(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 102073579 A
(43)申请公布日 2011.05.25

(21)申请号 201110025251.1

(22)申请日 2011.01.24

(71)申请人 复旦大学
地址 200433 上海市杨浦区邯郸路 220 号

(72)发明人 杨珉 毛迪林 李景涛 朱东来
张涛 谢鹏 臧斌宇

(74)专利代理机构 上海正旦专利代理有限公司 31200
代理人 陆飞 盛志范

(51)Int.Cl.
G06F 11/34(2006.01)
G06F 21/00(2006.01)
H04L 29/06(2006.01)

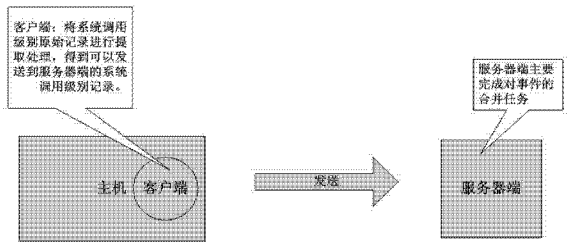
权利要求书 2 页 说明书 7 页 附图 2 页

(54)发明名称

Linux 文件系统审计事件合并和优化的方法

(57)摘要

本发明属于计算机系统安全管理技术领域，具体涉及一种 Linux 文件系统审计事件合并和优化的方法。本发明方法是在客户端和服务器端同时进行合并优化，客户端对目录下被监控文件通过审计规则进行判断，符合则形成最原始的审计记录，并通过提取处理后发送到服务器端；基于自定义过滤规则，不发送无审计意义的记录；服务器端对一新的审计记录与对应的综合审计记录根据匹配规则进行判断。若匹配，则合并进此综合审计记录；若不匹配，则新建一条综合审计记录，把这条审计记录合并进新的综合审计记录。从而清晰地还原对审计监控目标进行文件操作的线索。



CN 102073579 A

1. 一种对 Linux 文件审计系统调用级别的日志记录进行合并和优化的方法，其特征在于在客户端和服务器端同时进行合并优化，将合并优化的工作分别分担在服务器和客户端，具体步骤为：

（1）根据用户自定义的审计规则，形成配置文件，客户端对目录下被监控文件通过审计规则进行判断，符合则形成最原始的审计记录，并通过提取处理后发送到服务器端；

（2）事件合并：对一新的审计记录与对应的综合审计记录根据匹配规则进行判断，若匹配，则合并进此综合审计记录；若不匹配，则新建一条综合审计记录，把这条审计记录合并进新的综合审计记录；

（3）合并优化：基于自定义过滤规则，不发送无审计意义的记录。

2. 根据权利要求 1 所述的对 Linux 文件审计系统调用级别的日志记录进行合并和优化的方法，其特征在于所述事件合并的具体过程为：

每一条审计记录在数据库中都有一个自动生成的序列号 auditid 来唯一的标识它；

每一条审计记录都有一个 auditGid 的整数字段，这个字段中保存的是对应的综合审计记录的 auditid，综合审计记录的 auditgid 为 -1；每一条审计记录都有一个 integratedOperation 的整数字段，该字段用于表示合并后的事件操作，对于一般的审计记录，该字段为 -1；通过这两个字段将一般的审计记录和综合审计记录区分开，同时又可以在需要查询时将他们联系起来；

每一条审计记录都有一个 syscall 的字符串字段来保存产生该记录的系统调用；

一般审计记录的操作 operation 用一个整数字段表示，从 0 到 8 依次为：读出，写入，执行，改变属性，加载设备，卸载设备，打印，移动介质读出，移动介质写入；

当一条审计记录到达时，去数据库中查找是否有对应综合审计记录，综合审计记录与该记录的 PID，执行程序，目标路径相同，且两条记录的相差时间在一定范围内；如果有这样的综合审计记录，那么就不用新建一条综合审计记录，只需将审计记录合并进入综合审计记录；如果没有则新建一条综合审计记录；将到达的审计记录的 auditgid 设为综合审计记录的 auditid，将到达的审计记录的 syscall 字段与综合审计记录的 syscall 字段进行合并，将到达的审计记录的 operation 合并进综合审计记录的 intergratedOperation 字段；因为总共有九种操作，所以当审计记录的 operation 为 n 时，$n \in [0,8]$，将综合审计记录的 intergratedOperation 字段分别和 2^n 做“或”操作，即可将审计记录的 operation 合并进入综合审计记录，在要取出该事件进行了哪些操作时，将综合审计记录的 integratedOperation 字段分别与 2^n 进行“与”操作，如果“与”操作后大于零，则说明采取过对应的操作；最后插入到达的审计记录。

3. 根据权利要求 1 所述的对 Linux 文件审计系统调用级别的日志记录进行合并和优化的方法，其特征在于所述合并优化方法，具体如下：

（1）、在客户端

对于一些冗余和多余的信息提供相关的配置给用户进行选择，如用户可以配置在某个目录下的匹配规则，在该目录下被监控的文件只有符合匹配规则的才会发送到服务器端；用户通过配置过滤规则来对一些不必要的文件进行过滤；

(2)、在服务器端

主要完成对于事件的合并任务，其实现过程如下：

对同一个监控目标的同一个操作所产生的一系列审计记录，将新增加一条记录，用这条记录来实现这些记录的合并任务，称之为综合审计记录；当用户的一条审计记录到达时，首先根据记录中的 PID，执行程序和时间来判断是否数据库中已经有该事件的综合审计记录存在，如果已经有综合审计记录存在，则将这条审计记录的信息合并进综合审计记录，同时保存该条信息作为以后的功能扩展；如果没有对应的综合审计记录存在，那么则新建一条综合审计记录。

4. 一种基于系统调用级别的日志记录对 Linux 文件审计的系统，包括一个审计事件主机，一个终端服务器，数据存储装置，网卡，以及数据输入装置和输出装置，其特征在于还包括：

一个审计规则单元，用户自定义一个审计规则，形成相应的配置文件；

一个匹配规则单元，通过此规则对审计记录进行规则匹配；

一个过滤规则单元，通过一个自定义的过滤规则来对无审计意义的记录进行过滤。

Linux 文件系统审计事件合并和优化的方法

技术领域

[0001] 本发明属于计算机系统安全管理技术领域，具体涉及一种 Linux 文件系统审计事件合并和优化的方法。

背景技术

[0002] 安全审计是计算机系统安全管理的一个重要的组成部分。安全审计是记录用户的访问过程和各种行为形成审计数据的过程。对审计数据的分析可以发现系统中的安全问题、识别系统事故责任者、跟踪某些用户和站点，为及时采取相应处理措施提供依据。

[0003] 设置审计的目的，是在计算机系统中监视、记录和控制用户活动以便检测和判定对系统的恶意攻击和错误操作，将审计日志作为事后分析和追查的证据。合理地应用审计技术，可以有效地对影响系统安全的访问和访问企图起到威慑作用，为系统提供进一步的安全可靠性。

[0004] 安全审计跟踪的功能是：帮助安全人员审计系统的可靠性和安全性；对妨碍系统运行的明显企图及时报告给安全控制台，及时采取措施。一般要在网络系统中建立安全保密检测控制中心，负责对系统安全的监测、控制、处理和审计。所有的安全保密服务功能、网络中的所有层次都与审计跟踪系统有关。

[0005] 审计机制在 Linux 系统开机后会自动开启，处于审计状态，记录应该记录的内容。审计机制一般专门由审计管理员进行管理，审计管理员可以随时关闭审计功能。一旦审计功能被关闭，任何用户的动作都将不再处于审计系统的监视下，也不再记录任何审计信息，因此一般情况下不允许关闭审计功能。

[0006] 审计机制将审计结果的原始记录存放在审计日志，每次审计进程开启后，都会按照已设定好的路径和命名规则产生一个新的日志文件。对用户有用的审计记录一般包括如下信息：事件发生的日期和时间、执行事件的用户、发生事件的类型、事件的状态（成功或失败）等。

[0007] 本发明涉及的一些概念介绍如下：

定义 1：日志记录日志记录是通过审计配置文件和审计对象而形成的系统调用级别原始记录，包含所要的发送服务器的信息。

[0008] 定义 2：审计事件记录审计事件记录是日志记录经过提取处理出来以后生成的系统调用（syscall）级别的将要发送到服务器的记录，从而生成相应的字段，也就是处理后系统调用级别记录

定义 3：服务器端数据库表中的综合审计记录服务器端表中系统调用级别原始记录合并以后事件级别存在表中记录。即为服务器端数据库中存放事件级别的审计记录。

[0009] 对于 Linux 内核支持的审计模块来说，当配置好审计规则之后，审计守护进程会根据配置的规则对相应的文件或目录等进行监控，但这些监控都是基于系统调用级别，即对同一个文件或目录进行了不同的系统调用时就会产生不同的日志文件记录，且由于某些文件操作会产生很多中间文件，如 .swp,.swx 文件等等，在日志文件记录中对审计有意义

的一个文件操作可能对应若干条日志记录。因此日志记录非常繁琐和复杂，无法提供友好的界面，来清晰地还原对审计监控目标进行文件操作的线索。审计事件合并和优化的思想主要针对解决这样的问题而提出。

发明内容

[0010] 本发明的目的在于提供一种 Linux 文件系统审计事件合并优化方法，以便为日志记录提供友好的界面，清晰地还原对审计监控目标进行文件操作的线索。

[0011] 本发明提出的 Linux 文件系统审计事件合并优化方法，涉及到相关字段的合并，这些相关字段包括：

用户名 (user)、节点 (node)、审计序列号 (serial)、监测时间 (dtime)、记录生成时间 (ctime)、目标路径 (target)、操作 (operation)、结果 (result)、严重程度 (severity)、错误信息 (err)、系统调用 (syscall)、执行程序 (exe)、详情 (detial)。

[0012] 1、用户名 (user)

用户名信息为审计记录中相应操作的操作者，在审计原始记录中包含了操作者的 id 信息，优先选择 auid 来获得 user 名，如果 auid 无效，则选择 uid 来获取 user 名。

[0013] 2、节点 (node)

节点信息即为被监控的机器的 IP 地址信息，通过获取机器的 IP 地址即可得到该节点信息。

[0014] 3、审计序列号 (serial)

在产生的审计原始记录中包含有该记录的审计序列号来唯一标志同一台机器上的各条审计记录，通过函数接口可以得到该字段。

[0015] 4、监测时间 (dtime)

在产生的审计原始记录中包含有监控时的时间戳信息，可以提取出该事件信息，并通过转化成服务器端接收的标准格式即可得到该字段信息。

[0016] 5、记录生成时间 (ctime)

记录生成时间是将审计原始记录合并成为将要发送到服务器端的审计记录时的时间，通过在合并原始记录时获取时间戳并将其转化成服务器端接收的标准格式即可得到该字段信息。

[0017] 6、目标路径 (target)

目标路径由目标绝对路径或者是由目录加相对路径整合而成。

[0018] 7、操作 (operation)

审计原始记录中包含了一系列系统调用的参数，系统调用的操作不能简单的通过直接获取系统调用操作提取，因为同样的操作可能是出于不同的需要，例如打开文件和写入文件都需要先打开文件，在提取操作信息时要试图得到真正的用户操作，因此需要通过一些列的系统调用操作的参数分析最终整合得到最合理的操作字段。

[0019] 8、结果 (result)

审计原始记录中包含了每个系统调用操作的结果信息，将这些结果信息进行提取和综合即可得到该字段来显示系统调用是否成功完成。

[0020] 9、严重程度 (severity)

在配置所要监控的文件目录信息时，会要求用户配置相应的严重等级，在产生的审计原始记录的 key 中即会包含 severity 的字符串，提取出对应的字符串即可得到该字段。

[0021] 10、错误信息 (err)

当系统调用失败时，审计原始记录中会记录下调用失败的错误原因，可以提取到该原因作为该字段，让监控者知悉调用失败的错误。

[0022] 11、系统调用 (syscall)

因为审计原始记录是按系统调用级别来记录的，在审计原始记录中包含了系统调用的标识，通过解析这些标识可以得到系统调用的名称字段。

[0023] 12、执行程序 (exe)

在审计原始记录中包含了执行系统调用命令的执行程序信息，可以提取出该信息作为该字段的内容。

[0024] 13、详情 (detial)

详情字段根据具体的情况有所不同。对于不同类型的审计记录，可以根据实际的需要将附加信息写入详情字段，方便后续的操作。

[0025] 本发明提出的对 Linux 文件审计系统调用级别的日志记录进行合并和优化的方法，是在客户端和服务器端同时进行合并优化，将合并优化的工作分别分担在服务器和客户端，具体步骤为 ：

(1) 根据用户自定义的审计规则，形成配置文件，客户端对目录下被监控文件通过审计规则进行判断，符合则形成最原始的审计记录，并通过提取处理后发送到服务器端 ；

(2) 事件合并 ：对一新的审计记录与对应的综合审计记录根据匹配规则进行判断，若匹配，则合并进此综合审计记录 ；若不匹配，则新建一条综合审计记录，把这条审计记录合并进新的综合审计记录 ；

(3) 合并优化 ：基于自定义过滤规则，不发送无审计意义的记录。

[0026] 本发明的合并事件方法，具体如下 ：

每一条审计记录在数据库中都有一个自动生成的序列号 auditid 来唯一的标识它。

[0027] 每一条审计记录都有一个 auditGid 的整数字段，这个字段中保存的是对应的综合审计记录的 auditid，综合审计记录的 auditgid 为 -1 ；每一条审计记录都有一个 integratedOperation 的整数字段 , 该字段用于表示合并后的事件操作，对于一般的审计记录，该字段为 -1。通过这两个字段可以将一般的审计记录和综合审计记录区分开，同时又可以在需要查询时将他们联系起来。

[0028] 每一条审计记录都有一个 syscall 的字符串字段来保存产生该记录的系统调用。

[0029] 一般审计记录的操作 operation 用一个整数字段表示，从 0 到 8 依次为 ：读出，写入，执行，改变属性，加载设备，卸载设备，打印，移动介质读出，移动介质写入。

[0030] 当一条审计记录到达时，将会去数据库中查找是否有对应综合审计记录，综合审计记录与该记录的 PID，执行程序，目标路径相同，且两条记录的相差时间在一定范围内，例如在前后一秒钟之内，如果有这样的综合审计记录，那么就不用新建一条综合审计记录，只需将审计记录合并进入综合审计记录。如果没有则新建一条综合审计记录。将到达的审计记录的 auditgid 设为综合审计记录的 auditid，将到达的审计记录的 syscall 字段与综合审计记录的 syscall 字段进行合并，将到达的审计记录的 operation 合并进

综合审计记录的 intergratedOperation 字段，因为总共有九种操作，所以当审计记录的 operation 为 n 时（n ∈ [0,8]），将综合审计记录的 intergratedOperation 字段分别和 2^n 做“或”操作（int(intergratedOperation)=intergratedOperation|| 2^n），即可将审计记录的 operation 合并进入综合审计记录，在要取出该事件进行了哪些操作时，将综合审计记录的 integratedOperation 字段分别与 2^n 进行“与”操作，如果“与”操作后大于零，则说明采取过对应的操作。最后插入到达的审计记录。

[0031] 本发明的进一步优化方法，具体如下：

1、在客户端。

[0032] 在客户端方面，对于一些冗余和多余的信息提供了相关的配置给用户进行选择，如用户可以配置在某个目录下的匹配规则，在该目录下被监控的文件只有符合匹配规则的才会发送到服务器端；用户可以配置过滤规则来对一些不必要的文件进行过滤，例如将后缀名为 .log 的文件过滤而不发送，只需要配置一条规则即可。

[0033] 2、在服务器端。

[0034] 在服务器端主要完成对于事件的合并任务。其实现过程如下：

对同一个监控目标的同一个操作所产生的一系列审计记录，我们将新增加一条记录，这条记录来实现这些记录的合并任务，可以称之为综合审计记录。当用户的一条审计记录到达时，首先根据记录中的 PID，执行程序和时间来判断是否数据库中已经有该事件的综合审计记录存在，如果已经有综合审计记录存在，则将这条审计记录的信息合并进综合审计记录，同时保存该条信息作为以后的功能扩展。如果没有对应的综合审计记录存在，那么则新建一条综合审计记录。

[0035] 一种基于系统调用级别的日志记录对 Linux 文件审计的系统，该系统包括若干被审计主机，一个审计事件服务器，数据存储装置，网卡，以及数据输入装置和输出装置。每个主机包含一个客户端，客户端主将系统调用级别原始记录进行提取处理，得到可以发送到审计服务器端的系统调用级别记录。服务器端完成对事件的合并任务。还包括：

一个审计规则单元，用户自定义一个审计规则，形成相应的配置文件；

一个匹配规则单元，通过此规则对审计记录进行规则匹配；

一个过滤规则单元，通过一个自定义的过滤规则来对无审计意义的记录进行过滤。

附图说明

[0036] 图 1 主机、客户端和服务器端的作用图示。

[0037] 图 2 为合并事件流程图。

[0038] 图 3 为总流程图。

具体实施方式

[0039] 以监控用户对文件 /etc/passwd 的操作为例：

1. 在配置好对该文件的审计规则后，审计系统将根据这些规则进行监控。例如用 vi 对 /etc/passwd 进行了写入操作后，将后产生 16 条系统调用级别原始记录，可以按照审计

序列号分为四大条记录，如下所示：

第一条：

type=SYSCALL msg=audit(1295245487.625:813): arch=40000003 syscall=38 success=yes exit=0 a0=907b238 a1=90881e0 a2=907b238 a3=90881e0 items=4 ppid=18994 pid=19595 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts3 ses=2 comm="vi" exe="/bin/vi" subj=kernel key="audit-file-normal"

type=CWD msg=audit(1295245487.625:813): cwd="/mnt/hgfs/E/WangLuo/SourceCode/audit-1.7.18/audisp/plugins/hbaudit"

type=PATH msg=audit(1295245487.625:813): item=0 name="/etc/" inode=65409 dev=fd:00 mode=040755 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

type=PATH msg=audit(1295245487.625:813): item=1 name="/etc/" inode=65409 dev=fd:00 mode=040755 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

type=PATH msg=audit(1295245487.625:813): item=2 name="/etc/passwd" inode=205621 dev=fd:00 mode=0100644 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

type=PATH msg=audit(1295245487.625:813): item=3 name="/etc/passwd~" inode=205621 dev=fd:00 mode=0100644 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

第二条：

type=SYSCALL msg=audit(1295245487.625:815): arch=40000003 syscall=5 success=yes exit=3 a0=907b238 a1=8241 a2=1a4 a3=81a4 items=2 ppid=18994 pid=19595 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts3 ses=2 comm="vi" exe="/bin/vi" subj=kernel key="audit-file-normal"

type=CWD msg=audit(1295245487.625:815): cwd="/mnt/hgfs/E/WangLuo/SourceCode/audit-1.7.18/audisp/plugins/hbaudit"

type=PATH msg=audit(1295245487.625:815): item=0 name="/etc/" inode=65409 dev=fd:00 mode=040755 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

type=PATH msg=audit(1295245487.625:815): item=1 name="/etc/passwd" inode=205050 dev=fd:00 mode=0100644 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

第三条：

type=SYSCALL msg=audit(1295245487.625:816): arch=40000003 syscall=15 success=yes exit=0 a0=907b238 a1=81a4 a2=907a670 a3=1 items=1 ppid=18994 pid=19595 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts3 ses=2 comm="vi" exe="/bin/vi" subj=kernel key="audit-file-normal"

type=CWD msg=audit(1295245487.625:816): cwd="/mnt/hgfs/E/WangLuo/SourceCode/audit-1.7.18/audisp/plugins/hbaudit"

type=PATH msg=audit(1295245487.625:816): item=0 name="/etc/passwd" inode=205050 dev=fd:00 mode=0100644 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

第四条：

type=SYSCALL msg=audit(1295245487.630:817): arch=40000003 syscall=226

success=yes exit=0 a0=907b238 a1=2e749e3 a2=90859f8 a3=1c items=1 ppid=18994 pid=19595 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts3 ses=2 comm="vi" exe="/bin/vi" subj=kernel key="audit-file-normal"

type=CWD msg=audit(1295245487.630:817): cwd="/mnt/hgfs/E/WangLuo/SourceCode/audit-1.7.18/audisp/plugins/hbaudit"

type=PATH msg=audit(1295245487.630:817): item=0 name="/etc/passwd" inode=205050 dev=fd:00 mode=0100644 ouid=0 ogid=0 rdev=00:00 obj=unlabeled

2. 经过对记录的过滤合并,形成处理后的系统调用级别的记录,各条处理后系统调用级别记录分别如下 :

第一条处理后系统调用级别记录 :

root%72.6.67.8%813%Mon Jan 17 14:24:47 2011%Mon Jan 17 14:24:47 2011%/etc/passwd%1 %0 %0 %%rename%[19595][/bin/vi]vi%/etc/passwd rename to /etc/passwd~%

第二条处理后系统调用级别记录 :

root%120.14.67.8%815%Mon Jan 17 14:24:47 2011%Mon Jan 17 14:24:47 2011%/etc/passwd%1 %0 %0 %%open%[19595][/bin/vi]vi%flags=O_WRONLY|O_CREAT|O_TRUNC,mode=file,644%

第三条处理后系统调用级别记录 :

root%0.17.67.8%816%Mon Jan 17 14:24:47 2011%Mon Jan 17 14:24:47 2011%/etc/passwd%3 %0 %0 %%chmod%[19595][/bin/vi]vi%mode=file,644 %

第四条处理后系统调用级别记录 :

root%72.6.67.8%817%Mon Jan 17 14:24:47 2011%Mon Jan 17 14:24:47 2011%/etc/passwd%3 %0 %0 %%setxattr%[19595][/bin/vi]vi%%

记录的各个字段之间用"%"号分割,各个字段的含义依次为 :用户名 (user)、节点 (node)、审计序列号 (serial)、监测时间 (dtime)、记录生成时间 (ctime)、目标路径 (target)、操作 (operation)、结果 (result)、严重程度 (severity)、错误信息 (err)、系统调用 (syscall)、执行程序 (exe)、详情 (detial)。以第一条为例,进行操作的用户名为"root";被监控的机器节点 IP 为"72.6.67.8";审计代理产生的审计序列号为"813";监测时间为" Mon Jan 17 14:24:47 2011";记录生成时间为" Mon Jan 17 14:24:47 2011";监控的目标路径为" /etc/passwd";操作为"1",代表"写入"操作 ;结果为"0", 代表成功 ;严重程度为"0",代表 normal ;错误信息为空,代表没有发生错误 ;系统调用为"rename";执行程序为"[19595][/bin/vi]vi",代表执行该操作的进程号为"19595",执行的程序为"[/bin/vi]vi";详情为"/etc/passwd rename to /etc/passwd~"。

[0040] 3. 将处理后的系统调用级别的审计记录发送到服务器端,并形成服务器端数据库中存放的事件级别的审计记录。

[0041] 如果是第一条审计记录,则在数据库中查找没有发现该记录对应的综合审计记录,于是新建一条综合审计记录,该记录的 auditgid 为 -1,并将第一条记录的对应字段合并到综合审计记录中,这时综合审计记录的 intergratedOperation 字段的值为 $0 \| 2^1 = 2$;如果是第二条审计记录到达时,根据 PID, 执行程序,目标路径和时间范围查询到已经存在与

它对应的审计综合记录，于是直接将该记录插入数据库，它的 auditgid 字段的值为综合审计记录的 auditid，它的 intergratedOperation 字段的值为 -1， 同时将它的对应字段合并到综合审计记录中，此时综合审计记录的 intergratedOperation 字段的值为$2||2^1$ =2 ；同理，当第三条，第四条审计记录到达时采取更第二条类似的操作，intergratedOperation 字段的值依次为$2||2^3$ =10，$10||2^3$ =10，最终的综合记录 intergratedOperation 字段的值为 10，syscall 字段的值为 rename，open，chmod，setxattr.

4. 当审计记录存入数据库后，即可进行可视化的 web 呈现及提高高级查询功能，呈现给用户综合审计记录中的相关字段的信息，同时用户还可以选择查看更加详细的系统调用级别的审计记录信息。

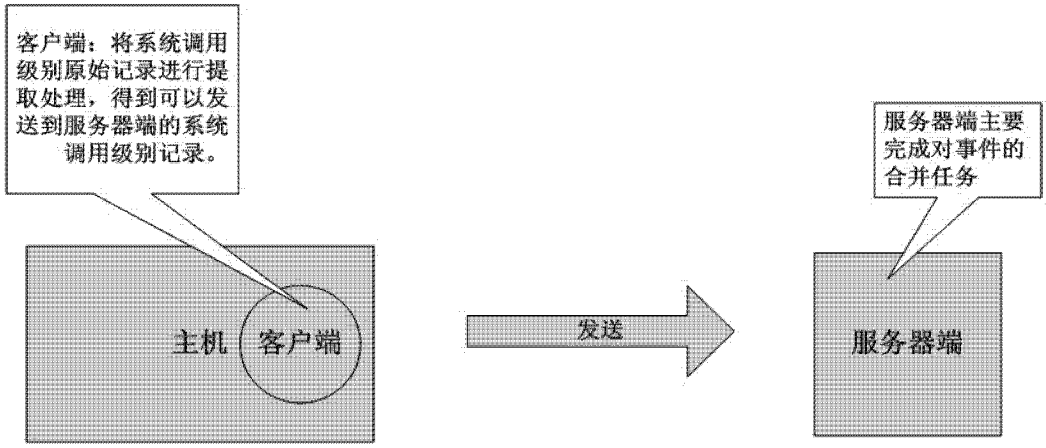

客户端：将系统调用级别原始记录进行提取处理，得到可以发送到服务器端的系统调用级别记录。
主机
客户端
发送
服务器端主要完成对事件的合并任务
服务器端

图 1

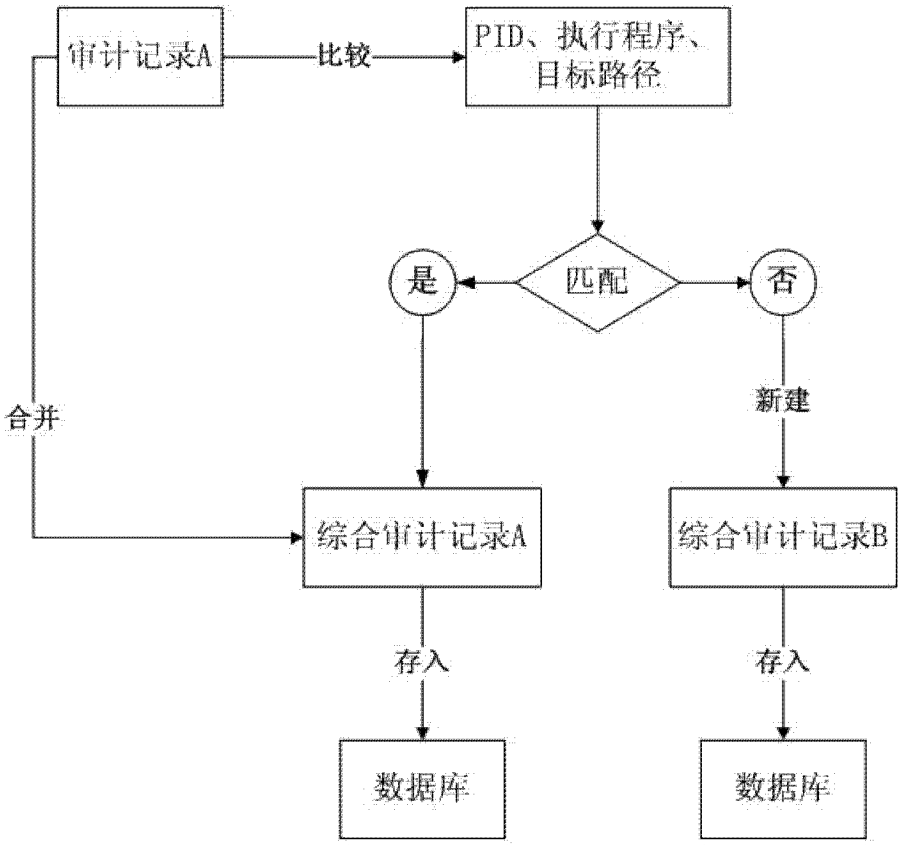

审计记录A
比较
PID、执行程序、目标路径
匹配
是
否
合并
新建
综合审计记录A
综合审计记录B
存入
存入
数据库
数据库

图 2

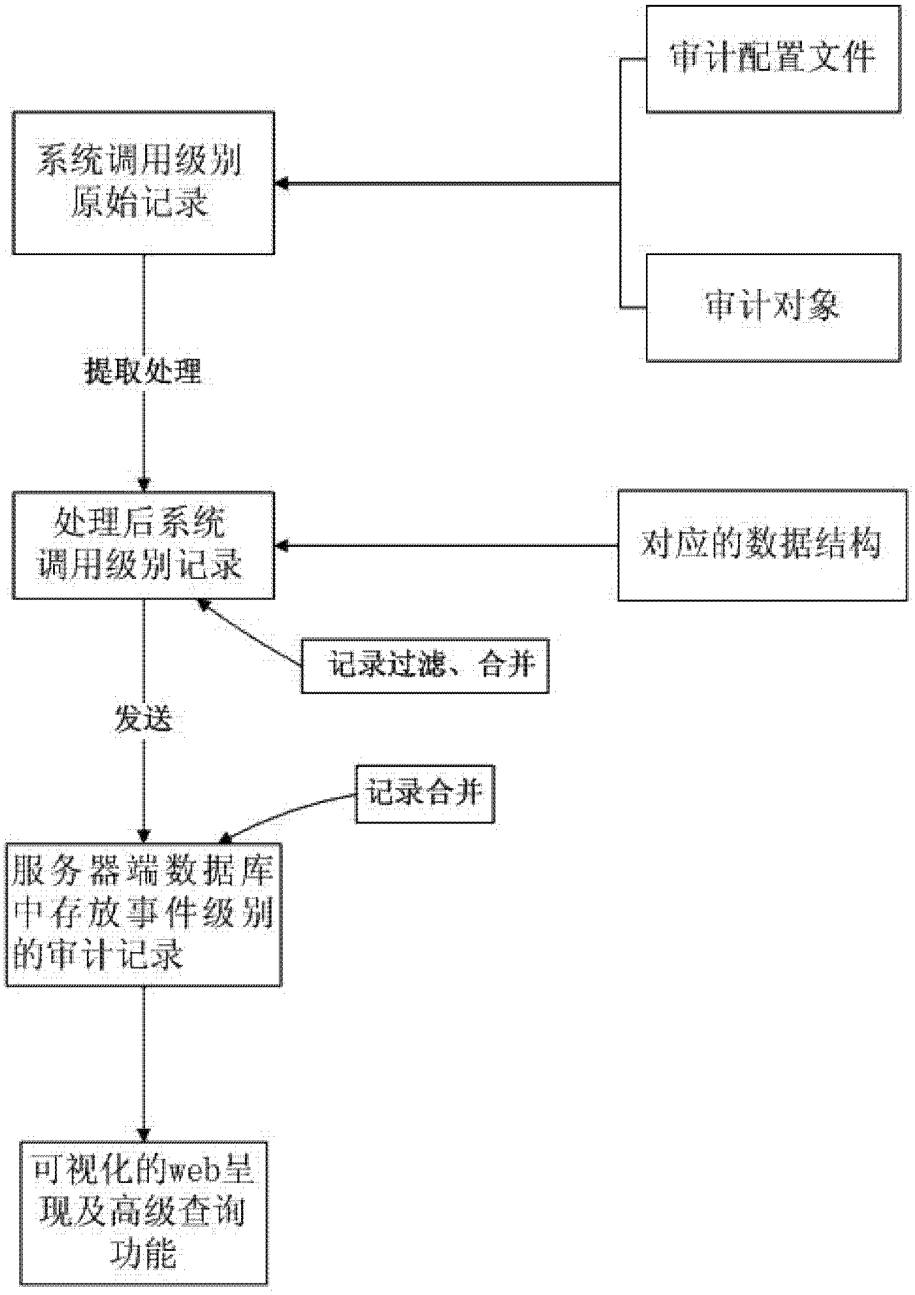

审计配置文件
系统调用级别原始记录
审计对象
提取处理
处理后系统调用级别记录
对应的数据结构
记录过滤、合并
发送
记录合并
服务器端数据库中存放事件级别的审计记录
可视化的web呈现及高级查询功能

图 3