第33卷 第14期 计 算 机 工 程 2007年7月
Vol.33 No.14 Computer Engineering July 2007
·安全技术· 文章编号：1000—3428(2007)14—0145—03 文献标识码：A 中图分类号：TP311

基于 Web Services 接口的信息安全综合审计系统

赵 艳[1,2]，翟伟斌[2]，杨泽明[2]，许榕生[2]
(1. 燕山大学信息工程学院，秦皇岛 066044；2. 中国科学院高能物理研究所计算中心，北京 100049)

摘 要：设计了一套信息安全综合审计系统，对系统的结构体系和审计功能进行了分析。利用 Web Services 及其相关技术 XML、SOAP、WSDL、UDDI 等，并结合一些其他成熟的协议对综合审计系统的接口部分进行研究，实现系统的重用性和松散耦合性，使其能够通过网络被用户方便快捷地调用。

关键词：Web services；审计；XML；SOAP 协议

Compositive Information Security Audit System Based on Web Services Interface

ZHAO Yan[1,2], ZHAI Weibin[2], YANG Zeming[2], XU Rongsheng[2]
(1. Information Engineering College, Yanshan University, Qinhuangdao 066004;
2. Computing Center, Institute of High Energy Physics, Chinese Academy of Sciences, Beijing 100049)

【Abstract】 A comprehensive information security audit system is designed, and system structure and audit function are analyzed. By studying the Web services and correlated technologies such as XML, SOAP, WSDL, UDDI, and other protocols, this paper describes the interface of the audit system. This audit system is a LCS, and can be reused and called by other users conveniently and frequently via network.

【Key words】 Web services; audit; XML; SOAP protocol

现在的防火墙、入侵检测和漏洞扫描等多种安全产品都会产生和收集大量的日志信息，但这些安全产品所收集的数据大多数都是噪音，因此，在海量日志中提取出有用的信息进行研究与分析，对于及时发现各种入侵倾向、攻击来源等具有重大的意义。有必要建立一套基于 Web services 的信息安全综合审计系统对各种安全产品产生的日志进行分析，以增强各产品之间的协调性，最大程度地发挥其安全作用，提高整个系统的健壮性。

1 综合审计系统的总体架构

1.1 系统结构

本系统设计的是一个综合的信息安全审计系统，安全审计并不仅仅是简单的“日志记录”分析，而是全方位、跨平台、分布式的监控和取证系统，研究包括主流的主机操作系统如 Windows、Linux、IBM AIX、HP-UX、SUN Solaris 等，以及主流的网络设备日志、主流数据库如 Oracle、MS SQL Server、Sybase、Informix、DB2 等。系统整体结构见图 1。

信息安全综合审计系统控制台通过 SNMP 协议实现与各操作系统的主机、主要网络设备及典型应用系统(如 Web, E-mail)之间的通信，设计一个动态 MIB(management Information base)描述管理的各组对象，可以查看管理对象的日志记录和接收来自代理的安全警告，通过 Internet 对管理对象进行数据收集并传输到主控制台。利用 ODBC 技术与数据库系统提供的 API 相结合，实现多个和多种数据库系统统一的审计和监控管理。通过统一的控制中心对多个数据库系统进行监控，集中式的管理方式保证了对数据库审计的高效性、方便性，实时记录远程和本地数据库的审计日志，监视用户对连接数据库的操作行为。采用数据库系统审计与动态创建触发器相结合的技术进行记录级和字段级的审计。

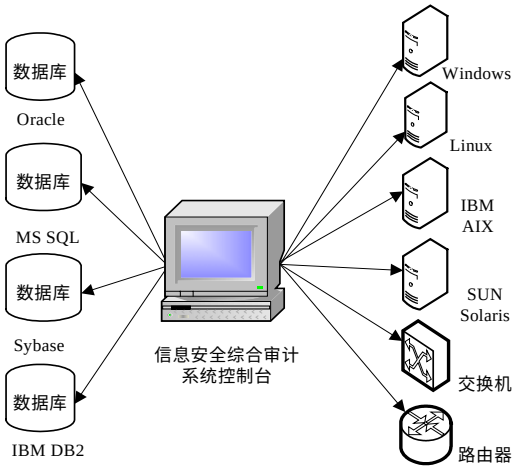

图 1 系统结构示意图

1.2 系统功能模块

基于 Web services 的信息安全综合审计系统除了具有一般的审计与取证系统的基本功能如系统管理、系统审计、报表统计、报表自动生成等，还具有重用性和松散耦合性，更好地发挥了系统本身的功能，符合互联网发展的要求。图 2 是系统功能构架。

系统管理模块分层次地管理审计的各个部分，对来自各种应用程序、操作系统和网络设备的日志信息进行实时采集和取证，根据预定义对海量数据进行过滤和压缩以提高分析

作者简介：赵 艳(1980－)，女，硕士研究生，主研方向：信息安全，安全审计；翟伟斌，博士研究生；杨泽明，博士；许榕生，研究员、博士生导师

收稿日期：2006-08-30 **E-mail：**z_haoyan@163.com

效率。通过设计通用、标准的审计数据格式，将格式相异、类型千差万别的审计数据统一起来以标准化的格式进行集中存储和管理。

系统审计模块则运用各种数据挖掘技术如关联分析、异常点分析等对海量日志数据进行分析，从全局的角度发现网络、系统及应用中存在的安全隐患和入侵行为，能够对安全侵害事件作出自动响应，提供审计自动报警功能。当审计系统检测到违反安全策略的可疑行为发生并达到预定的域值时，发出报警信息并自动断开该用户的连接，以便管理员进行及时阻断和改进。

报表模板分类分层地把日志分析结果可视化地上报给管理员，使管理员能够从不同角度对整个系统的安全进行全面监视。

用户不必考虑具体的操作系统、硬件环境和编程语言，可以方便地通过网络查找适合自己的审计系统并生成相应的SOAP 请求调用，对系统进行安全审计。

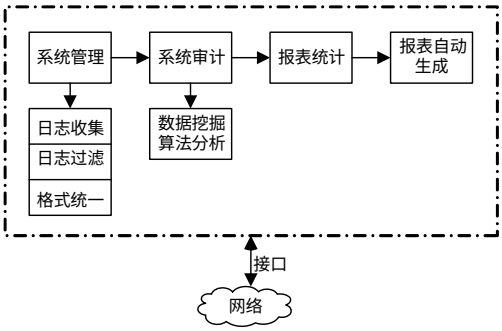

图 2　系统功能构架

1.3　信息安全审计系统的信息描述

计算机和网络中采集的日志一般都是纯文本或其他非通用的格式，不符合Web Services松散耦合的格式要求，因此需要对数据库中收集的原始日志进行预处理，定义一个全面的基于XML的标准日志格式。XML格式本身是一个树状结构，首先通过MSXML的DOM模型导入内存生成一个IXMLDOMDocument实体，其中包含XML声明、唯一的根节点、各级元素、元素属性和文本几部分，每一部分在MSXML中都有一个接口与之相对应[1]。程序中把IXMLDOMDocument接口对应的实体看作一个大容器，其中又包含很多小容器分别存放各个元素的内容，下面是基于XML定义的简单日志格式（以Linux系统安全启动为例）：

```
<Log>
<Syslog>
<Date>Nov 29 16:48:43</Date>
  <Hostname>redhat</Hostname>
  <Process>root[1759]</Process>
  <Description>ROOT LOGIN ON tty2</Description>
 </Syslog>
</Log>
```

其中，<Date>标签中存储日志的时间和日期；<Hostname>存储产生日志的主机名称；<Process>是日志的进程名称，就是记录系统登录所发生的行为；<Description>标签中的内容是对<Process>中记录进程的具体描述。通过把收集到的各种格式的日志统一转换为如上面的标准 XML 格式存储，利用XML 格式的分层结构将每一条日志内容分门别类地显示，方便了管理和进一步的分析研究。

2　基于 Web services 的信息安全综合审计系统接口

2.1　Web services 技术

Web services 是面向对象的技术架构，它的底层基础是技术已经非常成熟的 IP、HTTP、SMTP 等，其他的支持技术有：简单对象访问协议 (Simple Object Access Protocol，SOAP)、Web Service 描述语言 (Web Services Description Language，WSDL)、通用描述发现和集成协议（Universal Description，Discovery，and Integration，UDDI）。SOAP 是基于 XML 跨平台的轻量级通信协议，能够穿越防火墙在互联网的应用程序之间收发信息并已经发展成为 W3C 的标准。WSDL 是用 XML 语言写的用于描述定位 Web services 的文档。UDDI 规定了 Web services 如何公开自己以及如何在网络上相互发现和集成，它通过 SOAP 协议进行通信并为查找和访问服务定义了注册中心和相关的协议。

目前Web服务主要构架平台有以微软为主阵营的.Net平台和以Sun、IBM、BEA、ORACLE等为主要支持者的J2EE平台[2]。为了充分发挥Web services的可重用性、互操作性和松散耦合性，用XML语言撰写一个WSDL文件对其进行描述，将此WSDL发布到UDDI上进行注册。用户就可以根据需要到UDDI上搜索需要的WSDL，生成一个相应的SOAP消息嵌入在一个HTTPPOST中提出请求。Web services从产生到应用的流程如图 3 所示。

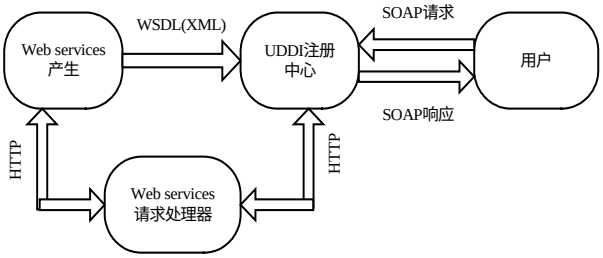

图 3　Web services 从产生到应用的流程

2.2　日志格式统一表示

将审计系统的日志信息用 XML 格式存储并以表格的形式呈现，用户就可以非常直观的查看日志信息的各项内容，也可以根据关键词查询感兴趣的特定日志，极大地提高审计效率。统一后的日志表格如图 4 所示。

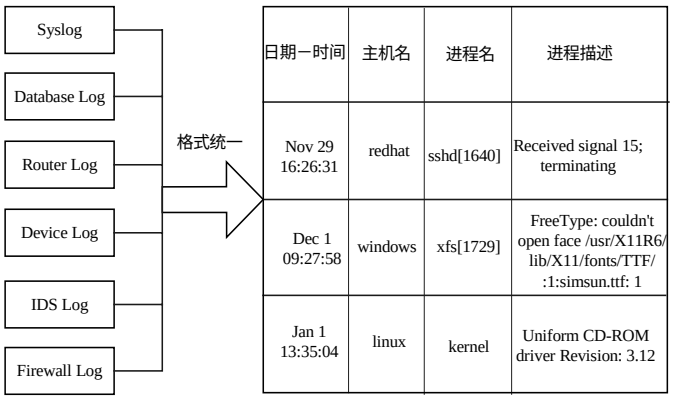

图 4　统一的日志表格

2.3　基于 Web services 的审计系统接口

通过 Microsoft.NET 平台建立一个 Web services 接口，就可以利用标准化 XML 消息传递在网络上访问和操作本文的审计与取证系统控制台。以下是实现过程：

(1)利用WSDL 1.2 版本对系统进行Web services接口描

述，其中，<message>标签定义程序的数据元素从一点到另一点的调用，可以包含多个<part>。<portType>是WSDL中最重要的部分[3]，描述程序进行的操作和涉及的信息。

```
<wsdl:definitions name=”AuditSystem”?
<message name=”getLogRequest”>
          <part name="term” type=”xs:_bstr_t”/>
</message>
<message name=”getLogResponse”>
  <part name=”DOM” type=”xs:_bstr_t”/>
</message>
<portType name=”LogAudit”>
  <operation name=”getLog”>
     <input message=”getLogRequest”/>
     <output message=”getLogResponse”/>
</operation>
</portType>
</wsdl:definitions>
```

(2)系统在 UDDI 进行注册。UDDI 定义了由 WSDL 描述的 Web services 接口的存储路径并且已经嵌入在 Microsoft.NET 平台中。将 Web services 的地址注册到 UDDI，通过访问此地址找到 WSDL 描述的信息安全审计系统的 Web services 接口。先在代码的头中加入 UDDI 的相关命名空间：

```
Microsoft.Uddi;
Microsoft.Uddi.Binding;
Microsoft.Uddi.Service;
Microsoft.Uddi.ServiceType;
```

再将 WSDL 文档创建成 TModel 结构并产生一个 TModelKey 用于和 Web Services 绑定，在 http://uddi.microsoft. com/进行注册。

(3)通过 SOAP 与 HTTP 协议绑定对系统的请求和响应。

```
<?xml version="1.0" encoding=”GB2312”?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body>
<m:GetSyslog>
<m:Item>//LogFiles//Syslog</m:Item>
</m:GetSyslog>
</soap:Body>
</soap:Envelope>
```

下面是 SOAP 与 HTTP 协议绑定：

1)消息请求

```
POST /item HTTP/1.1
Host: 192.168.10.239
Content-Type: text/xml; charset=GB2312
Content-Length:500
```

2)消息响应

```
500 OK
Content-Type: text/xml; charset=GB2312
Content-Length: 500
```

3 总结

本文设计的信息安全审计系统能实现对主机的操作系统信息获取、网络设备信息和日志管理。对网络中的拨号设备、主机非法接入、主机上共享某些特定文件实现安全监控。能够获取各种主流操作系统主机的现场证据，并对其进行安全的保护和传输。通过 Web services 软件接口可以实现跨多个操作系统、编程语言和硬件平台的松耦合问题，解决数据和应用程序集成的问题，将技术性的功能转换成面向服务的问题，具有广泛的应用空间。

参考文献

1 XML-binary Optimized Packaging[Z]. 2005-01. http://www.w3.org/TR/xop10.

2 胡海璐, 周涛武. Visual C++.NET 高级编程技术与范例[M]. 北京: 电子工业出版社, 2002.

3 WSDL Ports[Z]. 2006. http://www.w3schools.com/wsdl/wsdl_ports.asp.

4 Thomas J P, Thomas M, Ghinea G. Modeling of Web Services Flow[C]//Proceedings of the IEEE International Conference on E-Commerce. 2003: 391-398.

(上接第 144 页)

图 7 是医学图像做了边缘锐化处理后提取出的水印图像，可以看出原图像边缘处对应的水印部分不能正确检测出来，因此可以判断边缘做了处理。

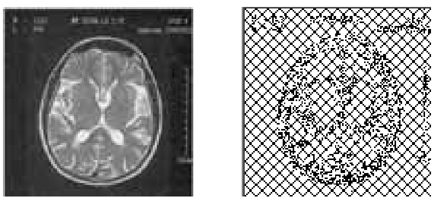

图 7 图像边缘锐化后的图像和提取出的脆弱水印

从实验结果 5，可以看出，脆弱性水印具有准确、直观的定位能力，并且可以通过分析水印的变化来推断图像受到的攻击类型。

参考文献

1 Voyatzis G, Pitas I. Chaotic Watermarks for Embedding in the Spatial Digital Image Domain[C]//Proc. of International Conference on Image Processing, 1998: 432-436.

2 Xuan G, Zhu J. Distortionless Data Hiding Based on Integer Wavelet Transform[J]. IEEE Electronics Letters, 2002, 38(25): 1646-1648.

3 华先胜, 石青云. 多类水印的同时嵌入[J]. 北京大学学报, 2001, 37(3): 305-313.

4 崔雪英, 杨厚俊. 基于提升小波多子带的复合型盲水印技术[C]//中国计算机大会论文集. 北京: 清华大学出版社, 2005.

5 孙圣和, 陆哲明, 牛夏牧. 数字水印技术及应用[M]. 北京: 科学出版社, 2004.

6 Kundur D, Hatzinakos D. Digital Watermarking for Telltale Tamper-proofing and Authentication[J]. Proceedings of the IEEE, 1999, 87(7): 1167-1180.