软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(5):996−1008 [doi: 10.3724/SP.J.1001.2011.03759]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

用于自动证据分析的层次化入侵场景重构方法*

伏 晓[1], 石 进[1,2+], 谢 立[1]

[1](计算机软件新技术国家重点实验室(南京大学),江苏 南京 210093)

[2](南京大学 国家保密学院,江苏 南京 210093)

Layered Intrusion Scenario Reconstruction Method for Automated Evidence Analysis

FU Xiao[1], SHI Jin[1,2+], XIE Li[1]

[1](State Key Laboratory for Novel Software Technology (Nanjing University), Nanjing 210093, China)

[2](School of National Information Security, Nanjing University, Nanjing 210093, China)

+ Corresponding author: E-mail: zgnjack@gmail.com

Fu X, Shi J, Xie L. Layered intrusion scenario reconstruction method for automated evidence analysis. *Journal of Software*, 2011,22(5):996−1008. http://www.jos.org.cn/1000-9825/3759.htm

Abstract: In order to analyze intrusion evidences automatically, a layered method for reconstructing intrusion scenario is proposed. It includes 3 main phrases. First, the intruder's abstract steps and the relationships between them are reconstructed by the alert correlation. Secondly, detailed behaviors of each step are reconstructed based on attack signatures and the OS-Level dependency tracking. Finally, the results are mapped and refined, and a behavior graph is generated. This graph can describe the completed intrusion process. The experiments on DARPA 2000 prove that the results are not only easy to understand, but are also full and accurate. Hence, it is fit to be presented in the court. Compared with current methods, this method shows more advantages. For example, it can process more complex scenarios.

Key words: intrusion forensic; evidence analysis; scenario reconstruction; dependency tracking; alert correlation

摘 要: 为了能够自动分析入侵证据,提出了一种层次化入侵场景重构方法.其原理是:首先,基于报警关联技术重构出入侵者的抽象攻击步骤及步骤间关系;然后,基于攻击特征和依赖追踪技术重构出各步骤的行为细节;最后,通过两层重构结果的彼此映射,调整获得完整的入侵行为图.基于 DARPA 2000 的实验结果表明,该方法的重构结果准确性和完备性均比较高,而且抽象与细节相结合的表示方法更易理解,也更适合作为法律证据.而与现有方法相比,该方法在重构场景的完整性、适用行为的复杂性以及方法安全性等方面也有一定的改善.

关键词: 入侵取证;证据分析;场景重构;依赖追踪;报警关联

中图法分类号: TP393 **文献标识码**: A

计算机取证(computer forensic)是专门研究如何按照符合法律规范的方式收集、处理计算机犯罪证据的学科[1].由于其重要意义和实用价值,该领域在近几年吸引了人们越来越多的关注.入侵取证(intrusion forensic)是计算机取证的子领域之一.因为入侵在计算机犯罪中占有很大比例,所以对入侵取证的研究一直是取证领域关

* 基金项目: 江苏省自然科学基金(Bk2009465)
收稿时间: 2009-04-09; 定稿时间: 2010-10-10

© 中国科学院软件研究所 http://www.jos.org.cn

注的重点.入侵取证的一个关键阶段是分析入侵证据,即根据收集到的原始证据得出推理性结论,例如重构入侵者的行为过程(即入侵场景).因为该阶段的处理结果需要作为法律证据提交,所以分析方法必须客观、可靠,分析结果也应具备翔实、准确、能够被法律界人士理解等特点.但是,现有入侵证据分析方法却以人工处理为主,这种方式缺点很多:首先,人工分析主观、易出错、分析过程难以重复,所以获得的证据法律采信度不高;其次,入侵正变得日益频繁、复杂,面对海量数据和繁重的取证任务,人工分析已很难满足需求.针对这些问题,近年来研究者提出应当用自动化分析方法取代传统的人工分析.这是因为自动化分析不仅能提高效率、避免主观解释、增加结果的准确性,而且方法过程可公开发表以接受评判,因此,所得证据的说服力和法律采信度也更高.

目前,对自动证据分析的研究仍处于起步阶段,其中,有关自动入侵场景重构技术的研究是近几年的热点.根据分析对象的不同,该技术可分为针对单个主机的场景重构和针对整个网络的场景重构.本文关注的是前者,在这方面,现有的方法不多而且存在不少缺陷,例如仅能重构部分入侵场景、不能处理复杂行为、重构结果难以理解、方法安全性低等.为此,本文提出了一种自动化方法——层次化入侵场景重构.其原理是把场景重构分成两个层次:高层行为场景重构和低层行为细节的重构,对高层场景的重构依靠现有报警关联技术完成,而在低层细节重构中则使用了一种基于攻击特征的依赖追踪方法.该方法是对现有依赖追踪技术改进之后得到的.上述两个层次的重构紧密相关:一方面,高层场景重构得到的报警关联图可以指导低层依赖追踪;另一方面,低层依赖关系也可以用来验证及调整高层场景.为了描述最终重构结果,我们还设计了一种层次化行为图,该图既能展现入侵者抽象的攻击步骤以及步骤间关系,又能描述每个步骤的实现细节.我们为本文方法创建了原型实现并在 DARPA 2000 上进行了实验.实验结果表明,本文方法能够有效地重构出复杂的入侵过程,而且在结果的完备性、准确性、易理解性以及方法安全性等方面比现有方法均有一定的改善.

本文第 1 节介绍相关工作.第 2 节~第 5 节介绍本文方法及其各阶段处理过程.第 6 节描述本文方法在 DARPA 2000 数据集上的实验及其与现有方法的比较结果和相关分析.第 7 节总结并展望未来工作.

1 相关工作

在针对单个主机的自动入侵场景重构方面,现有方法主要包括 3 种:① 基于操作系统层对象的依赖追踪技术[2,3],即首先记录那些会导致进程、文件等操作系统层对象间依赖关系的事件/系统调用(如进程 *A* 写文件 *B*),然后,在检测到入侵时根据历史记录从检测点开始向前追溯,重构出所有相关对象间的依赖关系图,该图即反映了入侵者的行为过程.② 基于已知场景的自动重构,即先用某种模型(如 Bayesian 网络[4]、本体加规则[5]等)描述可能的犯罪场景/正常行为过程,然后比较原始证据与已知模型以确定真实犯罪过程.③ 基于模拟的自动场景重构[6],即首先假设可能导致现有证据的候选场景,然后在类似环境下模拟这些场景,最后比较现有证据与模拟结果,选择特征最匹配的候选场景作为犯罪过程.与上述方法相比,本文方法优点如下:① 与依赖追踪技术相比,本文方法基于攻击特征重构低层行为,能够克服其过于依赖初始检测点(当检测点不准确或无法获取时重构很难完成)以及只能重构部分入侵行为(未在事件记录中留下痕迹的攻击或与检测点无依赖关系的行为无法被重构)的缺陷,而且方法生成的行为图也比依赖图更加翔实、易懂.② 与基于已知场景的方法相比,本文方法能够识别未知行为模式,也无须预先建模已知场景.③ 与基于模拟的方法相比,本文方法的分析过程不存在任何假设及模拟,这样即可避免其“难以保证假设场景已穷尽所有可能”以及“模拟环境与真实环境无法完全一致”等缺陷.方法的重构结果也更具说服力.④ 与上述方法相比,本文方法更适合处理复杂入侵过程.采用高、低两个层次彼此印证的证据流也使得结果的可信度更高,误/漏报更少.

在针对整个网络的自动入侵场景重构方面,现有方法主要分为两类:第 1 类关注入侵紧密相关主机的角色识别[7];第 2 类则关注入侵真实来源的定位(即归因技术[8]).目前,第 1 类方法还不是很多,第 2 类方法则是研究热点,相关技术包括 IP 追踪、踏脚石检测等等.本文方法与基于网络的入侵场景重构关注点不同(我们的调查对象是单个主机),但两者可互为补充.在调查复杂网络入侵时,常常需要这两类技术的结合使用.

除黑客入侵外,计算机犯罪还有很多,例如网络骚扰(cyberstalking)、E-mail 勒索、保存及散布色情图片等.对于这些犯罪行为的重构也是重要研究课题.相关方法已有不少,例如,文献[9]即提出了一种专门针对网络骚扰

© 中国科学院软件研究所 http://www.jos.org.cn

的自动取证系统,文献[10]则提出可用有穷状态机来自动重构 E-mail 勒索者的行为.与黑客入侵相比,这些犯罪过程要简单得多,因此其重构方法一般不适合重构复杂入侵行为.不过,现有的一些入侵场景重构方法(如基于已知场景或模拟的方法)对此类犯罪行为重构却同样适用.本文方法目前尚不能处理此类犯罪行为.

2 层次化入侵场景重构过程

本文方法的处理过程如图 1 所示.① 数据准备.其主要工作是数据采集、净化和格式统一,要收集的数据包括 IDS 报警和操作系统层事件记录,IDS 报警可来自多个任意类型的入侵检测系统,操作系统层事件记录则由预先安装在被取证机器中的事件监控机制产生.现有的监控机制很多,例如虚拟机监控[3]、专用嵌入式事件监控机制[2]、利用 BSM 等安全审计模块执行事件监控任务等.② 高层行为场景重构.本阶段的主要工作是重构出入侵者的攻击步骤及步骤间关系.这些步骤是抽象的,描述了入侵者的阶段性任务或目标.在本文方法中,攻击步骤用融合 IDS 报警得到的超报警来表示,步骤之间的关系则用报警关联算法来识别.③ 低层行为细节重构.本阶段的任务是为高层场景图中的每个步骤重构相应的行为细节,例如,为达到某阶段性目标执行了哪些程序或创建、删除了哪些文件.行为细节用操作系统层的对象及系统调用来描述,行为过程的重构则借助现有的依赖追踪技术实现.④ 层间映射.本阶段的主要工作是对高、低两个层次的重构结果加以合并、调整,以便生成描述入侵过程的完整行为图.在上述 4 个阶段中,后 3 个是本文方法的重点所在,下文将对其详细介绍.

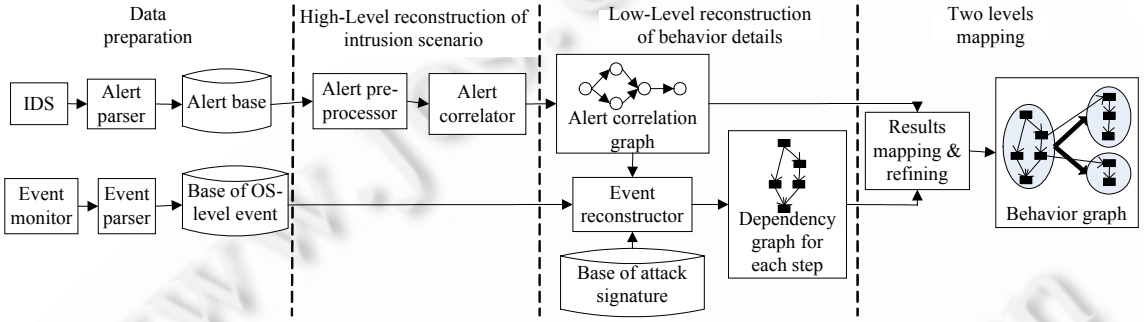

Fig.1 Process of layered intrusion scenario reconstruction

图 1 层次化入侵场景重构过程

3 高层行为场景重构

如第 2 节所述,本阶段的任务是借助报警关联技术重构出入侵者的高层攻击场景.报警关联是产生于入侵检测领域的数据分析技术,它能够从大量报警中识别彼此相关的报警,并据此构造出描述攻击者行为过程的关联图.这一阶段的具体过程如下:首先是报警预处理,即用现有的报警删减技术[11]去除冗余/无关报警,再用报警验证技术[12]去除错误报警.上述步骤的目的是提高关联的效率及准确性;然后是报警的聚类、融合,即将处理后的报警中与同一阶段性任务相关的报警聚为一类,并将每个聚类融合成超报警.这样,一条超报警即可代表一个抽象攻击步骤.其中,聚类可借助数据挖掘中的聚类算法完成.融合的方法[12]则包括合并相同报警属性、用抽象属性代替具体属性等;最后是用关联算法分析超报警,识别它们之间的关系并创建报警关联图.现有的关联算法很多,我们目前采用的是 Peng 等人提出的基于前因后果的关联算法[13].其主要思想是,将每条超报警类型描述为(事实,前提,结果)三元组.其中,前提和结果均是谓词的逻辑范式.对于任意超报警 A,若存在超报警 B 发生在 A 之前且 B 的结果中含有与 A 的前提谓词相同的部分,则称 B prepare for A.此时,可关联这两条报警(即在关联图中添加代表 A,B 的结点以及从 A 指向 B 的边).

© 中国科学院软件研究所 http://www.jos.org.cn

4 低层行为细节重构

本阶段的目标是为高层关联图中的每个步骤(即超报警)重构出行为细节.为此,我们先抽象出不同类型攻击对应的特征(详见第 4.1 节),然后再根据攻击特征用依赖追踪技术完成细节重构(详见第 4.2 节).

4.1 攻击特征的获取及描述

在本文方法中,攻击特征被用作高层报警与低层对象间的桥梁.其定义如下:

定义 1(攻击特征(attack signature)). 攻击特征是从操作系统层面对攻击时所采用的权限、工具、动作、作用点、目的及后果等方面特点的描述.它用三元组(positive signature,optional signature,negative signature)来表述.其中,positive signature 表示为实现攻击不可或缺的条件;optional signature 表示可选条件;negative signature 表示反面证据,若这类证据出现,则攻击不可能发生.每种 signature 均由进程、文件等操作系统层的对象以及相关的系统调用组成.

由上述定义可知,若知道一条超报警的攻击特征,就可以得知该报警大致会对应哪些操作系统层对象/系统调用,从而也就为该步骤的低层行为细节重构提供了指导.攻击特征属于先验知识,由领域专家预先分析得到并保存于知识库中.对于利用系统漏洞发起的攻击,其特征可通过分析漏洞原理得知.例如,分析 CERT/CC advisories 等公开的漏洞库,可以得知利用某个漏洞时必须执行的系统调用或必须读、写的文件.而对于漏洞无关攻击,则需要总结攻击行为的表现来生成相应特征.在总结行为表现时,有时还需区分攻击发起者和被攻击者.例如,当某台主机被用来发起 DDoS 攻击时,其表现一般包括:① 安装、使用某种已知 DDoS 工具或执行某未知 DDoS 脚本;② 系统针对某一目标大量发送数据包,即出现大量用于数据发送且针对同一目标的系统调用(例如 send).而当主机是被攻击目标时,其表现又会是另外的情形(即接收到大量数据包).

为实现自动场景重构,我们必须用机器可理解的方式描述攻击特征,使其成为重构算法可利用的知识,这种描述应当简单、易懂、易扩展.为此,我们基于谓词逻辑设计了一种特征描述语言 ASDL(attack signature description language).在 ASDL 中,攻击特征的 3 种 signature 均可以用 signature 表达式来表示.其定义如下:

定义 2(signature 表达式(signature expression)). 一个 signature 表达式由两个部分组成:第 1 部分是事实范式(fact paradigm),即由对象或施加在对象上的系统调用组成的逻辑合并;第 2 部分是属性列表(attribute list),即对上述对象或系统调用所具有的属性的详细描述.这两部分之间用保留字“WHERE”来分割,即

Signature 表达式::=事实范式 WHERE 属性列表,

其中,事实范式及属性列表的 BNF 定义如下:

- 事实范式::=事实{逻辑连接符 事实}
- 逻辑连接符::=∧|∨
- 事实::=系统调用(操作系统层对象)|操作系统层对象
- 系统调用::=导致进程/进程依赖的事件|导致进程/文件依赖的事件|导致进程/文件名依赖的事件
- 操作系统层对象::=进程|文件|套接字|管道
- 属性列表::=[属性表达式{;属性表达式}]
- 属性表达式::=进程属性表达式|文件属性表达式|套接字属性表达式|管道属性表达式|系统调用属性表达式
- 进程属性表达式::=PROCESS 进程名 WITH 进程属性名 进程属性值{,进程属性名 进程属性值}
- 进程属性名::=FUNCTION|OWNER|PRIVILEGE|PLATFORM
- 文件属性表达式::=FILE 文件名 WITH 文件属性名 文件属性值{,文件属性名 文件属性值}
- 文件属性名::=FUNCTION|OWNER|ACCESS MODE
- 套接字属性表达式::=SOCKET 套接字名 WITH 套接字属性名 套接字属性值{,套接字属性名 套接字属性值}
- 套接字属性名::=SOURCE|DESTINATION

© 中国科学院软件研究所 http://www.jos.org.cn

- 管道属性表达式::=PIPE 管道名 WITH 管道属性名 管道属性值{,管道属性名 管道属性值}
- 管道属性名::=SOURCE|DESTINATION
- 系统调用属性表达式::=SYSTEMCALL 系统调用 WITH 系统调用名 系统调用值{,系统调用名 系统调用值}
- 系统调用属性名::=ARUEMENT|EXECUTOR

为了增强 ASDL 的表达能力,我们还定义了几个特殊的保留字:任意(*any*)、所有(*all*)、每个(*each*)和很多(*many*).其中,*many* 可由调查者给出具体定义.例如,本文系统中它被规定为每秒大于等于 3.这些保留字可与保留字 *process*,*file*,*socket*,*pipe*,*systemcall* 连用,例如,可用 *any process* 指代任意进程.另外,进程、文件等各类属性名的定义在实际应用中可视真实情况扩展.而在系统调用的定义中,每类事件具体包含哪些则与被监控的系统平台有关:Unix 系统中,导致进程/进程依赖的事件包括 fork,clone 等,导致进程/文件依赖的事件包括 write,read,chmod 等,导致进程/文件名依赖的事件则包括 open,creat,rename 等.而在 Window 系统中,它们代表的又是另一套系统调用.

借助 ASDL,前文所述的 DDoS 攻击的特征(被取证主机是 DDoS 攻击的发起者)可以用如下 signature 表达式来表示,该表达式可作为此类攻击的 positive signature:

(*many* sendto(*any socket*)∨*many* sendmsg(*any socket*)∨*many* send(*any socket*)∨*many* sendto(*any pipe*)∨
many sendmsg(*any pipe*)∨*many* send (*any pipe*))∧*any process*
WHERE
PROCESS *any* WITH FUCTION DDoS

4.2 基于攻击特征的行为细节重构算法

定义好每类攻击的特征后,还需要把具体攻击步骤(即超报警)与这些特征关联起来.为此,我们在超报警的基本属性之外新增了“攻击特征”这一属性,其取值为该报警类型对应的攻击特征在特征库中的编号.

知道了关联图中每条超报警对应的攻击特征之后,就可以为其重构相应的行为细节了.重构过程如下(如图 2 所示):

Step 1. 筛选相关事件记录.即根据关联图中每条超报警的时间戳筛选可能对应该步骤的低层事件.因为网络延迟的存在以及 IDS 报警的滞后性,相关事件的时间范围与超报警时间戳一般会有差异.为了尽可能完整地获得相关事件,我们将超报警时间戳扩大了一定范围,并根据扩展后的时间段来查找对应事件.最后,所有筛选出的记录均会被加入相关事件集 S_{Event}.

Step 2. 基于 positive signature 生成依赖图.对于关联图中的每条超报警,若其攻击特征的 positive signature 非空,则根据该 signature 及 S_{Event},利用基于 signature 的依赖图生成算法 SignatureBasedTacer(如图 3 所示)为其重构出依赖图 G_P.

Step 3. 基于 optional signature 补充依赖图.若该报警的 optional signature 非空,则根据该 signature 及 S_{Event},利用 SignatureBasedTacer 算法重构得到依赖图 G_O.比较 G_O 和 G_P,将 G_O 中存在但 G_P 中缺少的部分添加到 G_P 中.

Step 4. 基于 negative signature 裁剪依赖图.若该报警的 negative signature 非空,那么首先根据 negativesignature 及 S_{Event},利用 SignatureBasedTacer 算法重构得到依赖图 G_N.然后比较 G_N 与 G_P,若两者有重叠,则将重叠部分从 G_P 中删除.需要注意的是,如果某重叠结点还有不在 G_N 中的子女,则该结点应予以保留.

上文提到的基于 signature 的依赖图生成算法(SignatureBasedTacer)基本思想如下:① (图 3 第 2 行~第 6 行)检查 signature 表达式中事实范式部分列出的每个事实,若其出现在 S_{Event} 中,则将其加入队列 Q,Q 按事实在 S_{Event} 中出现的时间先后排序.若同一事实多次出现,则用最晚一次出现的时间作为排序依据.事实在 S_{Event} 中的一次出现是指 S_{Event} 中存在一条包含该事实的事件记录.当检查事实是否出现时,不仅需要查看事实名称,还应查看属性列表中所指出的其他属性是否匹配.② (图 3 第 7 行~第 14 行)检查 Q 中的事实能否使 signature 表达式的事实范式部分为真,若不能,则说明该 signature 不被满足.若当前处理的是 positive signature,则意味着该攻击不

© 中国科学院软件研究所 http://www.jos.org.cn

可能发生,此报警为误报.此时,应先将该报警从关联图中删除,再转而处理关联图中的下一条报警.若当前处理的是 optional/negative signature,则直接退出即可.若 Q 能使事实范式为真,则继续执行算法余下部分.③ (图 3 第 15 行~第 18 行)以 Q 中的最后一个事实为初始检测点,用 TraceBack 算法重构依赖图.然后检查依赖图,将图中已出现的事实从 Q 中删除.之后,对 Q 中余下的事实重复上述步骤,直至 Q 为空.

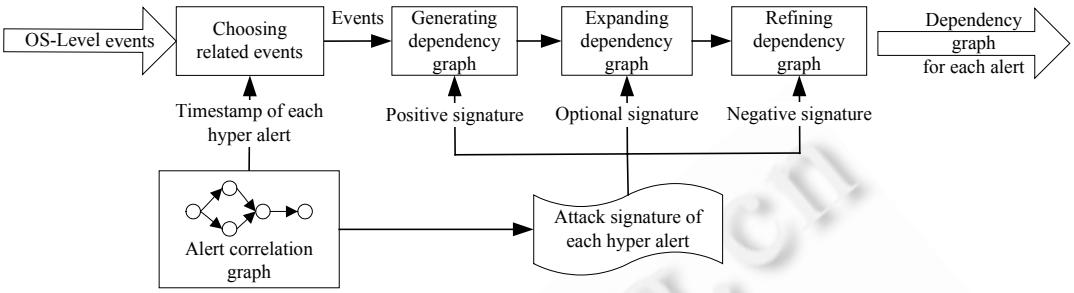

Fig.2 Process of low-level behavior details reconstruction

图 2 低层行为细节重构过程

```
Algorithm. SignatureBasedTacer.
Input: Exp            //Signature Expression
       S_Event        //A set of events related to the hyper alert to be processed
Output: Dependency Graph.
01 begin
02     Queue Q   //Q contains candidate initial points for TraceBack
03     for each fact f in the Fact Paradigm of Exp Do
04         if S_Event contains f then
05                 Put f into Q;   //Q is sorted ascendingly by the time that the facts appears in S_Event;
06         end if
07     if Q can't satisfy the Fact Paradigm of Exp then
08         if Exp is positive signature then
09             Deleting False Positive from Alert Correlation Graph;
10             return to Step 1 and continue to process the next alert;
11         else if Exp is optional signature or negative signature then
12             return;
13         end if
14     end if
15     While Q is not empty Do
16         f=the last item of Q;
17         TraceBack(f,S_Event);   //Generate Dependency Graph based on the last item of Q
18         Remove f and other fact that has already appeared in the Dependency Graph from Q;
19 end
```

Fig.3 Signature-Based algorithm for generating dependency graph

图 3 基于特征的依赖图生成算法

在 SignatureBasedTacer 中,第 17 行的 TraceBack 算法用于依赖图实际构造.TraceBack 是对 Samuel 的依赖图生成算法[3]加以改进之后得到的.算法的基本过程如下(图 4 中第 2 行~第 7 行以及第 18 行~第 25 行是我们新增的部分):① (图 4 第 2 行~第 5 行)在 Samuel 的算法中,初始检测点是操作系统层对象,而我们输入的检测点为事实.因此,在构造依赖图之前需要先从事实中抽取相应操作系统层对象.② (图 4 第 6 行、第 7 行)根据输入事实 P 限定相关事件的范围,即截取 S_{Event} 中从最早的事件开始到最后一个包含 P 的事件为止的部分,将这些事件加入集合 S'_{Event}.③ (图 4 第 8 行~第 17 行)从 S'_{Event} 中最晚的事件开始,逐条读取事件并构造依赖图:对于每条事件,首先评估其是否影响现有依赖图中的对象.若是,则查看事件中起影响作用的对象(源对象)是否已存在于依赖图中.若没有,则先将其加入依赖图,再在图中添加一条从源对象指向被影响对象(目标对象)的有向边.依赖图中的每个对象均有一个时间阈值,用于标识与该对象相关的事件可能发生的最晚时间.④ (图 4 第 18 行~第 25 行)为进程、文件结点加上时间戳.该步骤是为之后的合并依赖图做准备(见第 5 节中的 Step 2).进程用创建时间做时间戳,可防止同名的不同进程被合并.文件加上最近一次修改时间,则可防止不同版本的同一文件被合并.

© 中国科学院软件研究所 http://www.jos.org.cn

在我们的依赖图中,同一文件的不同版本需用不同结点来表示.

```
Algorithm. TraceBack.
Input: fact P    //Initial detection point
       S_Event   //A set of events related to the hyper alert to be processed
Output: Dependency Graph.
01 begin
02      if P is an OS-Level object then    //Generate initial graph for P
03          add P to graph
04      else   //P is a system call performed on curtain OS-Level object
05          add the OS-Level object contained by P to graph;
06      Find the latest event that contains P in S_Event, and record it as E_p.
07      Create a subset S′ for S_Event. S′ contains all the events whose time belong to [the earliest event time in S_Event, the time of E_p].
08      for each event E in S′ Do    //Read events from latest to earliest
09          for each object O in graph Do
10              if E affects O by the time threshold for object O then
11                  if E's source object not already in graph then
12                                                  add E's source object to graph;
13                                                  set time threshold for E's source object to time of E;
14                                          end if
15                                          if the edge from E's source object to E's sink object not already in graph then
16                                                  add such an edge;
17                                          end if
18                                          if E is create a file or fork a process then
19                                                  set timestamp of E's sink object to time of E;
20                  else if E is write a file then
21                          if timestamp of E's sink object has already exit then
22                              add E's sink object to graph again;
23                          end if
24                          Set timestamp of E's sink object to time of E;
25                                      end if
26                              end if
27 end
```

Fig.4 TraceBack algorithm

图 4 TraceBack 算法

如果借助上述算法为第4.1节末尾提到的signature构造依赖图,结果将如图5所示.该图是有向无环连通图,图中的结点代表进程、文件等操作系统层对象.对于任意一对结点 $n1$,$n2$,图中存在一条从 $n1$ 到 $n2$ 的边当且仅当 $n1$ 影响 $n2$(或称 $n2$ 依赖于 $n1$).

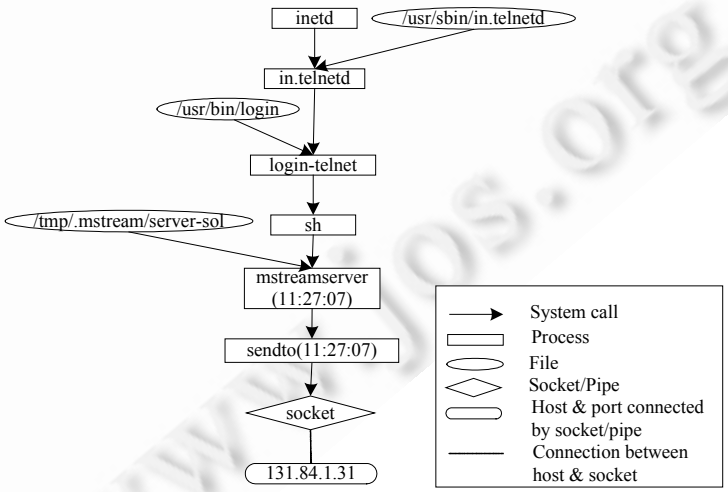

Fig.5 Dependency graph for Stream_DoS

图 5 Stream_DoS 的依赖图

© 中国科学院软件研究所 http://www.jos.org.cn

5 两层场景映射

重构完高层场景及每个步骤的行为细节后,还需要映射、调整重构结果,以便生成描述入侵过程的完整行为图.该图既要能够展现抽象攻击步骤及步骤间的关系,又要能够描述每个步骤的细节.为保证行为图的准确性,构造时还需要删除错误关联,过滤无关行为以及识别漏报步骤.行为图的构造过程如下:

Step 1. 以高层场景重构得到的关联图为初始行为图(记为 G),将描述各攻击步骤行为细节的依赖子图添加到 G 中对应的超报警结点中.若报警无对应依赖子图,则无须添加.

Step 2. 根据 G 中超报警间的关系连接依赖子图.若关联图中存在从报警 A 指向报警 B 的边,A,B 对应的依赖子图分别为 Ga,Gb,那么可通过以下步骤连接 Ga,Gb:

Step 2.1. 合并 Ga,Gb 的公共结点.若结点是文件/进程,则判断两个结点是否相同时需考虑其创建/修改时间,以免错误合并同名进程或同一文件的不同版本.若结点是套接字/管道,则只要连接到同一地址即可视为相同结点.依赖子图中可能存在入侵者修改过的文件未标注时间戳,这是由于创建/修改该文件的事件不在"相关事件集"内所致,合并前需为此类结点补上时间戳,这可以通过扫描全部相关事件记录来完成.但若结点并非在入侵过程中修改/创建,则无须标注时间.

Step 2.2. 寻找从 Ga 到 Gb 的路径:

(1) 截取[begin_time of Ga,begin_time of Gb]时间段内的事件记录,其中,begin_time of Gi (i=a/b)指 Gi 所描述的事件中最早发生的事件对应的时间.

(2) 将 Gb 中入度为 0 的进程/文件结点加入队列 Q.但若文件结点的修改时间早于 begin_time of Ga(说明该结点依赖的对象不在 Ga 中),则无须加入 Q.

(3) 若 Q 非空,则从队尾开始分别以 Q 中的每个对象为起点,用 TraceBack 算法根据截取的事件记录构造依赖图.若重构出的依赖图包含与 Ga 相同的结点,则将该图添加到 G 中,否则将其丢弃.

Step 3. 寻找遗漏路径.关联图中可能存在遗漏的连接或攻击步骤,而各步骤对应的依赖子图之间也可能存在其他路径,因此有必要继续遍历余下的事件记录,查找这类遗漏路径.具体方法如下:

(1) 在当前行为图 G 中寻找有无其他重复结点,若有则加以合并.

(2) 在 G 中寻找有无其他入度为 0 的文件/进程结点,若有则将其加入队列 Q.

(3) 若 Q 非空,则从队尾开始分别以 Q 中的每个对象为起点,根据 Step 2 中遍历余下的事件记录用 TraceBack 算法构造依赖图,并将得到依赖子图添加到现有依赖图中.

Step 4. 删除错误关联.文献[14]指出"若报警 A,B 因果相关且 A prepare for B,那么 A,B 对应的依赖子图 Ga,Gb 之间必定存在从 Ga 到 Gb 的路径(或存在公共结点)".我们可以根据该结论用低层依赖关系来识别超报警间的错误因果关联,即检查 G 中每一对因果相关报警,若其对应依赖子图不满足上述条件,则可判定它们之间的关联是错误的,应将描述该关联的有向边从 G 中删除.

Step 5. 过滤无关行为.因为属于同一攻击场景的所有攻击行为必定彼此相关(否则就不应归入同一场景),所以同一场景对应的低层依赖图应当是连通图.因此,若 G 中出现独立行为分支,则应将其删除.如果这类分支对应的超报警也独立于其他报警,那么也应将其从 G 中删除.

Step 6. 识别漏报步骤.G 中有时会出现不属于任何超报警的低层行为分支.它们代表的很可能是未知或漏报攻击.我们为这类分支建立了虚拟攻击步骤,并在行为图中特别标识出来,以便调查者进一步分析.例如,图 6 中的 Install_mstream 即是这样的虚拟步骤.

© 中国科学院软件研究所 http://www.jos.org.cn

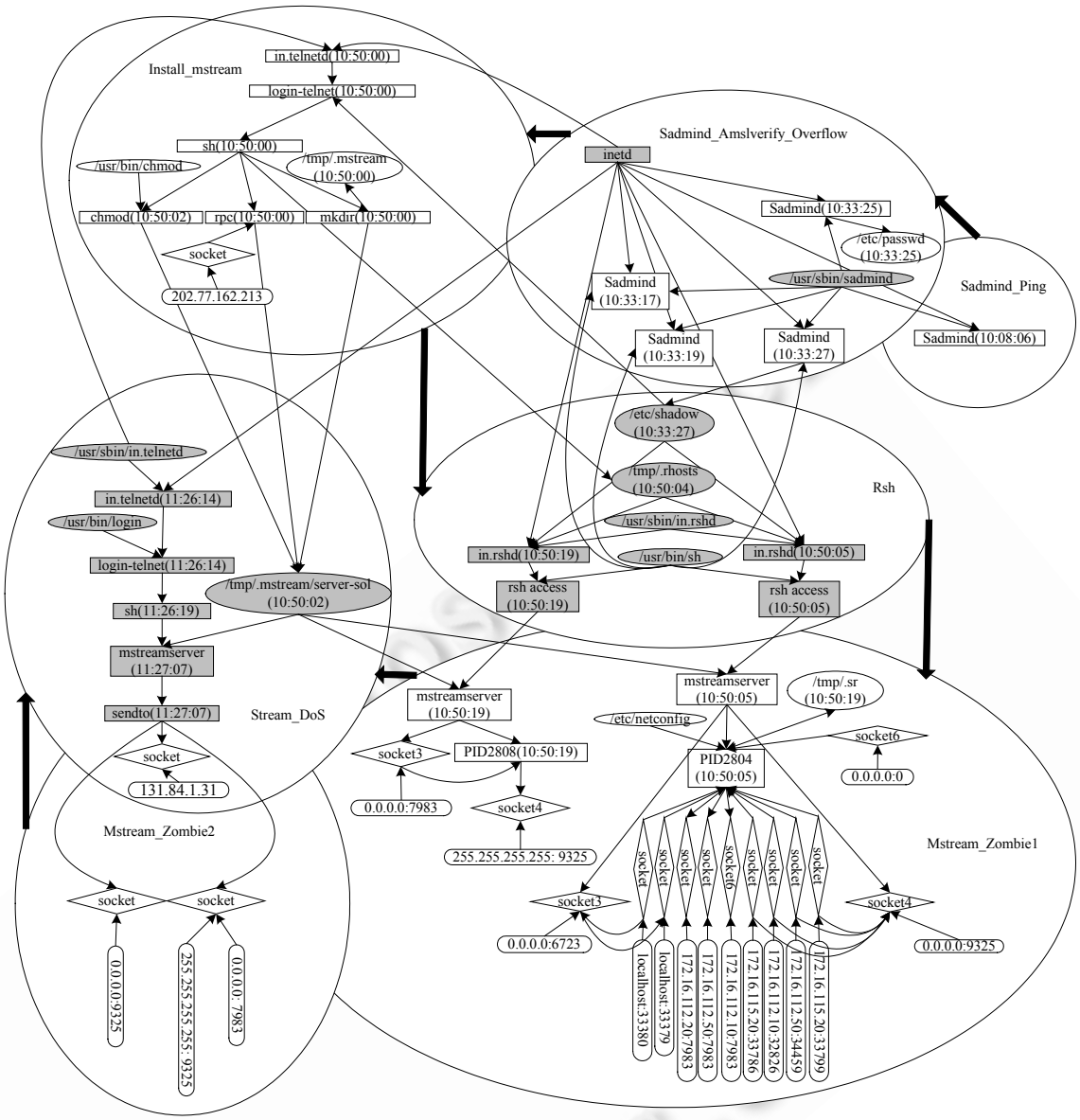

Fig.6 Intrusion behavior graph for host mill in LLDoS 1.0 scenario

图 6 LLDoS 1.0 场景下主机 mill 的入侵行为图

6 实验与分析

6.1 DARPA 2000实验结果

为验证本文方法,我们为其创建了原型实现,并在 DARPA 2000 数据集上对原型进行了实验.DARPA 2000 模拟了黑客入侵的过程,并提供了相关主机的 BSM 审计日志以及被侵入网络的 TCP dump 记录.将这些日志/记录输入 IDS 后即可产生相应报警,而 BSM 审计日志经过处理后可充当操作系统层事件记录.

实验采用的是 DARPA 2000 中的 LLDoS 1.0 场景.它包含以下几个阶段:① 从远程主机向实验网络发起 IP Sweep 攻击以刺探活动主机;② 寻找含有 Sadmind 漏洞的活动主机;③ 利用 Sadmind 漏洞进入这些主机;④ 在

© 中国科学院软件研究所 http://www.jos.org.cn

成功进入的主机(mill,pascal 和 locke)上安装 mstream DDoS 软件;⑤ 利用安装好的工具发起 DDoS 攻击.为检验系统能否重构上述攻击场景,实验步骤如下:首先获取 IDS 报警.为此,我们在一个安装了网络 IDS(realsecure network sensor)的独立网络中用 NetPoke 重放了 LLDoS 1.0 的网络事务;然后,将 IDS 报警及 BSM 审计日志输入原型系统;最后,由原型系统为各主机自动重构入侵场景.下文以主机 mill 为例给出了原型系统的重构结果.

在高层场景重构阶段,系统为 mill 重构的报警关联图如图 7 所示.其中,Sadmind_Ping 对应攻击阶段②,Sadmind_Amslverify_Overflow 对应阶段③.之后的 4 条报警则对应阶段④、阶段⑤.在低层行为细节重构阶段,原型系统又为图 7 中的每个攻击步骤重构了详细的实现过程.例如,图 5 即是 Stream_DoS 这一步骤对应的行为细节.为便于理解,我们用 Samuel 提出的依赖图过滤规则[3]从低层依赖关系中过滤了一些无关紧要的分支(例如某些例行的读系统文件行为),这种省略不会影响对入侵过程的描述.在两层场景映射阶段,经过验证及调整之前的重构结果,系统得到了如图 6 所示的完整入侵行为图.在该图中,灰色结点表示攻击步骤的重叠部分,白色结点表示每个步骤的独有部分,粗箭头表示高层关联,细箭头表示低层依赖.另外,每个攻击步骤用一个椭圆来表示,椭圆内是该步骤的行为细节.以实线为边界的椭圆代表 IDS 发现的攻击步骤,以虚线为边界的椭圆代表虚拟步骤.比较该图与真实入侵过程可知,它描述了 LLDoS 1.0 中的绝大部分攻击阶段,并能为每个攻击步骤提供翔实的行为细节.经检查发现,它描述的全部是真实入侵行为.但 LLDoS 1.0 中有一个攻击阶段(即利用 IP Sweep 刺探活动主机)未被图 6 展现.经分析发现,缺失的原因是 IDS 未能检测到 IP Sweep 攻击,若使用检测能力更高的 IDS 即可避免这种情况.比较图 6 与图 7 还可以看出,本文方法重构出的高层场景比仅用报警关联技术得到的场景更为准确.这是因为本文方法能够发现漏报攻击(如图 6 中的 Install_mstream),而且能够识别错误的因果关联(如图 7 中 Rsh 到 Mstream_Zombie2 的连接所示).

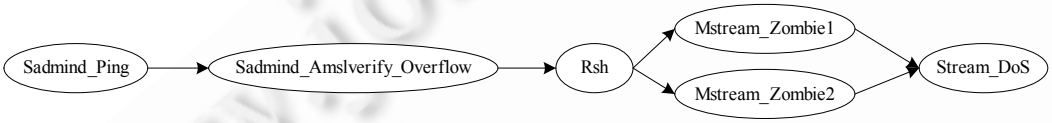

Fig.7 Alert correlation graph for host mill in LLDoS 1.0 scenario

图 7 LLDoS 1.0 场景下主机 mill 的报警关联图

6.2 方法比较

为了进一步地检验本文方法的效果,我们又将其与现有方法进行了比较.依赖追踪技术是目前最为流行的自动入侵场景重构方法,因此我们首选该技术作为比较参考.比较选用了著名的依赖追踪系统:BackTracer[3].为公平起见,我们在同样的数据集上用 BackTracer 对同一台主机(mill)进行了场景重构.重构的初始检测点为连接到 131.84.1.31 的套接字.它代表入侵场景的最后一个行为:向主机 131.84.1.31 发送大量数据包以执行 DDoS 攻击.重构结果如图 8 所示.本文方法与依赖追踪技术的比较将从完备性、准确性、易理解性和方法安全性这 4 个方面进行.其中,完备性指重构结果能否完整地展现入侵过程,准确性指重构结果中真实入侵行为所占的比例,安全性则指重构方法是否包含可能被攻击者利用的弱点.

在完备性方面,对比图 6、图 8 可知,BackTracer 的重构结果仅是本文方法重构结果的子集.对于 LLDoS 1.0 的 5 个攻击阶段,本文方法能重构出其中 4 个,完备性达 80%.BackTracer 却只能识别出 3 个(阶段③~阶段⑤),而且对每个阶段的描述也不完整(缺 Sadmind_Ping,Rsh,Mstream_Zombie1,Mstream_Zombie2 的对应行为),因此完备性<60%.在准确性方面,本文方法准确度可达 100%,BackTracer 的准确度却依赖于初始检测点的选择,并需要取证者来确保选择的正确性.这种方式效率低且有违自动化证据分析的初衷.在易理解性方面,BackTracer 仅重构出低层行为过程,若无专业知识,则很难看懂这些操作系统层对象及其依赖关系是何含义;而我们的行为图则先描述高层攻击场景,再把低层行为作为高层步骤的细节来展现,理解起来更加容易.在安全性方面,依赖追踪系统可能存在的问题包括[2]:① 入侵者使用隐蔽信道或“低控制事件”(即依赖图中大量存在、难以辨识的事件)来完成攻击,从而增大重构难度;② 入侵者将攻击与大量无关行为混合,使系统得到难以理解的大型依赖图.本文方法可较为有效地避免上述问题.这是因为本文方法直接根据高层场景中的步骤来选择需要处理的事件

© 中国科学院软件研究所 http://www.jos.org.cn

记录,并借助高层关联来指导低层依赖关系的重构.这可以比较有效地降低隐蔽信道、低控制事件以及无关行为对低层重构的影响.另外,本文方法在重构低层行为时不依赖于单个检测点,这也会增强方法的抗干扰能力.

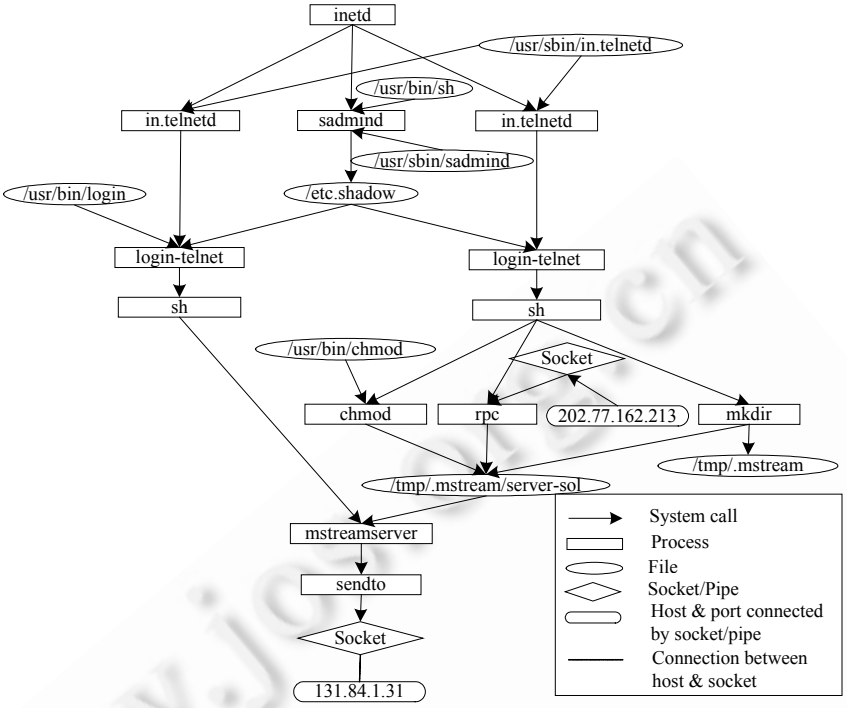

Fig.8 Dependency graph for host mill in LLDoS 1.0 scenario

图 8 LLDoS 1.0 场景下主机 mill 的依赖图

现有自动场景重构方法还包括基于已知场景和基于模拟的重构技术,目前,这两种技术尚无可供评估的原型系统,也缺少具体的实现细节,因此我们未对其进行实验.但从原理上看,这些方法与本文方法相比在重构复杂入侵行为时稍显不足:若用模拟方法重构 LLDoS 1.0 场景,需测试的可能场景将会非常多(因为黑客可用多种方式实现利用 mill 发起 DDoS 攻击的目标);而用已知场景法重构时,建模复杂的 LLDoS 1.0 场景更是非常困难.

6.3 方法相关法律问题分析

作为证据分析方法和简单的作为入侵场景重构方法具有本质区别:前者需要保证结果在法律上是可信、可靠的,而后者只需能够有效重构出入侵场景即可.本文目标是提供一种能够用于自动证据分析的入侵场景重构方法.因此,下文将就本文方法重构结果的可信度和可靠性加以讨论,以探讨方法用作取证工具的可能性.

可信的证据分析结果包含两层含义:首先,结果应当客观、原始、非人为虚构.其次,结果应能被法庭理解.本文方法采取了以下措施来保证结果的可信度:① 处理过程尽量减少人工干预,除攻击特征库需专家参与创建外,其他分析均能自动完成.另外,方法无需人为调节的参数,也无需假设或模拟来帮助分析,这些均有助于提高结果的客观性.② 重构结果与原始数据具有等价性.本文方法得到的行为图虽然与原始记录的表示形式不同,但它仅仅是对后者的抽象和整理,并不改变其实质内涵.③ 提高方法安全性,以防对分析结果的篡改.本文方法对入侵者的恶意干扰具有一定抵御能力(见第 6.2 节),而且系统可被安装在独立、安全的主机之上,以保证结果的可信度.④ 在结果的易理解性方面,本文生成的行为图对法律界人士而言也比较容易理解.

可靠性指证据分析结果应当是最佳、合法及可验证的.最佳指结果具有较低的误、漏报率并确实与被调查入侵密切相关.本文方法通过高低两层场景的互为佐证,可显著减少误、漏报.实验结果表明,其重构结果确实能够描述真实入侵过程而且漏报率低于 20%,误报率更可达到 0.在合法性方面,保护案件无关的隐私数据是关键

© 中国科学院软件研究所 http://www.jos.org.cn

问题.本文方法能够自动分析原始数据,重构结果也只与被调查入侵相关.这样即可减少调查者接触案件无关信息的机会,使调查过程更加合法.在可验证性方面,本文方法公开、透明,可接受领域专家的评估.而对原型系统的已有测试也表明,我们的系统比较稳定(不同调查者在相同数据集上分析同一入侵可得到相同结果).

数据源也是影响证据分析结果可信度和可靠性的重要因素.本文原型系统的数据源是 IDS 报警和 BSM 日志:BSM 是低层审计机制,记载了文件创建等安全相关系统事件;IDS 则是高层监控系统,其报警是分析系统/网络中的各类日志之后得到.这两类数据分处两层,可彼此印证.它们作为场景重构数据源的缺陷是 BSM 日志可能被入侵者删除/修改过,而 IDS 则存在报警量大和误、漏报率高等问题.这无疑会影响证据分析结果的说服力,对于系统的效率和准确度也会有一定削弱.为此,我们采取了如下措施:① 通过预处理净化数据源.例如,对于 IDS 报警,在分析前先用报警删减技术[11]去除冗余/无关报警,再用报警验证技术[12]去除错误报警.② 通过防篡改机制保护数据源.例如,将 BSM 日志实时备份到更为安全的机器,采用加密、强制访问控制等完整性保护机制等.另外,借助专用取证监控机制获得数据也是很好的方法.例如,本文重构低层场景时需要的操作系统层事件记录也可由取证专用的事件监控系统(如 Forensix[2])提供.这类机制能够较为有效地保证数据来源的可信度和可靠性.

目前,对于证据分析工具的可信度和可靠性并无统一的量化评估标准,因此我们仅从定性的角度分析了本文方法在保证结果可信度和可靠性方面采取的措施,其作为证据分析工具的能力还有待专家的进一步评估.

7 结 论

本文提出了一种自动证据分析技术——层次化入侵场景重构.它能够根据原始证据自动重构出入侵者的行为过程,而且重构结果较为准确、完备、易懂.基于 DARPA 2000 的实验已经证明了本文方法的有效性,而与现有方法的比较则显示它能够处理未知行为,适合分析复杂入侵,安全性也有所增强.未来我们将关注的是多个主机的联合场景重构.另外,如何利用重构出的场景自动筛选入侵证据也值得研究.

References:

[1] Ding LP, Wang YJ. Study on relevant law and technology issues about computer forensics. Journal of Software, 2005,16(2): 260−275 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/260.htm [doi: 10.1360/jos160260]

[2] Goel A, Feng WC, Maier D, Feng W, Walpole J. Forensix: A robust, high-performance reconstruction system. In: Proc. of the ICDCS 2005. Washington: IEEE Press, 2005. 155−162. [doi: 10.1109/ICDCSW.2005.62]

[3] King ST, Chen PM. Backtracking intrusions. ACM Trans. on Computer Systems, 2005,23(1):51−76. [doi: 10.1145/945445.945467]

[4] Duval T, Jouga B, Roger L. The Mitnick case: How Bayes could have helped. In: Proc. of the IFIP Int'l Conf. on Digital Forensics. Heidelberg: Springer-Verlag, 2005. 91−104. [doi: 10.1007/0-387-31163-7_8]

[5] Bradley S, Mohay G, Clark A. Generalising event forensics across multiple domains. 2004. http://eprints.qut.edu.au/9987/1/9987.pdf

[6] Carney M, Rogers M. The Trojan made me do it: A first step in statistical based computer forensics event reconstruction. Int'l Journal of Digital Evidence, 2004,2(4):1−11.

[7] Wang W, Daniels TE. Building evidence graphs for network forensics analysis. In: Proc. of the ACSAC 2005. Washington: IEEE Press, 2005. 254−266. [doi: 10.1109/CSAC.2005.14]

[8] Ponec M, Giura P, Brönnimann H, Wein J. Highly efficient techniques for network forensics. In: Proc. of the ACM CCS 2007. New York: ACM Press, 2007. 150−160. [doi: 10.1145/1315245.1315265]

[9] Aggarwal S, Burmester M, Henry P, Kermes L, Mulholland J. Anti-Cyberstalking: The predator and prey alert (PAPA) system. In: Proc. of the SADFE 2005. Washington: IEEE Press, 2005. 195−205. [doi: 10.1109/SADFE.2005.2]

[10] Gladyshev P, PATEL A. Finite state machine analysis of a blackmail investigation. Int'l Journal of Digital Evidence, 2005,4(1): 1−13.

[11] Pietraszek T. Using adaptive alert classification to reduce false positives in intrusion detection. In: Proc. of the RAID 2004. Heidelberg: Springer-Verlag, 2004. 102−124.

© 中国科学院软件研究所 http://www.jos.org.cn

[12] Christopher K, Fredrik V, Giovanni V. Intrusion Detection and Correlation: Challenges and Solutions. Heidelberg: Springer-Verlag, 2005.

[13] Ning P, Cui Y, Reeves DS. Constructing attack scenarios through correlation of intrusion alerts. In: Proc. of the ACM CCS 2002. New York: ACM Press, 2002. 245−254. [doi: 10.1145/586110.586144]

[14] Zhai Y, Ning P, Xu J. Integrating IDS alert correlation and OS-level dependency tracking. In: Proc. of the ISI 2006. Heidelberg: Springer-Verlag, 2006. 272−284. [doi: 10.1007/11760146_24]

附中文参考文献:

[1] 丁丽萍,王永吉.计算机取证的相关法律技术问题研究.软件学报,2005,16(2):260−275. http://www.jos.org.cn/1000-9825/16/260.htm [doi: 10.1360/jos160260]

伏晓(1979－),女,江苏徐州人,博士,讲师,主要研究领域为网络安全,机器学习.

石进(1976－),男,博士,讲师,主要研究领域为保密技术,信息安全.

谢立(1942－),男,教授,博士生导师,主要研究领域为分布式计算,信息安全.

© 中国科学院软件研究所 http://www.jos.org.cn