的脆弱性，存在被攻击的可能性。
- 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：
  - A. 规避NIDS的检测；
  - B. 针对NIDS自身发起攻击，使其无法正常运行；
  - C. 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保护的是一个网络，其数据流量通常会**比单机高出一到两个数量级以上**，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(异常检测方法)**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中发挥令人满意的功效。因此，特征匹配(**MD，误用检测方法**)成为NIDS分析引擎的一个不可或缺的功能模块。
- 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。**
- 特征匹配是脆弱的，这种脆弱性是固有的、可以在某个时间降低却不可以根除。事实上，目前很多Anti-NIDS技术都是针对特征匹配脆弱性的。

### (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二维分布。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、用箭头获得上一次输入的命令)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实现，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（比如是否计算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。

谢谢！