基于扩展有限状态机的诱骗服务器关键技术研究*)

陈云芳[1] 王汝传[1,2] 杨学刚[1]

(南京邮电学院计算机科学与技术系 南京210003)[1]

(中国科学院研究生院信息安全国家重点实验室 北京100039)[2]

摘 要 作为防火墙和入侵检测的有效补充,诱骗服务器成为网络安全的越来越重要的部分,本文在对当前诱骗服务器的研究水平进行了仔细分析的基础之上,提出了使用有限状态机理论来构建诱骗服务器的方案,并对其中的关键技术做了详细阐述。

关键词 诱骗服务器,有限状态机,扩展有限状态机

The Study of Network Trap Based on Extended Finite State Machine

CHEN Yun-Fang[1] WANG Ru-Chuan[1,2] YANG Xue-Gang[1]

(Department of Computer Science and Technology, Nanjing University of Post and Telecommunications, Nanjing 210003)[1]

Abstract As the effective supply of the firewall and IDS, the honeypot become more and more important aspect of network security. Based on our detail research of current honeypot development, the paper gives a scenario of building a practical honeypot by using extended finite state machine. And we also describe the crisis technology that our system used.

Keywords Honeypot, Finite state machine, Extended finite state machine

1 引言

随着信息化网络的发展和不断普及,网络攻击事件不断增加,网络安全日益成为人们关注的焦点。传统的主要技术手段如防火墙、安全路由器等只是起到防御功能,而系统一旦被攻破,就完全处于被动了。入侵检测系统作为目前最流行的主动网络安全防护技术,它能发现外部攻击和合法用户滥用特权,是动态安全技术中最为核心的技术之一。然而,随着网络服务的不断增加,系统所面临的危险也成倍增长。大量的攻击工具被写成了图形界面工具和内核级的 rootkit,网络攻击越来越智能化和傻瓜化。

因此,一个完整的网络安全解决方案,不仅应该有防火墙等防御手段,有能够对网络进行实时监控发现入侵行为的检测系统,还应该能承受住一定程度上的网络攻击并尽量获取攻击者的信息。诱骗服务器就是在这种情况下诞生的,笔者认为,防火墙、入侵检测系统和诱骗服务器必将成为计算机网络安全的三大支柱。

诱骗服务器通常是一个没有经过安全加固的操作系统或者是有意设计成有缺陷的系统,这个服务器和网络中的其它服务器没有什么大的区别,但在系统中安装了很多虚假的文件路径和其它一些有价值的信息,使得黑客相信当他们入侵系统后能获得重要信息。

诱骗服务器主要有两个目的,其一是拖延攻击者对其真正目标的攻击,让攻击者在蜜罐上浪费时间。其二是为起诉恶意黑客搜集证据,这看起来有"诱捕"的感觉。通过诱骗服务器我们能更多地了解他们所使用的攻击技术,以便更好地保护我们的网络。

2 扩展有限状态机

定义1(有限状态计算机) 有限状态计算机之所以被称为"有限"是指在两种方式下有限:

(1)它只有有限数目的状态;

(2)它的时间过程是"离散"的。

目前的数字计算机从根本上来说都是有限状态计算机。状态是计算机的基本数据特征。时间虽然是连续流动的,但是计算机在运行时是离散的,因为控制计算机的钟是数字的。我们可以使用有限状态机理论来进行计算机的状态分析与模拟。

1992年 Stanford 大学 Dill 等人开发了有限状态验证系统 Murphi,验证了存储器的相关协议,其后又成功分析了 TMN 协议、Kerberos 协议和 SSL 协议。由此可见,利用有限状态机来分析协议是完全可行的。目前的网络服务绝大多数架设在 TCP/IP 协议之上,网络协议基本都是点对点的协议。TCP/IP 协议本身就是使用有限状态机来描述其状态迁移的,某个具体的网络协议,特别是一些通过字符终端接入服务的网络协议,它的状态更加有限。服务器/客户端都可以看作是一个有限状态机。所以从理论上来讲,我们设计出自己的有限状态机解释器来代替系统原有的某个具体的网络服务,处理远程用户的请求是完全可行的。精心设计我们的有限状态机处理器,接管所有的用户请求就能够达到对用户请求信息的完全控制。

为了达到诱骗服务器的欺骗效果,使得网络攻击者把我们所模拟的网络服务当作系统原有的正常网络服务,我们要使用有限状态机来模拟已有的网络协议和具体服务程序的数

*)本课题得到国家科学基金(60173037)、江苏(2003105)、国家高科技项目八六三(2002AA776032)、江苏省计算机信息处理技术重点实验室基金(kjs03061)和中兴通讯研究基金资助。陈云芳 博士生,主要研究方向是计算机软件、计算机网络、信息安全、移动代理等;王汝传 教授,博士生导师,主要研究方向是计算机软件、计算机网络、信息安全、移动代理和虚拟现实技术等;杨学刚 硕士研究生,主要研究方向为计算机在通信中的应用。

据处理过程。基于有限状态机进行网络协议的模拟，针对攻击者最常用的攻击行为，通过状态穷举的方法来模拟攻击者所有可能达到的协议状态。我们所以要求网络协议的状态有限，是为了保证对状态空间的遍历可以结束。对应某种针对网络协议的攻击行为，可能的状态转移也许有多个，所以利用有限状态机来分析网络协议一般采用扩展的有限状态机模型。

定义2(EFSM 模型)　扩展有限状态机(Extended Finite State Machine)可以形式化地表示为一个六元组：

$\langle S, S_0, I, O, T, V \rangle$

其中：S 表示一个非空的有限状态集合；S_0表示初始状态；I 表示非空的输入交互原语集合；O 表示非空的输出交互原语集合；T 表示非空的变迁集合；V 表示变量集合。

T 的每个元素又是一个五元组，

$T = \langle Src\text{-}State, Dest\text{-}State, Input, Predicate, Compute\text{-}Block \rangle$。

其中：*Src-State* 表示的是 T 的首状态；*Dest-State* 表示的是 T 的末状态；*Input* 表示来自于 I 的输入原语或空；*Predicate* 是关于 V 中变量、输入原语输入的参数和某些约束的类 *Pascal* 的谓词表达式；*Compute-Block*：指的是包括类 *Pascal* 的赋值语句和输出语句的计算模块。

EFSM 相对于传统的 FSM 增加了额外的变量用于描述系统中更为复杂的状态，操作可能在改变一般的系统环境变量的同时，还改变变量，这样可以控制有限状态机的状态爆炸问题，大大增加状态机的描述能力。

在以下的描述中，我们假设网络协议所描述的 EFSM 表示是确定性的，而且是完全描述的；初始状态在给定的有效的上下文环境中，EFSM 模型是强连通图。

3　系统关键技术

3.1　系统结构

我们通过 Linux 提供的 TCP Wrapper 服务截获用户的网络服务请求，将用户请求交由我们的诱骗服务程序处理，经过我们的诱骗服务器的扩展有限状态机分析处理之后，返回我们所定制的简单屏幕信息给用户，使得用户看到的永远是错误的返回信息，而无法得到我们系统的真实信息，从而产生疑惑，浪费更多时间在我们的诱骗服务器上；同时，我们的诱骗服务器程序对客户的服务请求记录到日志当中，获取入侵的证据。

针对不同的网络服务，我们只需要设计对应的服务的有限状态变迁集合就可以实现多种服务的诱骗。通常，攻击者在攻击成功之前最需要的就是客户连接，我们针对黑客最需要平凡使用的 telnet 协议为例设计我们的基于有限状态机的诱骗服务器。我们使用的平台是 Linux red hat 7.2，系统的体系结构如图1所示。

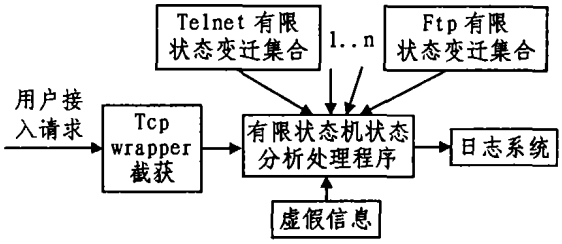

图1　系统体系结构

3.2　截获系统网络服务

在系统运行之前，我们要进行相关的设置工作来保证用户的接入请求被我们的诱骗服务器程序接管。Linux 中的大多使用 TCP Wrapper 来监测和过滤 finger，ftp，telnet，rlogon，exec，tftp 等等具体网络服务的客户请求。一旦用户连接请求通过 Linux 的简单防火墙 ipchains、系统网络服务端口 service 检查之后，就将面对 TCP Wrapper。TCP Wrapper 是为了提高了系统的安全性和系统管理的方便性而设立的，它将来自客户端的请求同一个独立的守护进程 xinetd(eXtended InterNET services daemon)直接通信，而其它请求的目标服务都被 TCP Wrapper 包裹起来。通过定制\etc\xinetd.conf 配置文件，我们就可以把客户请求发送给我们的处理程序。在分发具体服务之前，Linux 还使用 hosts.allow、hosts.deny 来进行主机的接入检查。一个网络服务的大致流程如图2所示。

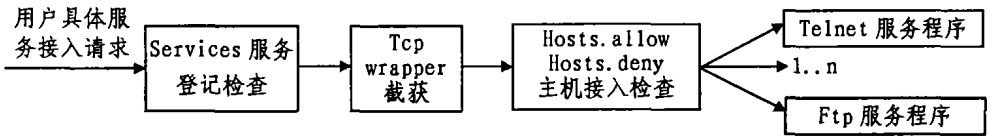

图2　系统网络服务截获

从以上的说明中我们可知，有两种方法可以截获网络服务：

第一，在\etc\xinetd\目录下修改我们需要截获的系统网络服务文件，使得其服务处理程序直接指向我们的诱骗服务器程序，使用 server 关键字指向我们的诱骗服务程序。

第二种方法不修改在\etc\xinetd\目录下系统网络服务文件，而是在 hosts.allow 中使用 twist 关键字使得服务处理程序间接指向我们的诱骗服务器程序。

使用 Linux 已有的技术，我们不但可以很方便地把系统网络服务重新定向，而且可以使得我们的某个具体的网络服务诱骗服务的设计独立性得到提高。我们甚至可以针对特定的服务单独开发我们所需要的特定的诱骗服务程序。

3.3　有限状态的变迁集合的结构设计

关于 telnet 网络协议的 EFSM 表示我们不再进行验证，针对 EFSM 中的非空的变迁集合 T，我们设计的状态变迁集合如表1所示。

表1　状态变迁集合

我们使用的状态变迁集合	对应与 EFSM 中变迁集合中的元素
State	*Src-State*
Input	*Input*
Next	*Dest-State*
Parm	*Predicate*
Cmd	*Compute-Block*
Message	/

其中，*State* 和 *Next* 都属于 telnet 协议的状态集合，在这里，我们使用简单的整型数来表示该服务连接之后所处的状态。*Input* 表示用户请求的请求信息，对应于 telnet 服务来说，就是用户的输入请求信息的字符串。*Parm* 表示一些输入原语的参数或者是某些约束条件。*Cmd* 表示输出语句的命令模块，对我们要传递给连接请求用户的字符串的输出方式加以处理。*Message* 是我们增加的一项，表示的是要返回给请求连

接用户的信息。

下面我们给出 telnet 服务的状态变迁集合的部分片断，如表2所示。

表2 telnet 服务状态变迁集合

```
#port 23 (telnet) finite state machine translation set table
# State    Input    Next  Parm  Cmd    Message
0          START    1     1     cat    @23.banner
0          NIL      0     1     cat    @23.banner
0          ERROR    0     1     cat    @23.banner
# initial prompt
1          NIL      2     1     cat    @23.login
1          ERROR    2     1     cat    @23.login
# user IDs
2          fc       3     1     echo   Password:
2          guest    3     1     echo   Password:
2          root     3     1     echo   Password:
2          daemon   3     1     echo   Password:
# passwords
3          root     4     1     echo   HOST_NAME $
3          guest    4     1     echo   HOST_NAME $
3          fc       4     1     echo   HOST_NAME $
3          daemon   4     1     echo   HOST_NAME $
# some commands
4          ls       4     1     cat    @23.ls
4          df       4     1     cat    @23.df
4          pwd      4     1     cat    @23.pwd
```

从以上的片断中，我们可以清晰地看出有限状态机的变迁集合的工作机制，以一个用户 telnet 登录为例，假如他连接上之后依次输入的是 telnet ×.×.×.×(登录)，guest(用户名)，guest(密码)，ls(命令)，他将依次看到的返回信息是文件@23.login 中的信息、字符串"Password:"、全局变量 HOST_NAME $ 的值，文件@23.ls 中的信息。

4 增强诱骗服务器的欺骗质量

我们的系统能实施有效的欺骗，但是欺骗质量并不高，攻击者经过多次试探之后很可能就能发现系统是个诱骗服务器。我们可以采取以下的手段来提高系统的欺骗质量：

(1)状态爆炸问题的解决。利用有限状态机分析和模拟协议所面临的主要问题是状态爆炸。为了控制状态爆炸，应寻找一定的方法减少状态的生成。使用人工智能从当前网络中的网络服务中的协议状态中学习，总结出最常使用的状态能有效减少状态数。

(2)模拟网络数据流，使得一些网络数据流分析系统不能发现欺骗。一种办法是获取当前内部网络的真实网络数据流进行简单的重放，第二种办法是获取外部网络与内部网络交换的数据流进行重放。网络数据流的重放能有效地迷惑攻击者的监听行为。

(3)动态配置欺骗服务器。真实的网络总是在不断变化中，诱骗服务器应当尽可能地反应出真实系统的一些变化，一成不变的诱骗服务器很容易被攻击者发现。我们可以使用一定的算法来随机产生返回给客户的信息。

(4)建立一个欺骗性的网络。单一的诱骗服务器不能有效地实施诱骗服务器的诱骗功能，如果攻击者没有发现诱骗服务器，那么诱骗服务器就失去了它的价值。我们可以在系统中设置多台诱骗服务器，并且在网络的 DNS 中增加相应的条目，提高诱骗服务器的主动诱骗功能。

结束语 诱骗服务器是网络安全的热点课题，本文详细分析了基于有限状态机的诱骗服务器的理论可行性，并就在 Linux 平台上实际架构诱骗服务器所必须处理的关键技术做了详细的阐述。当然，目前的诱骗服务器缺少动态的变化，自身的隐藏能力差，主动诱骗能力缺乏等等都影响其进入实用阶段，这些也是我们未来研究的重点内容。

参 考 文 献

1 Dill D L, The Murphi Verification System. In: Int'l Conf. Computer Aided Verification, 1996. 390～393
2 Klug D. HoneyPot and Intrusion Detection. Http://www.sans.org/infosecFAQ/Honeypots.html
3 RFC(Request For Comments)－764, Telnet Protocol specification
4 王建国，吴建平. 基于扩展有限状态机的协议测试集生成研究. 软件学报，2001，12(8)：1197～1204
5 夏春和，吴震，等. 入侵诱骗模型的研究与建立. 计算机应用研究，2002，4：76～79
6 刘湘辉，等. 利用有限状态机分析 TCP 协议握手过程的安全问题. 计算机工程与科学，2002，24(4)：21～23

(上接第78页)

```
{
[property,int] CacheSize;
// 从缓冲区中读取所需要的对象，并返回未查找到的对象 OID 列表
[method,long] ReadObjects([in,out,int] *pNum, [in,OID,size_is(*pNum)] *pOIDList, [in,boolean] bSpatial, [in,string] TBLName, [out] void **ppResult,[out,int] pNum2, [out,OID,size_is(*pNum2)] *pOIDList2 );
// 向缓冲区中写入对象
[method] WriteObjects ([in,out,int] pNum, [in,OID,size_is(pNum)] *pOIDList, [in,boolean] bSpatial, [in,string] TBLName, [in,out,long] *pLen, [in] void *pValues);
// 更新缓冲区中的对象
[method] UpdateObjects([in,out,int] *pNum, [in,OID,size_is(*pNum)] *OIDList, [in,boolean] bSpatial, [in,string]TBLName, [in,out,long] *pLen, [in] void *pValues);
// 删除缓冲区中的对象
[method,long] DeleteObjects([i,out,int] pNum, [in,OID,size_is(*pNum)] *OIDList,[in,string] TBLName);
};
```

总结 高速缓冲技术作为一种能够提高系统性能的有效技术广泛的应用于各种系统中。在分布式空间数据库中，基于重叠区域的客户端高速缓冲技术能够减少不必要的网络数据传输，提高分布式空间数据库的性能和可伸缩性。本文给出了客户端高速缓冲技术在分布式空间数据库中应用的系统结构，提出符合空间数据特点的高速缓冲置换策略，采取这种置换策略比单纯采用 LRU 策略能够获得更好的性能，最后讨论了基于事务的缓冲一致性问题。

参 考 文 献

1 Franklin M J, Carey M J, Livny. Transactional Client-Server Cache: Alternatives and Performance. ACM, 1997, 0362-5915/97/0900-0315
2 Güting R H. An Introduction to Spatial Database Systems. In processdings: VLDB Journal, 1994, 3(4)
3 Brinkhoff T. A Robust and Self-Tuning Page-Replacement Strategy for Spatial Database Systems. In: Proc. 8th Intl. Conf. on Extending Database Technology, Prague, Czech Republic, 2002
4 Beckmann N, et al. The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. In: Proc. ACM SIGMOD Intl. Conf. on Management of Data, Atlantic City, NJ, 1990. 322～331
5 Guttman A. R-trees: A Dynamic Index structure for Spatial Searching. In: Proc. ACM SIGMOD Intl. Conf. on Management of Data, Boston, 1984. 47～57