基于CSCW的分布式入侵检测模型研究

苏 磊,傅秀芬,陈柏兴,吴 丹,陈长瑶
(广东工业大学 计算机学院,广东 广州 510075)

摘 要:分布式入侵检测技术是目前安全检测领域的研究热点之一。在分析了现有分布式入侵检测系统所存在问题的基础之上,基于CSCW的原理,提出了一种新的分布式入侵检测系统模型。该系统模型采用CSCW概念重新构造系统的检测组件,通过协同机制和安全通信,实现了分布式入侵检测系统各个检测组件之间的数据共享、知识共享和负载均衡,解决了分布式系统检测组件之间缺乏有效协作和信息共享的问题,避免了关键节点的处理瓶颈,提高了系统的协同检测能力和资源利用率。

关键词:计算机支持的协同工作;分布式入侵检测;信息共享;协同分析
中图分类号:TP393.08 **文献标识码**:A **文章编号**:1673-629X(2008)05-0150-03

A Model Research of Distributed Intrusion Detection Based on CSCW

SU Lei, FU Xiu-fen, CHEN Bai-xing, WU Dan, CHEN Chang-yao
(School of Computer, Guangdong University of Technology, Guangzhou 510075, China)

Abstract: Distributed intrusion detection is one of the hotspots of security detection field nowadays. Based on the analysis of the existing problems of current distributed intrusion detection system and the principle of CSCW, a kind of new distributed intrusion detection model is proposed. The components of the model are newly created using CSCW concept. By cooperation mechanism and security communication, the model actualizes data share, knowledge share and load balancing among components of distributed intrusion detection system, avoids disposal bottle-neck of critical nodes, and improves the ability of cooperative detection and the availability of system resources.

Key words: CSCW; distributed intrusion detection; information share; cooperation analysis

0 引 言

互联网的快速发展为全球范围内实现高效的资源和信息共享提供了方便,同时也对网络安全和入侵检测系统(IDS)提出了新的挑战。网络结构的日益复杂,海量存储和高带宽传输技术的普及,以及当前网络攻击手段的复杂化,使得传统的分布式入侵检测系统越来越满足不了系统的安全需求[1],这表现在以下几个方面:

(1)检测组件之间缺乏协作和信息共享,只局限于所在的管理域;

(2)检测数据来源和检测方法比较单一;

(3)结构设计缺少统一的标准,限制了扩展性和交互性。

针对上述问题,引入了计算机支持的协同工作(CSCW)的概念,提出了一个基于CSCW的分布式入侵检测系统模型。该系统模型采用分布式信息收集、分布式信息处理,在各个检测组件之间实现了信息共享和协同分析,提高了对分布式协同攻击的检测能力,而且模型采用通用的告警信息格式和安全通信协议,提高了系统的扩展性和可靠性。

1 CSCW

1.1 CSCW的定义

CSCW即计算机支持的协同工作,是在计算机技术支持的环境下(即CS),特别是在计算机网络下,一个群体协同工作完成一项共同的任务(即CW)[2]。

在分布式入侵检测系统(DIDS)中的CSCW可以这样理解:针对复杂的入侵行为,分布在不同管理域的检测组件,在一定网络环境(有线或无线)的支持下,相互协作,共同完成一个入侵检测任务。其中,有效的数据共享是分布式协同检测成功的前提;而检测过程中用于交流、协调、决策等的数据和知识的协同管理是分布式协同检测成功的保证。

收稿日期:2007-08-24
基金项目:广东省自然科学基金项目(06021484)
作者简介:苏 磊(1981-),男,山东淄博人,硕士研究生,研究方向为入侵检测、网络安全、计算机协同;傅秀芬,教授,硕士生导师,研究方向为网络安全、计算机协同、网络多媒体技术。

1.2 CSCW在入侵检测中的应用

将CSCW应用于分布式入侵检测中,主要目的是通过协作为系统组件之间提供多种信息共享方式,增加单个检测组件可用的数据和资源,从而提高系统的检测效率和准确度。

通过引入CSCW,可提供的信息和资源的共享方式如下:

(1)数据共享。

一个节点IDS能直接获取的数据是有限的,当单个节点IDS没有足够的数据判断是否存在入侵时,可以向其他节点IDS请求协作,从其他管理域收集一些相关数据,以便集成分散的可疑数据,还原出入侵迹象[2,3]。

(2)知识共享。

单个IDS所检测到的入侵种类是有限的,当一个节点IDS察觉到可疑事件时,可以请求与其他节点IDS协作,利用其他节点IDS的检测方法和入侵规则来确定是否存在入侵以及入侵的类型。此外,不同节点IDS之间还可以共享新的入侵类型与检测规则。

(3)负载均衡。

节点IDS可能因为系统繁忙而顾不上对可疑事件进行深入分析,因而可能漏掉安全事件。此时,负载重的节点IDS可以请求负载轻的节点IDS分担一些检测或者分析任务。这样,各节点IDS的负载不至于有太大的差异,提高了整个系统的资源利用率[3]。

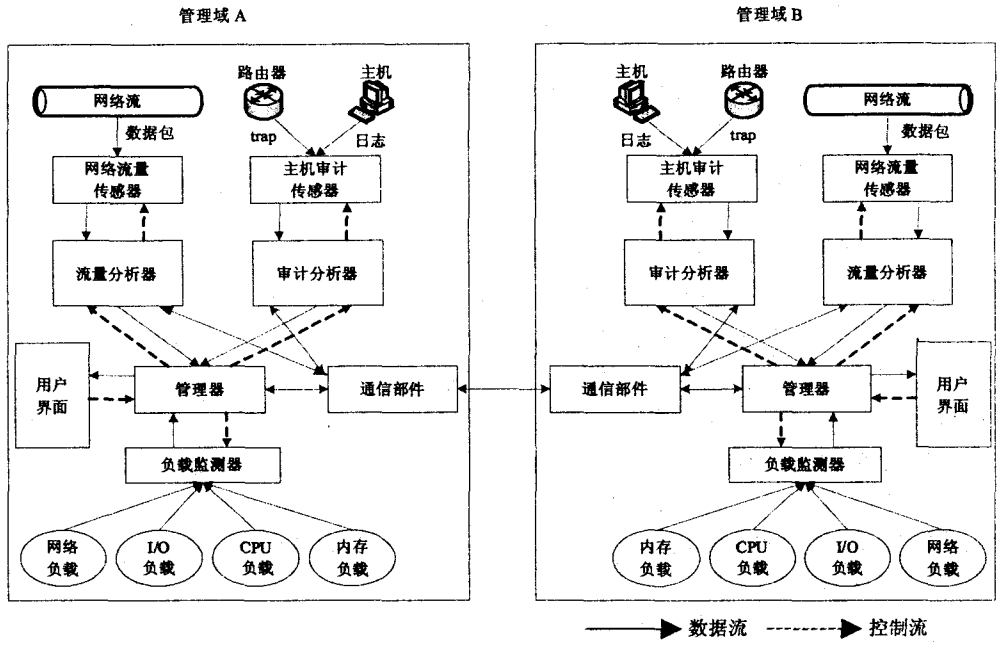

图1　基于CSCW的分布式入侵检测模型

2 基于CSCW的分布式入侵检测模型

基于CSCW的分布式入侵检测模型包括的组件有:传感器、分析器、负载监测器、管理器、通信部件和用户界面,如图1所示。以下分别介绍各个功能组件的结构和工作原理。

2.1 传感器(Sensor)

传感器有主机审计传感器和网络流量传感器,主要用于采集主机审计日志、路由器trap信息和网络流量等原始数据。采集的数据经过过滤、格式转化等简单预处理后,发送给分析器。

2.2 分析器(Analyzer)

分析器是整个入侵检测系统的功能核心,由消息接受器、分析引擎、规则生成器和控制器组成,其结构如图2所示。分析引擎从消息模块接受传感器发送来的数据,根据规则库中的入侵规则判断可疑事件的发生。如果分析引擎根据本地入侵规则检测到入侵行为,则直接向管理器发送报警信息;如果根据所获得的数据和本地规则不能确定可疑事件是否为入侵行为,则发送协作请求给管理器,管理器再根据协同策略,向其他管理域的节点IDS请求协同分析。分析器可以根据本地管理器下发的协同命令,对其他域的检测任务提供协助。在和其他管理域进行通信的时候需要通过通信部件把数据或者消息的格式化为所有节点IDS都

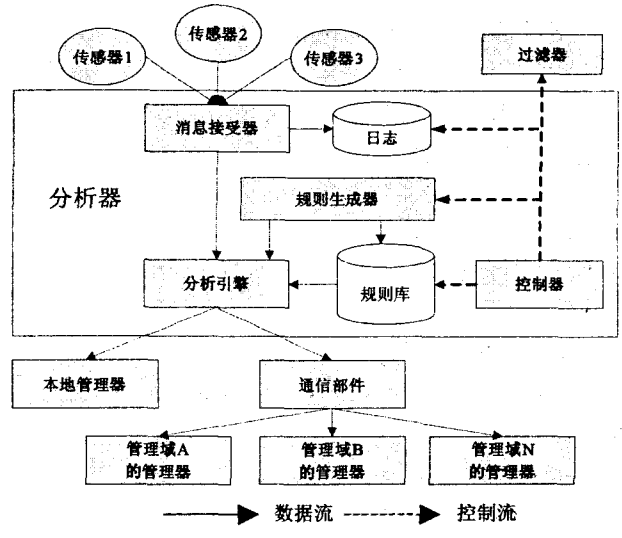

图2　分析器结构

能识别的统一标准格式。

控制器负责执行管理器发来的命令，包括对分析引擎和规则生成器的配置管理、消息日志的维护、规则库的增删改，以及对传感器设置过滤规则，使之只采集与被保护对象相关的数据。

2.3 负载监测器(Load Monitor)

负载监测器负责监测本节点计算机的负载变化情况，它按一定的时间周期监测 CPU 使用率、I/O 负载、内存使用率以及网络负载，并将这些数据进行综合，得出节点计算机的负载值，发送给管理器。负载值的计算使用文献[4]中提到一种动态负载权值公式。

2.4 管理器(Manager)

管理器是分布式入侵监测组件之间进行协同管理的关键部件，主要由策略模块、响应模块和协作节点计算机列表等部分组成。

管理器一方面根据负载监测器发送来的该节点负载值，来确定该节点 IDS 的对外协作能力。如果负载过重，则协作能力弱，且达到一定阈值，需要向其他管理域的节点 IDS 发送分担检测任务的协助请求。如果负载比较轻，则协作能力强，在不妨碍本地检测任务的情况下，可以向其他管理域的节点 IDS 提供协助。协作能力有两个指标：协作工作量和协作持续时间。

管理器另一方面接受分析器或其他域的管理器发送来的消息，如果是告警信息，则根据既定的响应策略来决定告警事件的响应动作，包括向操作员报警、终止当前活动、向其他管理器转发等；如果是协作请求信息，则根据既定的协同策略决定向其他管理域或本地分析器发送相应的协作消息，包括相关数据的协作收集、事件的协助检测等；如果是其他域发来的规则更新信息，则根据本地规则库的情况，决定是否接受新规则入库。

2.5 通信部件(Communication)

通信部件的功能主要有以下几个方面：

(1)为消息提供路由服务，识别并接受本系统中使用的定向包和组播包。

(2)通过 IDMEF_XML_Plugin 将本系统发向其他域的交互信息和数据转化成统一的标准格式 IDMEF，并负责解析其他域传送过来的 IDMEF 信息。

(3)通过前端组件接受其他域发来的信息后，经过过滤器对信息进行分类，明确信息类型和取得信息相关的解析文档，查询信息源是否注册，并验证信息的可靠性。

2.6 用户界面(User Interface)

用户界面是管理员和节点 IDS 之间的交互平台。用户界面以直观的形式显示告警事件，并且指出入侵事件的类型和可能的危害；另外还反映当前的网络活动和检测对象状态。管理员可以通过用户界面对节点 IDS 进行一些配置操作和入侵规则的手工更新。

3 通信格式与协议

在分布式 IDS 中，负责监控不同管理域的节点 IDS，可能运行于不同的平台上，并具有不同的数据格式。为确保各个节点 IDS 之间能正确理解相互间传递的各种数据的语义，以实现各个节点之间的数据传输与交互，使用入侵检测消息交换格式(IDMEF)[5]来表示不同类型 IDS 输出信息的数据模型，以提高系统之间的互操作性。IDMEF 定义了两种消息类型：Alert 和 HeartBeat。前者用于传递报警信息，后者用于报告传感器的状态。

针对系统模型的需求，对 IDMEF 进行了扩展，增加了管理器与管理器、管理器与分析器之间的协作通知消息类型(Notification)来描述协查请求、协查回复、攻击状况通报等消息。

通信协议采用入侵检测交换协议(Intrusion Detection Exchange Protocol, IDXP)[6]来实现入侵检测组件之间交换数据，该协议提供面向连接协议之上的双方认证、完整性和保密性等安全特性。

4 结 语

把 CSCW 理论与分布式入侵检测技术相结合，提出了一种新的分布式入侵检测系统模型。和已有的系统相比，该模型以及对应的系统能够更好地适应大规模异构网络环境下的入侵检测，在一定程度上解决了分布式入侵检测系统组件之间缺乏必要的协作和信息共享、数据来源和检测方法单一、系统扩展性和交互性差等问题，提高了对分布式协同攻击的检测能力。下一步研究工作是：分析因为协作和数据共享而产生的风险问题，引入风险控制策略，提高分布式入侵检测系统的安全性和可靠性。

参考文献：

[1] 连一峰，戴英侠，胡 艳，等. 分布式入侵检测模型研究[J]. 计算机研究与发展，2003，48(8)：1195－1202.

[2] Dong Y－L，Qian J，Shi M－L. A cooperative intrusion detection system based on automous agents[C]//CCECE'2003. Montreal：[s. n.]，2003：861－864.

[3] 史美林，董永乐，钱 俊，等. CSCW 支持下的协同入侵检测[C]//第三届全国 CSCW 暨第一届全国 AIN 学术会议论文集. 呼和浩特：[出版者不详]，2002：9－14.

[4] Alam M S，Javed Q. Adaptive load balancing architecture for

(下转第 175 页)

Bean)进行接收,处理完后将订单标志设为"received",完成一次交易。下面仅给出主要模块的代码:

(1)顾客选中商品后在购物篮中创建一个订单:

```
public String buy() {
try {
OrderHome home = getOrderHome();
String orderID = makeUniqueID();
Order order = home.create(orderID, owner, "unverified", subTotal, taxes);// 创建一个订单对象
Vector orderItems = addOrderLineItems(order); //创建订购商品列表对象
order.setLineItems(orderItems); //将 orderlineitems 存入订单中
sendMessage(orderID); //将 orderID 作为 JMS 消息发送到主题,此方法下面将会介绍到
return orderID;
} catch (Exception e) {
throw new EJBException(e); } }
```

(2)为发送一个消息到主题创建所有必要的对象:

```
public void init(Context ctx, String topicName) throws NamingException, JMSException{
tconFactory = (TopicConnectionFactory)ctx.lookup(JMS_FACTORY); //从上下文环境查找连接工厂
tcon = tconFactory.createTopicConnection(); //创建连接工厂
tsession = tcon.createTopicSession(false, Session.AUTO_ACKNOWLEDGE);//创建主题会话
topic = (Topic)ctx.lookup(topicName);//从上下文环境中查找队列
tpublisher = tsession.createPublisher(topic);//创建发布者
msg = tsession.createTextMessage();//创建文本消息
tcon.start(); //启动连接}
```

(3)发送一个消息到主题:

```
Public void sendMessage(String message) throws JMSException {
try{
InitialContext ctx = getInitialContext(URL);
init(ctx, TOPIC);
send(message); //调用 send()方法
}catch(Exception e){
throw new JMSException( e.getMessage());
}finally{ close(); } }
public void send(String value) throws JMSException {
//发送一个消息到主题
msg.setText(value);
tpublisher.publish(msg); }
public void close() throws JMSException {
//关闭 JMS 对象
tpublisher.close();
tsession.close();
tcon.close(); }
```

(4)接收消息:

```
public class OrderDisposeBean implements MessageDrivenBean, MessageListener {
//这里消息驱动 Bean 就相当于一个消息监听器,实现消息监听接口,当消息到达后,会自动调用其中的 onMessage()方法
protected MessageDrivenContext ctx;
public void onMessage(Message msg) {
TextMessage tm = (TextMessage) msg;
try {
String orderID = tm.getText();
Context ctx = new InitialContext();
OrderHome home = (OrderHome) PortableRemoteObject.narrow (ctx.lookup("OrderHome"), OrderHome.class); //远程调用资源
Order order = home.findByPrimaryKey(orderID);
order.setStatus("received"); }
catch (Exception e) {
e.printStackTrace();
throw new EJBException(e); } }
```

4　结束语

消息队列中间件对企业分布式应用系统的开发具有很大的吸引力,它向开发者展现了一种灵活、丰富且非常简单的通信模式。JMS提供了一组与具体实现无关的接口,各种分布式应用程序可通过这组接口来访问支持JMS的消息中间件。现在JMS技术已得到了广泛的工业支持,有着广阔的前景。

参考文献:

[1] 冯　磊. JMS给中间件市场加热[J]. 信息系统工程,2007(1):86-88.

[2] 王小霞,陈　亮. 一种消息队列中间件的设计与实现[J]. 计算机工程,2005,31(21):81-83.

[3] 徐　晶,许　玮. 消息中间件综述[J]. 计算机工程,2005,31(16):73-76.

[4] Momson-Haefel R, Chappell D A. Java Message Service[M]. [s.l.]:O'Reilly,2001.

[5] J2EE. Java Message Server(JMS)[EB/OL]. 2007. http://java.sun.com/products/jms.

(上接第152页)

SNORT[C]//INCC 2004. Lahore, Pakistan:[s.n.],2004:48-52.

[5] The Intrusion Detection Message Exchange Format, draft-ietf-idwg-id-mef-xml-12[S/OL]. 2005-04. http://www.ietf.org.

[6] The Intrusion Detection Exchange Protocol (IDXP), draft-ietf-idwg-beep-idxp-07[S/OL]. 2005-04. http://www.ietf.org.