网络集成防护系统的分层事件驱动风格*)

陈 波 卢显良 韩 宏 任立勇
(电子科技大学计算机学院 成都 610054)

摘 要 网络集成防护系统是有效防御网络攻击的一种手段,其面临的最大问题之一是如何有效集成响应和处理系统,并在可扩展性和快速响应之间取得折衷。本文吸收了分层框架和消息驱动框架的优点,提出并实现了一种分层事件驱动框架风格,它在提供了较高的灵活性的同时也强调了足够的系统响应速度。该框架风格的优势在我们实现的集成防护系统中得到了验证。

关键词 框架风格,分层事件驱动风格,基于消息的驱动,分层构架

A Layered Event-driven Architecture Style of Network Integrated Protection System

CHEN Bo LU Xian-Liang HAN Hong REN Li-Yong
(Department of Computer Science, UESTC, Chengdu 610054)

Abstract Network Integrated Protection System is an efficient security facility. It is confronted with a greatest problem which is how to integrate response and handling systems, while balancing the extensibility and speed of response. This paper combines the advantages of both layered architecture and message-driven architecture to present a new architecture style, Layered Event-driven Architecture. It gets a tradeoff between the flexibility and response speed. It is implemented in our network integrated protection system.

Keywords Architecture style, Layered message-driven architecture style, Message-driven, Layered architecture

1 引言

网络集成防护系统必须处理各种扩展性问题。而安全事件的分析和处理是该系统面临的基本事务,例如存储告警,分析告警,动态配置防火墙,对告警进行数据融合等等,并且响应措施也是多样化的。如何构建系统以达成良好的扩展性,集成不同类型的响应行为,并优化提高响应速度就成为系统设计面临的重要难题。

Mary Shaw 指出框架由组件(component)和连接器(connector)构成[1]。在防御框架中,组件为防御功能模块,连接器为防御功能管理子框架、表示服务子框架、控制服务子框架等七个子框架[2]。以下因素要求框架支持分层式事件驱动风格:

- 逻辑层、表示层、数据层和通讯层分离使系统具有良好模块化特性;
- 消息传递过程中,系统呈现明显的分层特性;
- 该系统为交互系统,事件驱动提供更好的互动能力;
- 分层系统需要支持跨层互动性;
- 分层和事件驱动混合使用在某些情况下能提高实时性;
- 事件驱动和隐式调用方法,使系统具有良好扩展性。

基于以上因素,我们提出了分层事件驱动风格。

2 分层事件驱动框架风格

2.1 分层系统及其交互性支持

交互系统对传统严格分层模型提出了挑战,本节给出了跨层交互性的解决办法。框架从逻辑上分为表示、数据、控制和通讯层。通讯层自身也是典型分层模式,包括认证、加密和具体通讯三层。通讯层完全隐藏了认证、加密等信息,如图 1 所示。

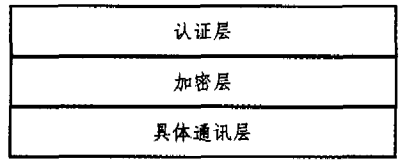

图 1 采用单纯分层的通讯层

分层使认证加密以及通讯措施对上层使用者透明。但这一透明性在表示层引发了问题:表示层位于通讯层上方,不知道认证层存在,但认证层的相关信息却需要在表示层显示。例如,需显示认证失败信息。如果令表示层直接耦合认证层,又违反了分层目标。

本文采用事件驱动方式在层上增加观察点,通过类似观察者模式[3],观察点上能插入多个观察者。这些观察者可来自其它层(本例中来自表示层)。观察者在分层机制中引入了极大灵活性,通讯相关处理逻辑可轻易扩展。例如:可将认证失败加入黑名单,通讯失败数据自动重发等功能实现为观察者,并将其插入观察点。事件驱动优点是:认证层不直接了解某种特定类型观察者,而观察者也并不知道认证层的接口(只了解相关事件),这种弱耦合使系统扩展较为容易。分层混合事件驱动的通讯层模型如图 2 所示。

*)本文受信息产业部生产发展基金项目网络安全集成防护系统支持,项目代号信部运[2002]546。陈 波 博士,主要研究方向:网络安全、软件工程。卢显良 教授,博士生导师,主要研究方向:计算机网络、操作系统。韩 宏 博士,主要研究方向:网络安全。

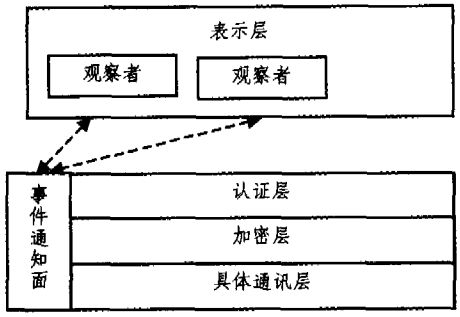

图 2　分层事件通知混合型通讯层模型

图中事件通知面和通讯各层产生交截。其实现可通过两种方法：1. 事件通知面在各层插入被通知点；2. 用异常机制拦截穿越各层的事件。采用方案 1 的伪代码(对象 pascal 语法)如下所示：

```
TLayer=class
Public
    Procedure AddObserver(Observer: TObserver);virtual;
    //override by subclasses
End;
TObserver = class
Public
    Procedure ProcessEvent(Event: TEvent);virtual;
    //override by subclasses
End;
TCommLayer = class(TLayer)
Public
    Procedure AddObserver(Observer: TObserver);override//
    Event notification plan could add using it.
End;
TAuthenticationLayer = class(TLayer)
Public
    Procedure AddObserver(Observer: TObserver);override//
    Event notification plan could add using it.
End;
TEventNotificationPlan = class(TObserver)
Public
    Procedure AddObserver(Observer: TObserver);//observer
    of other modules could add using it, like observer of presen-
    tation layer.
    Procedure ProcessEvent(Event: TEvent);override; //called
    by layers
End;
```

该系统中除了通讯部分呈现出分层与交互两种特性外，分析层和决策层亦然。因此，集成防御框架需要支持两种方式的混合风格。

2.2　复合分层与事件驱动提高实时性

系统须支持高实时性。传统分层模型在实时性方面存在问题。来自底层的事件必然要经过 N 层后才能在相应位置得以响应；反之响应亦需通过 N 层传递。问题详细可参见文[1]。

在分层系统中消息传递需经过以下层，如图 3 所示。

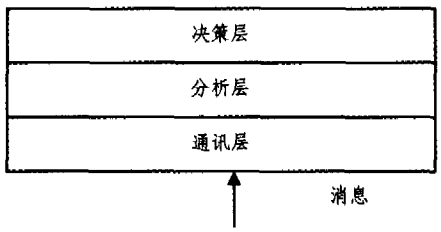

图 3　分层结构消息的响应过程

分析层通过不同算法对消息综合分析，决策层根据分析结果和预设策略动态选择响应方式。部分消息意义明确不需进一步分析，应立刻响应。因此该类消息应直接由决策层处理。但越层调用接口不仅违反分层目标，且使通讯层责任模糊，增加耦合度，破坏了系统维护性和扩展性。

解决方式如前，在层的基础上加入面的概念，面中筛选消息并通知决策层。该方式减少了消息响应时间。复合分层与事件驱动的结构如图 4 所示。

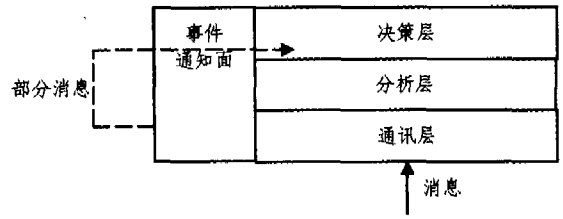

图 4　分层式事件驱动风格消息响应过程

2.3　分层式事件驱动风格

该风格由层、面和面观察者构成。

层：一方面支持标准分层模式中层的功能；另一方面支持事件通知，层内事件被通知到面。

面：和层交接，将层通知的事件过滤并通告面中的面观察者。

面观察者：注册到面中的事件响应实体。面观察者来自注册实体，该实体可属于任何层。通过面观察者，层实体间能形成信息前馈或反馈。

图 5 是三者间的关系。

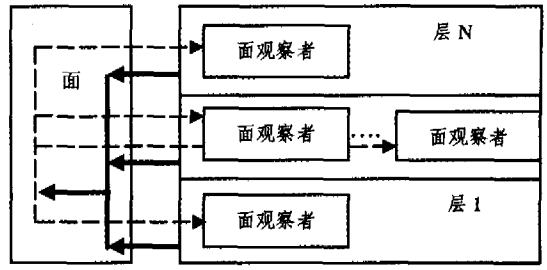

图 5　分层式事件驱动风格

图中粗实线为层流向面的消息，虚线是面通知面观察者的消息。从组件和连接器描述，该风格如下：(1)组件：层、面观察者。(2)连接器：过程调用(层与层间的连接器)和面。

2.4　消息驱动增加解耦度，提高系统扩展能力

交互式系统中较为有名的构架有 MVC(mode-view-controller)[4] 和 PAC (presentation-abstract-controll)[5]。MVC 构架源于 smalltalk(最早的面向对象语言之一，由贝尔实验室提出，属动态型语言)，并成功地运用于 smalltalk 编译器。MVC 衍生出多种变体：微软的 MFC[6] 中文档视图概念是裁剪后的 MVC；java 将 view 和 controller 合并形成 MVC 模式简约版并成为其 Swing 的框架[7]；ET ++[8] 也使用文档视图变体。另外从广泛意义上讲，一些设计思想也源自 MVC：比如微软的 DNA 构架，它将系统分为表示层、逻辑层和数据层，其中可找到 MVC 中对等实体：表示层对应视图，逻辑层对应控制器，数据层对应模型。

MVC 由多个设计模式构成，包括观察者模式、命令模式、工厂模式等[3]。其核心事件驱动方式由观察者模式支持，解决了模型和视图的解耦问题。其模型到视图的弱耦合方式令视图可独立于模型变化，完成灵活的用户接口设计。

事件驱动模型在编译器中也得到运用，例如 Field System[9]。该系统中编辑器、变量监测器注册所需调试器断点事件。当调试器停止在某个断点时，将通知注册实体。该方式使调试器在不知道注册者类型的情况下与其合作，降低了系

统耦合度，提高了系统重用性。

操作系统中，Windows 系统就是典型事件/消息驱动。其 GUI 完全靠消息驱动实现，一些重要机制亦然，如定时器、异步 I/O 读写以及内核不可抢占级别下与用户空间的通讯。消息机制大大提高了系统扩展性。

事件驱动方式能明显降低耦合度，提高扩展性。本框架因应用需求必须充分解耦，支持高扩展性。MVC 模式虽然给出方案，但不能完全满足本文需求，例如其核心模式——观察者模式中，观察者到模型是强耦合关系。通过基于消息的事件驱动，我们获得了更大的灵活性。

结论 分层事件驱动风格对不同的系统需求进行了支持，包括交互式应用，响应的实时性，系统可扩展性。在实际应用中，我们将不同的入侵分析系统，响应系统和日志系统通过该风格的框架集成到系统中。特别是事件驱动方式在分层构架中提供的足够的实时性，使得系统在速度和扩展性的选择方面获得了良好的灵活性。

参 考 文 献

1 Shaw M, Garlan D. Software architecture perspectives on emerging discipline. Prentice Hall, 1996

2 韩宏，卢显良. 插件式网络安全集成防护框架. 计算机科学，2005，32(4)

3 Gamma E, Helm R. Design Patterns Elements of Reusable Object-Oriented Software. Addison Wesley Longman, 1995

4 Krasner G E, Pope S T. A cookbook for using the Model-View-Controller user interface paradigm in smalltalk-80. Journal of Object-Oriented Programming, SIGS Publications, New York, NY, USA, 1988

5 Coutaz J. PAC, an Object Oriented Model for Dialog Design, Human-Computer Interaction. In: INTERACT'87 proceedings, 1987

6 Kruchten P B. The 4+1 View Model of Architecture. IEEE Software, Nov. 1995

7 Fowler A. A Swing Architecture Overview: The Inside Story on JFC Component Design. http://java.sun.com/products/jfc/tsc/articles/architecture/

8 Buschmann F, Meunier R. Pattern-Oriented Software Architecture Volum1: A System of Patterns. John Wiley & Sons, Inc. 1996

9 Reiss S P. Connecting tools using message passing in the Field Environment. IEEE Software, July 1990

(上接第 74 页)

的相关性能。这也是混沌序列能被广泛地应用到信息加密领域的一个重要原因。此处我们给出一个对比实验以示说明。仍然使用明文 msg1 和密钥 key1，混沌系统参数 $p=0.367$ 和 $p=0.366$(相差 10^{-3})，实验结果如图 4 所示。实验结果表明，参数 p 对密文的影响是很显著的。这说明如果将混沌系统参数作为密钥的一部分，将增大系统的安全性，提高抗破译的能力。设参数 p 的精度为 2^{-16}，则加密算法被攻破的概率为 $2^{-64}\cdot 2^{-16}=2^{-80}$。

结论 本文基于 Feistel 网络提出了一种新颖的反馈式分组混沌密码算法，并对算法进行了详细描述。与 DES 或 HOST 等算法不同的是，每个分组执行的轮数是根据前一分组的输出动态确定的，并且每一轮使用的 S-盒的序号也是由混沌映射动态产生的。将混沌引入到加密算法中来，由于混沌的固有特性，使得加密系统变得更加复杂，更加难以分析和预测，其抗攻击的能力大幅提高。实验结果也显示出本算法具有优良密码学特性，对明文和密钥以及混沌系统参数等非常敏感。此外，还对算法的安全性进行了分析，对于使用穷举法对系统进行攻击，我们认为其难度是相当大的。

文中的算法对更长的加密密钥以及其他混沌映射仍然适用。本文是我们在混沌分组密码研究工作的一个阶段性成果，后续的工作将对算法在抵抗差分密码分析和线性密码分析等方面做深入的研究，结果将另文报道。

参 考 文 献

1 Matthews R. On the derivation of a chaotic encryption algorithm. Cryptologia, 1989, XIII (1): 29～42

2 Habutsu T, Nishio Y, Sasase I, et al. A secret cryptosystem by iterating a chaotic map. In: Advance in cryptology - EUROCRYPT'91, LNCS 547 (Springer - Verlag, Berlin), 1991. 127～140

3 Biham E. Cryptanalysis of the chaotic-map cryptosystem suggested as EUROCRYPT'91. In: Advance in cryptology - EUROCRYPT'91, LNCS 547 (Springer - Verlag, Berlin), 1991. 532～534

4 Kocarev L, Jakimoski G. Logistic map as a block encryption algorithm. Phys Lett A, 2001, 289 (4～5): 199～206

5 Wong K W. A fast chaotic cryptographic scheme with dynamic look-up table. Phys Lett A, 2002, 298(4): 238～242

6 Murali K. Heterogeneous chaotic systems based cryptography. Phys Lett A, 2000, 272: 184～192

7 Schneier B. 应用密码学—协议、算法与 C 源程序. 吴世忠，祝世雄，张文政，等译. 北京：机械工业出版社，2000

8 Pareek N K, Patidar V, Sud K K. Discrete chaotic cryptography using external key. Phys Lett A, 2003, 309(1～2): 75～82

9 GOST R 34.11-94, Gosudarstvennyi Standard of Russian Federation. Information technology. Cryptographic Data Security. Hashing function. Government Committee of the Russia for Standards, 1994