

# 基于 Spring 的网站文件安全监测系统设计

丁振凡, 吴根斌

(华东交通大学 信息工程学院, 江西 南昌 330013)

**摘要:**由于黑客攻击,网站文件被改动的情况常有发生。因此,对站点的文件进行安全监测和处理也就有较大的实际意义。文中介绍的站点安全检测系统是基于 J2EE 服务器环境。采用 Spring 技术编程实现。安全检测功能是通过当前文件信息与初始信息的比对进行判定,对攻击行为给予报警和处理。系统实现了 Web 站点文件状况的远程监控。另外,系统还利用 Spring 的任务调度功能进行任务安排,实现每天定时检测,可在站点非繁忙时间自动实现检测和处理。在实际使用中,对网站的文件安全起到了很好的保护效果。

**关键词:**Web 站点安全;文件攻击检测;任务定时计划;Spring;AOP

中图分类号:TP393

文献标识码:A

文章编号:1673-629X(2012)12-0179-04

## Design of Monitoring System Based on Spring for Web File Security

DING Zhen-fan, WU Gen-bin

(School of Information Engineering, East China Jiaotong University, Nanchang 330013, China)

**Abstract:** Because of hackers attack, the status of the website files that are changed quite often happens. Therefore, safety monitoring and dealing with the website files have greater practical significance. The security monitoring system of the website is based on J2EE server. It uses Spring technology to realize programming. Security monitoring function determines the file status and gives an alarm and processing by comparing the current file information with initial information. The system realizes remote monitoring of the Web site file status. In addition, it also uses the task scheduling function of Spring to realize task arrangement and regularly check every day. It can realize automatic detection and treatment in free time to the site. Spring plays a good effect on file security of website in actual use.

**Key words:** security of Web site; file attack monitoring; task scheduled plan; Spring; AOP

## 0 引言

Web 应用已成为 Internet 的主流应用。长期以来,Web 站点上的文件安全事件也经常发生。黑客可通过系统的安全漏洞将破坏性程序上传到服务器上,或者对服务器上的已有程序进行修改,添加病毒脚本,从而影响网站的正常运作<sup>[1]</sup>。

目前,国内外对文件安全检测的研究成果不多。郑晓红等从文件安全保护角度总结了通过隐藏、加密、删除、访问控制四种方法<sup>[2]</sup>,研究了 Unix 环境下如何对文件进行安全防护。Tripwire 是 Unix 环境下文件系统完整性检查的软件工具,此软件采用的技术核心就是对每个要监控的文件产生一个数字签名,并保留下来。它是一个文件完整性监测工具,用来监测对文件

系统进行未加认证的修改<sup>[3]</sup>。它通过取得文件的数字“快照”或“指印”,并保存在数据库中。然后,把文件当前的快照和原来的快照进行比较。已被添加、删除或更改内容的文件由 Tripwire 进行标记,使系统管理员能够采取相应的措施,从而既监测了入侵,也为入侵后系统的恢复节省了时间<sup>[4]</sup>。

文中介绍的文件安全检测监控系统是采用 Web 技术实现安全检测,监测对象是 Web 服务器指定路径下的文件,系统采用 Spring 框架编程实现。

## 1 系统设计与实现

本系统将某目录下的文件安全检查的功能封装在一个 Java Bean 中,它根据服务器文件的原始状况与当前状况的比对来检测服务器上文件的变化,包括文件的修改、删除和新增。

文件安全检测系统设计为 Web 使用方式,管理员在 Web 界面中可远程监控服务器上文件变化,并做出相应处理。考虑到管理员不一定经常去访问管理系

收稿日期:2012-03-28;修回日期:2012-06-29

基金项目:江西省自然科学基金项目(20114bab201026)

作者简介:丁振凡(1965-),男,江西丰城人,教授,硕士生导师,主要研究方向为语义 Web、分布式计算、计算机辅助教学;吴根斌(1987-),男,江西吉安人,硕士研究生,研究方向为语义 Web。

统。因此,还提供了另外一种检测报警方式。通过让服务器每天定时执行一个定时检测服务,调用其功能检查文件的变化,并将变化信息登记在一个 XML 文件中。当管理员登录系统时,系统自动查阅存放报警信息的 XML 文件,并通过弹出消息框提醒管理员。该软件系统的组件基本构成如图 1 所示。

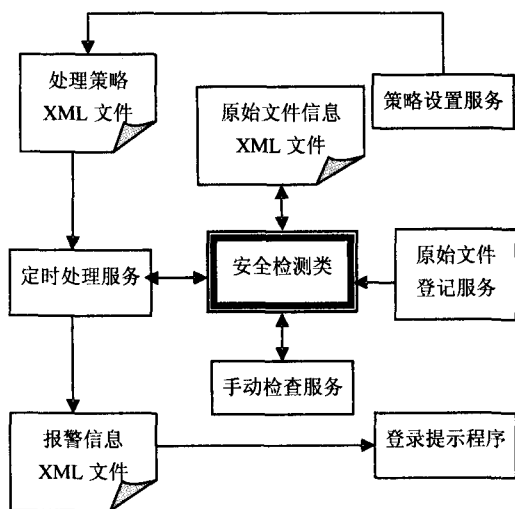


图 1 软件系统的基本构成

说明如下:

(1) 为了完成文件的变化检查,必须首先对被保护的文件夹下的所有文件进行登记。原始文件登记服务的设计采用 Spring MVC 控制器的 REST 风格的实现。登记内容包括每个文件名称和文件最后修改时间。

(2) 安全检测类是整个系统的核心,其中包括两个方法,一是初始登记,另一个是变化检测。原始文件登记服务、手动检查服务和定时处理服务均要利用该检测类提供的功能。

(3) 策略设置服务用于设置出现安全问题的处置方式。包括自动恢复处理和由管理员通过 Web 界面进行手动恢复处理,或重传破坏的文件等处理形式。

### 1.1 安全检测类的功能设计

安全检测类是整个系统的核心,其功能在其他安全处理模块中调用。文件安全检测类 FileDetect 的设计主要功能利用 JDK 的 File 类的功能实现。

(1) 初始登记 logToXML(): 将某目录下文件安全状态信息记录到 XML 文件中。每个文件含文件名和最后修改日期两个数据项。

(2) 检测文件变化 findDetect(): 返回有变化的文件信息集合。变化信息保存在一个 List 列表中。每个列表项包括文件名,变化原因标识。变化原因可分为: 新增文件、被删除、有修改等情形。

(3) 将某文件内容写入数据库。当文件被修改后,要恢复原始文件。因此,需要考虑对文件内容进行

备份,以便恢复处理使用。系统采用 Spring 的 JDBC 模板实现对数据库的操作访问。

(4) 从数据库恢复某文件内容。将受破坏文件用数据库保存的内容替换。

实现安全检测功能的方法 findDetect() 的具体实现算法如下:

步骤 1. 访问原始登记 XML 文档,所有文件表示为 NodeList 对象;

步骤 2. 获取检测目录的文件列表,放入列表对象 list1 中;

步骤 3. 循环检查 NodeList 中每个文件项;

步骤 3.1 根据文件在列表 list1 中是否存在,判断文件是否删除;

步骤 3.2 根据文件修改时间是否与当前该文件修改时间一致,判断文件是否修改;

步骤 4. 将原始 XML 中记录所有文件放入列表对象 list2 中;

步骤 5. 根据检测目录下文件在列表 list2 中是否存在决定是否是否为新增文件;

步骤 6. 将所有检测结果放入到存放问题的列表对象中返回。

### 1.2 文件的修复

目前,文件的修复可考虑两种方法实现。一是通过远程备份的形式,当管理员得到某文件被破坏的信息时,重新将文件上传到服务器进行覆盖。新增的文件通常考虑由管理远程进行删除,具体借助 JDK 的文件对象的 delete 方法实现<sup>[5]</sup>。另一种是利用数据库保存的文件的备份内容进行恢复<sup>[6]</sup>。自动恢复处理是采用此方式进行恢复,系统利用 Mysql 数据库保存文件信息,每个文件占用数据库的一条记录。定时检查服务在发现问题时根据配置可自动执行恢复,也可由管理员通过远程的 URL 请求触发执行。

远程备份是在管理员自己的机器上保留程序备份。恢复时重新将文件上传即可<sup>[7]</sup>。文件上传采用 Spring MVC 模式编程实现<sup>[8]</sup>,Spring MVC 支持 REST 风格的路径表示,可方便通过 @RequestParam、@PathVariable 等注解符变量获取表单请求参数和 REST 的路径参数,还可在方法参数中注入 HttpServletRequest 等类型的标准对象,从而方便了请求数据获取和功能调用。

以下为实现文件上传功能的几个编程要点:

①在 Web 应用的 lib 目录下引入 Apache Commons FileUpload 和 Apache Commons IO 两个 JAR 文件;

②使用 CommonsMultipartResolver 解析器处理上传。在 MVC 控制器的配置文件中包括如下 Bean,其中,上传文件大小限制可根据需要设定。

```
<bean id="multipartResolver" class=
"org.springframework.web.multipart.commons.CommonsMul-
tipartResolver">
<property name="maxUploadSize" value="100000" />
</bean>
```

③设置提交表单的 enctype 属性为“multipart/form-data”,在表单中通过类型为“file”的输入元素选择上传文件,表单的提交方法为“POST”。在处理上传请求的控制器中通过 MultipartFile 类型的参数对象获取上传文件数据和文件名。

### 1.3 自动定时检测

为了减轻管理员的检测负担,系统提供自动定时检测功能,并将出现的问题通过自动短信或自动发送邮件提醒管理员。自动定时检测处理采用 Spring 的任务调度功能实现。系统中设置任务执行时间为每天晚上的 11 点,定时调用“文件安全检测”的 Java Bean 的功能,对指定文件夹下的所有文件进行检测处理,并根据配置策略进行处理。Spring 提供了多种实现任务定时执行的方法。使用最简单的是利用 Spring3 提供的基于注解的定时调度功能<sup>[9]</sup>。具体实现要点如下:

(1) 加入必要 JAR 包。Spring3 的任务调度实际上也是 AOP 的应用<sup>[10]</sup>,除了加入 Spring 框架中的 AOP 和 aspects 包外,还要加上 aopalliance.jar。

(2) 在配置文件中要增加如下配置信息:

```
<context:component-scan base-package="ecjtu.jx.task" />
<!-- 上面设置需要扫描部件的包路径-->
<context:annotation-config /> <!-- 开启注解-->
<task:annotation-driven/> <!-- 定时器开关-->
```

(3) 通过 @Scheduled 注解设置方法定时调度执行。

```
import org.springframework.scheduling.annotation.Scheduled;
import org.springframework.stereotype.Service;
@Service
public class TaskCheck {
    @Scheduled(cron="0 0 23 * * ?")
    public void check() {
        FileDetect x = new FileDetect("d:\\mydir"); //指定检查的目录
        x.findDetect(); // 调用安全检查程序进行检查
        .....//根据策略设计对出现的问题进行处理
    }
}
```

其中,cron 表达式“0 0 23 \* \* ?”代表每天晚上十一点触发。

Spring 将根据注解配置的定时设置内容,自动创建 TaskScheduler 实例并调用指定的方法。

### 1.4 Web 手动检测管理

管理员可在任何时刻通过 Web 访问服务器上的

手动检测管理程序,由控制器根据用户的 URL 请求调用安全检查 Bean 的功能来实现文件安全检测。在浏览器页面中显示检测结果。管理员可以通过页面上的超链对文件进行变动处理。

管理员操作部分的 Web 安全管理主要功能有:

(1) 原始文件信息登记。每次应用该功能就意味着更改系统的初始设置。它将服务器文件的当前状况进行记录,以便以后作为检查变化的依据。

(2) 检查文件变动。将显示文件变动信息,并提供超链对变动进行处理,如:删除新增文件、恢复被改文件、恢复被删文件等。

(3) 自动处理策略设置。对每天自动检测后发现的问题的处理策略进行设置。

(4) 系统文件备份。将文件原始信息保存到数据库,以便恢复使用。

#### 1.4.1 启动检测

在 Web 界面中通过超链访问实现对服务器启动检测功能的调用。

以下为 Spring MVC 控制器中调用安全检测功能的访问 Mapping 设置, Spring MVC 支持 REST 风格 URL 定义。

```
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

@Controller
public class SecurityControl {
    // 初始状况登记
    @RequestMapping(value="/writerootfile", method=RequestMethod.GET)
    public String initlog(HttpServletRequest request) {
        FileDetect x=new FileDetect("d:\\cai");
        int k=x.logToXML();
        request.setAttribute("amount",k);
        return "havelog"; //由视图文件 havelog.jsp 显示登记文件数量
    }

    //文件检测
    @RequestMapping(value="/checkchange", method=RequestMethod.GET)
    public String check(HttpServletRequest request) {
        FileDetect x=new FileDetect("d:\\cai"); //安全保护的目录路径
        List<MyResult> r=x.findDetect(); //调检测功能
        request.setAttribute("result",r); //检测结果作为模型数据
```

