计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2020,29(12):272−276 [doi: 10.15888/j.cnki.csa.007669]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于随机响应队列的 ActiveMQ 安全应用①

龚建华

(武汉科技大学 城市学院, 武汉 430083)
通讯作者: 龚建华, E-mail: gongjh123@163.com

摘　要: 随着消息中间件在大型分布式系统中的广泛应用, 消息中间件应用的安全性需要得到足够重视. 文中分析了 ActiveMQ 传统共享响应队列应用模式中存在的安全隐患, 提出了一种基于随机响应队列的消息中间件应用模式. 在该应用模式中, 响应队列名称由客户端随机生成, 只有该客户端和服务端知道这个随机名称, 因此这个随机队列很隐蔽且只能被该客户端独享, 从而确保消息队列应用的安全性. 文中给出了该应用模式的基本框架、操作流程和安全性分析, 并运用理论计算的方法分析了该模式的运行性能. 结论表明, 在保证安全性的同时, 该应用模式不会影响系统的运行性能.

关键词: 消息中间件; 请求队列; 响应队列; 安全性; 抓包

引用格式: 龚建华.基于随机响应队列的 ActiveMQ 安全应用.计算机系统应用,2020,29(12):272–276. http://www.c-s-a.org.cn/1003-3254/7669.html

Security Application of ActiveMQ Based on Random Response Queue

GONG Jian-Hua

(City College, Wuhan University of Science and Technology, Wuhan 430083, China)

Abstract: With the wide application of message-oriented middleware in large-scale distributed systems, its security should be paid more attention. This study analyzes the security risks in the traditional shared response queue application of ActiveMQ, and it proposes a message-oriented middleware application mode based on random response queue. In this application mode, the response queue name is randomly generated by user client. Only the client and server know the random name. The random queue is very hidden, so it can ensure security application of the message queue. The basic framework, operation flow and safety analysis of the application mode are given. The operation performance is analyzed by theoretical calculation. The conclusion shows that the security application mode does not affect the operation performance of the system.

Key words: message-oriented middleware; request queue; response queue; security; packet capturing

1　引言

消息中间件是大型系统中的重要组件, 它具有松耦合、异步消息、流量削峰、可靠投递、广播、流量控制、最终一致性等一系列功能, 已经成为异步 RPC 的主要手段之一. 目前常见的消息中间件有 ActiveMQ、RabbitMQ、Kafka、RocketMQ 等[1]. 消息中间件在大型系统和分布式系统中应用非常广泛[2–18], 因此消息中间件应用的安全性应该得到足够重视.

2　共享响应队列安全风险分析

面对分布式应用的信息安全问题, ActiveMQ 采取了一些必要的安全措施, 主要有两种安全访问策略[19],

① 基金项目: 全国高等院校计算机基础教育研究会计算机基础教育教学研究项目 (2020-AFCEC-342)
Foundation item: Computer Basic Education Teaching Research Project of Association of Fundamental Computing Education in Chinese Universities (2020-AFCEC-342)
收稿时间: 2020-04-06; 修改时间: 2020-04-28; 采用时间: 2020-05-10; csa 在线出版时间: 2020-11-30

一是简单认证策略, 二是 Java 认证与授权 (JAAS) 策略. 由于消息中间件在多用户应用中通常采用共享响应队列应用模式, 导致上述 ActiveMQ 安全措施仍然不能防止合规用户的越界访问和违规获取信息.

2.1 简单认证策略及其安全风险

简单认证策略中, 需要在 ActiveMQ 服务器的 activemq.xml 文件中配置允许访问消息队列的用户, 具体说就是在简单认证插件 (SimpleAuthenticationPlugin) 中配置用户名、密码和分组. 然后在客户端建立到消息队列连接时, 填写用户名和密码即可. 这种策略的本质是经过授权用户的可以使用消息队列, 类似于用户名/密码的登录机制.

这种策略的优点是使用起来比较方便简单, 但也存在安全风险, 合法用户对消息队列的操作没有限制, 可以管理、读、写任何消息队列. 在此策略基础上, 如果采取常用的共享请求队列 (即各客户端的请求发送到一个队列中) 或共享响应队列模式, 合法用户可以利用第三方工具软件从共享请求队列中或共享响应队列中非法读取他人请求信息.

2.2 Java 认证与授权策略及其风险

Java 认证与授权策略中, 也需要配置用户名、密码和分组, 并将这 3 项配置内容分散在 3 个文件中[19], 如在 login.config 文件中配置用户名密码管理文件和用户分组管理文件, 在 user.properties 文件中设置用户名和密码, group.properties 文件中设置分组和用户名.

与简单认证策略不同, JAAS 策略中增加授权机制, 即每一个队列只有指定的分组才可以管理或读或写, 对合法用户可以访问的消息队列进行限制, 比如只能向公共请求队列发送消息而不能读取消息, 从而保证发送到公共请求队列的消息不被合法用户违规读取. 在这种授权机制中, 上行方向单向应用消息队列是安全的, 即客户端向服务器端发送请求信息时采用共享请求队列可以保证安全性. 如果下行方向也采用分组共享响应队列, 不能保证其安全性.

Wireshark 是非常流行的网络封包分析软件, 可以抓取各种网络封包, 显示网络封包的详细信息.

图 1 是利用该软件分析得到的访问消息列队的用户名和密码, 利用该软件分析出得到的响应队列名称, 同样利用该软件还可以分析出消息中间件服务器的 IP 地址、端口号等, 有了这些分析信息, 合法用户可以通过第三方软件违规获取其他用户的响应信息.

由此可见, 虽然 JAAS 策略比简单认证策略有所改进, 但是下行方向采用常见的共享响应队列时仍然存在安全风险.

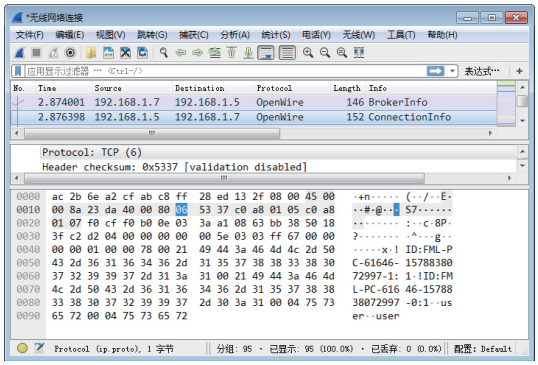

图 1 利用 Wireshark 抓包分析用户名和密码

3 基于随机响应队列的安全机制

3.1 系统基本架构

消息中间件应用中, 共享响应队列模式是影响系统安全性的关键因素, 为了提高系统安全性, 每个客户端应该独立使用一个响应队列. 在 ActiveMQ 中, 如果给每个客户端单独配置用户名、密码和响应队列, 当客户端很多时, 配置工作量将会非常大, 而且动态增减用户时, 必须更新配置文件并重启消息中间件服务器, 这会影响系统的连续运行.

本文提出随机响应队列应用模式, 既不用大量维护配置文件和重启服务器, 又可以使每个客户端隐蔽地单独占用一个响应消息队列, 系统架构如图 2 所示.

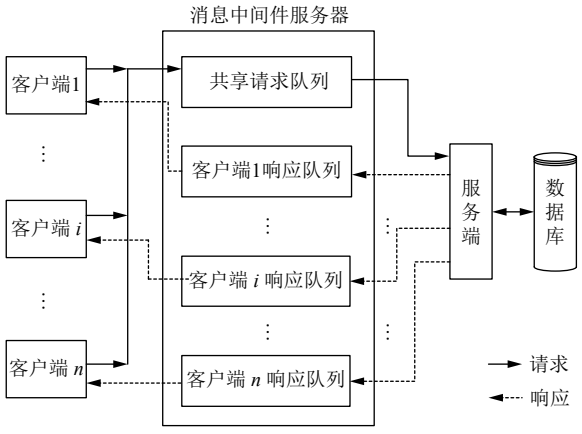

图 2 随机响应队列系统架构

在系统架构中, 消息中间件服务器中有 1 个共享请求队列和 n 个随机响应队列. 各个客户端对共享请求队列有写权限无读权限, 服务端对共享请求队列有读权限无写权限. 服务端对所有 n 个客户端响应队列有写权限无读权限, 各个客户端对所有 n 个客户端响应队列有管理和读权限无写权限.

在设置系统配置文件时, 给所有客户端配置共同的用户名和密码, 能够管理和读取某个固定前缀 (如 USER.response) 的响应队列. 为了保证客户端 i 响应队列只被客户端 i 读取, 而不被其他客户端读取, 将客户端 i 响应队列的名称设置为: “固定前缀+用户名+随机码”, 例如 USER.response.useri-xdWE2qUsz4, 加入“固定前缀”可以确保所有客户端都可以创建类似的队列, 加入“用户名”可以保证队列名不同重复, 加入“随机码”可以保证队列名不可预知从而保证队列名称的隐蔽性. 虽然各客户端理论上可以读取这个随机响应队列中的信息, 但是由于这个响应队列名称的随机性且只有创建它的客户端知道, 因而事实上这个响应队列只能被创建它的客户端使用, 从而保证安全性. 为了让服务端也知道这个随机响应队列的名称, 客户端给服务端发送请求时, 只需要附带这个随机响应队列名称即可.

为了完成上述任务, activemq.xml 文件需要增加如下配置[19]:

```
……
<authorizationEntry queue="USER.requeset"
        read="servers"
        write="users"
        admin="admins" />
<authorizationEntry queue="USER.response.>"
        read="users"
        write="servers"
        admin="users" />
……
```

3.2 系统运行过程

在随机响应队列系统架构中, 共享请求队列是固定的, 由消息中间件服务器创建. 客户端 i 发送请求和获得响应的完整闭环过程如下:

(1) 客户端 i 启动时, 动态创建客户端 i 响应队列, 如 USER.response.useri-xdWE2qUsz4.

(2) 客户端 i 通过共享请求队列发送请求, 请求信息除了业务信息外, 还需要附带客户 i 响应队列名称, 如, textMessage.setStringProperty("queueName", "USER.response.useri-xdWE2qUsz4"), 以便告知服务端响应信息发送到哪个队列.

(3) 服务端从请求队列接收消息, 解析 queueName, 如, textMessage.getStringProperty("queueName"), 并将业务响应结果发送到 USER.response.useri-xdWE2qUsz4 响应队列中.

(4) 客户端 i 监听 USER.response.useri-xdWE2qUsz4 响应队列, 当发现新消息时, 从响应队列中读取消息, 完成业务请求-响应闭环.

(5) 客户端 i 退出时, 删除随机响应队列 USER.response.useri-xdWE2qUsz4.

3.3 系统安全性分析

在随机响应队列系统架构下, 假定合规客户 i 想要非法获取其他客户端请求和响应信息, 在客户端 i 启动时, 在本地使用抓包软件进行分析, 他只能获取消息中间件服务器的 IP 地址、端口号、共享请求队列的名称、客户端 i 响应队列的名称等. 共享请求队列是只写的, 因此他不能通过读取请求队列的内容, 也无法通过此途径违规获取有关信息. 由于不能获得客户端 i 响应队列之外的其他队列的名称, 因此也无法访问其他响应队列, 无法违规获取其他客户有关信息. 虽然能够获取客户端 i 响应队列的名称, 也可以获取客户端 i 响应队列的内容, 但是此内容已在他自己的客户端中可以展现, 再用第三方软件抓取已经没有实际意义. 经上述分析, 在独立响应队列系统架构下, 可以保证消息队列中的信息安全.

消息中间件服务器的安全防护不是本文的研究内容, 用户非法直接攻击消息中间件服务器, 非法获取或篡改配置信息不在本文研究范围之内, 本文假定消息中间件服务器本身已经采取了防攻击措施.

4 性能分析

前面分析系统的架构、运行过程和安全性, 还需要进一步分析系统在点对多点分布式应用中的性能, 从定量角度分析探索哪些因素是制约系统运行性能的关键因素. 在客户端/服务器计算模式中, 系统响应时间是一项重要的指标, 可用经典排队论模型对系统性能进行分析.

系统时间模型如图 3 所示, 系统运行时间为客户端发送业务请求到收到请求响应所经历的时间, 设客

户端 i 的请求到达共享请求队列的传输时间为t_1, 该请求在共享请求队列中等待的时间为t_2, 该请求从共享请求队列传输到服务端的传输时间为t_3, 该请求在服务端业务处理时间为t_4, 响应从服务端传输到客户端 i 响应队列的时间为t_5, 响应在队列中等待时间为t_6, 响应从队列到达客户端的时间为t_7.

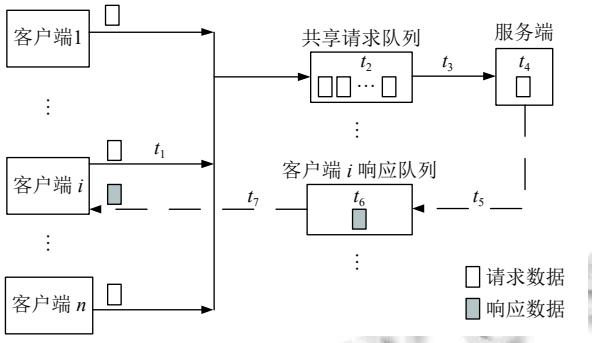

图 3 系统时间模型

(1) 设信息在系统中的传输时间$t_{传}$, 则$t_{传}=t_1+t_3+t_5+t_7$, $t_{传}$与系统网络硬件环境有关, 因此随机响应队列应用模式不会改变$t_{传}$的大小.

(2) 设信息在系统中的等待时间$t_{等}$, 则$t_{等}=t_2+t_4+t_6$, 由于客户端 i 响应队列为客户端 i 独享, 因此$t_6=0$, 最终$t_{等}=t_2+t_4$, 与系统中的客户端数量、单位时间内客户端发送的请求数量以及服务端的处理能力有关, 与客户端 i 响应队列无关.

综上所述, 系统运行时间主要受系统网络硬件环境、客户端数量、单位时间内客户端发送的请求数量以及服务端的处理能力有关, 采用随机响应队列模式不会影响系统运行时间或性能. 下面着重讨论为保证服务质量应当采取的相关措施.

假设客户端发送请求为简单流 (泊松流), 单位时间内, 客户端平均发送λ次请求, 则单位时间内发送次数服从分布为$X\sim P(\lambda)$[20].

对于 n 个相互独立的客户端, 设每个客户端单位时间平均发送λ_i次请求, 则到达业务请求队列的请求次数服从分布为:

$$X\sim P\left(\sum_{i=1}^{n}\lambda_i\right) \tag{1}$$

假设服务端完成业务处理的时间服从负指数分布, 且在单位时间内平均完成μ条业务处理, 则概率密度为:

$$f(t)=\begin{cases}\mu e^{-\mu t}, & t>0\\ 0, & 其它\end{cases} \tag{2}$$

假设消息队列长度不受限制, 请求响应服从 $M/M/1$[20] 排队系统模型, 则平均系统逗留时间为:

$$w_q=\frac{1}{\mu-\sum_{i=1}^{n}\lambda_i} \tag{3}$$

为简便起见, 设$\lambda_i=\lambda$, 则:

$$w_q=\frac{1}{\mu-n\lambda} \tag{4}$$

式 (4) 中, $n\lambda$表示总的请求强度, 只有$\mu>n\lambda$时, 业务请求队列的平均队长才是一个有限值, 请求逗留时间才是一个有限值, 否则平均队长会越来越大趋于无限, 使得等待时间也趋于无限, 最终请求应答不可完成.

根据条件的限制, 当服务端的业务处理能力一定时, 即μ一定时, 必须满足$n\lambda<\mu$, 也就是说, 如果n比较大, 那么λ必须足够小, 其表示物理意义是, 如果客户端数量较多, 那么每个客户端的请求强度不能太高; 反过来, 如果λ比较大, 那么n必须很小, 其表示的物理意义是, 如果每个客户端的请求强度比较大, 则客户端的数量不能太多.

当$\mu>>n\lambda$时, w_q比较小, 当$\mu\to n\lambda$时, w_q迅速增大, 假定用户对逗留容忍的值为$W_{\max}$, 用户的操作习惯决定了请求强度为λ, 服务器的性能决定了服务能力为μ, μ和λ通过样本均值统计得到, 那么系统可允许的客户端数量为$n_{\max}=\frac{\mu W_{\max}-1}{\lambda W_{\max}}$, 为了保证服务质量, 在线用户数达到$n_{\max}$服务端应该拒绝为新用户提供服务, 确保已在线用户可以得到满意的服务在质量.

5 结论

围绕消息中间件安全应用问题, 本文提出了基于随机响应队列的应用模式, 给出了系统架构和运行过程. 通过定性分析, 隐蔽的随机响应队列可以有效防范非法获取响应信息; 通过时间模型分析, 随机响应队列应用模式不会影响系统运行性能; 通过理论计算表明, 在请求强度和服务能力确定的情况下, 为保证服务质量, 在线用户规模应控制在$\frac{\mu W_{\max}-1}{\lambda W_{\max}}$以下.

参考文献

1 倪炜. 分布式消息中间件实践. 北京: 电子工业出版社, 2018. 1–274.

2 胡栋梁, 秦晓军, 王晓锋. 基于消息中间件的分布式网络扫

描研究. 计算机工程: 1–15. https://kns.cnki.net/kcms/detail/detail.aspx?doi=10.19678/j.issn.1000-3428.0056018. (2019-12-07).
3 樊鹏, 邱俊宏, 戚振伟, 等. 基于开源 ActiveMQ 的电力故障分析系统. 自动化与仪表, 2018, 33(3): 58–61. [doi: 10.19557/j.cnki.1001-9944.2018.03.014]
4 李洋. 基于消息队列遥测传输协议的智能家居消息中间件设计. 计算机应用, 2018, 38(S1): 162–164, 217.
5 徐进, 黄勃, 冯炯. 基于消息通信的分布式系统最终一致性平台. 计算机应用, 2017, 37(4): 1157–1163.
6 曹健, 刘琼, 王远. 基于数据流转发的实时数据交换系统设计. 计算机应用, 2016, 36(3): 596–600, 615.
7 张政, 侍守创. 基于消息中间件的制造执行系统的设计与实现. 计算机应用与软件, 2016, 33(10): 118–121.
8 刘尧, 宁芊. 基于消息中间件的信息系统数据传输与同步设计. 人民长江, 2016, 47(18): 106–109, 113. [doi: 10.16232/j.cnki.1001-4179.2016.18.023]
9 周聪. 基于改进的 ActiveMQ 的通信模型的设计和实现 [硕士学位论文]. 长春: 吉林大学, 2017.
10 范晓文. 基于消息中间件的机场生产运营系统的设计与实现 [硕士学位论文]. 北京: 北京邮电大学, 2017.
11 香华冠. 消息中间件在省邮政电子商务平台的应用探讨. 邮政研究, 2019, 35(4): 10–11. [doi: 10.13955/j.cnki.yzyj.2019.04.004]
12 雷卫延, 敖振浪, 刘艳中, 等. 基于模块化和 ActiveMQ 的通用数据采集软件设计. 电子测量技术, 2019, 42(12): 28–32. [doi: 10.19651/j.cnki.emt.1802533]
13 刘峰, 王昭鹏, 于波, 等. 基于消息中间件及 MongoDB 的物联网应用服务平台. 计算机系统应用, 2019, 28(5): 90–94. [doi: 10.15888/j.cnki.csa.006891]
14 王鹏, 从波, 李国杰, 等. 基于 ActiveMQ 消息总线的性能测试方法. 测试技术学报, 2019, 33(2): 147–152.
15 叶姣姣. 基于消息中间件技术的智慧园区解决方案. 电信科学, 2018, 34(S2): 185–191.
16 李松涛, 尹清爽. 消息中间件在物联网网关中的应用. 物联网技术, 2018, 8(12): 48–49, 54. [doi: 10.16667/j.issn.2095-1302.2018.12.012]
17 王重楠, 王宗陶, 鲍忠贵, 等. 发布/订阅模式测控消息中间件系统设计. 计算机应用, 2015, 35(3): 878–881.
18 吕德奎, 崔艳军. 开源消息中间件复杂并发连接控制的研究与实现. 网络安全技术与应用, 2016, (12): 66–67.
19 Snyder B, Bosanac D, Davies R. ActiveMQ in Action. London: Manning Publications, 2011. 117–142.
20 何选森. 随机过程与排队论. 长沙: 湖南大学出版社, 2010. 79–184.