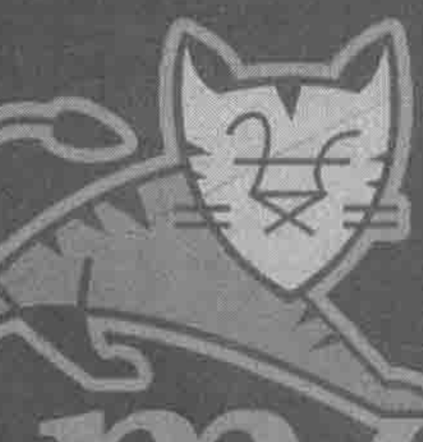
异步图书
www.epubit.com.cn

Tomcat内核设计剖析

汪 建◎著

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS



Tomcat内核设计剖析

汪 建◎著

人民邮电出版社
北 京

图书在版编目（CIP）数据

Tomcat内核设计剖析 / 汪建著. — 北京 : 人民邮电出版社, 2017.6
ISBN 978-7-115-45130-9

Ⅰ. ①T… Ⅱ. ①汪… Ⅲ. ①JAVA语言—程序设计
Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2017)第069990号

内 容 提 要

Tomcat 是一款免费的开源应用服务器，因其性能稳定、体积小巧、扩展性好等特点而被传统和互联网行业广泛应用。

本书是深入剖析 Tomcat Web 服务器运行机制的权威图书，共分为 22 章。本书从 Web 服务器相关的基础知识及原理开始逐渐深入 Tomcat 内部设计，比如涵盖了 HTTP 协议、Socket 通信及服务器模型等必备的基础知识。另外还包括 Servlet 规范，这些都是深入 Tomcat 必不可少的知识。然后介绍了 Tomcat 的启动与关闭过程，接着从整体预览 Tomcat 的内部结构，让读者对 Tomcat 内部有个整体的了解。最后开始层层剖析 Tomcat 内部结构，包括 Server 组件，Service 组件，内存泄漏检测，Connector 组件（HTTP 协议、AJP 协议、BIO 模式、NIO 模式和 APR 模式），Engine 容器，Host 容器，Context 容器，Wrapper 容器（Servlet 种类机制、Comet 模式、WebSocket 协议、异步 Servlet），生命周期管理，日志框架及其国际化（日志系统、日志国际化及访问日志），公共与隔离的加载器（多个 Web 应用如何做到资源隔离），Mapper 组件（局部路由、全局路由），Tomcat 集成 JNDI，JSP 编译器（JSP 语法解析、JSP 编译成 Servlet、Servlet 编译成 Class），运行及通信的安全管理，处理请求和响应的管道（管道机制），多样化的会话管理器（标准会话管理器、持久化会话管理器、集群增量会话管理器及集群备份管理器），高可用的 Tomcat 集群的实现（从单机到集群），Tomcat 集群通信框架，Tomcat 内部监控与管理。

本书适用于想深入了解 Web 服务器原理、想知道在浏览器上点击某个按钮后发生的事情、想了解 Tomcat 内部工作原理、想基于 Tomcat 做二次开发的人员。

◆ 著　　汪　建
责任编辑　傅道坤
责任印制　焦志炜
◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司印刷
◆ 开本：787×1092　1/16
印张：22.75
字数：487 千字　　2017 年 6 月第 1 版
印数：1 – 2 500 册　　2017 年 6 月河北第 1 次印刷

定价：79.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广字第 8052 号

作者简介

汪建，毕业于广东工业大学光信息科学与技术专业，毕业后从事航空系统、电信系统、中间件、基础架构、智能客服等研发工作，目前主要关注分布式、高并发、大数据、搜索引擎、机器学习等方面的技术。崇尚开源，崇尚技术自由，更崇尚思想自由。个人博客地址为blog.csdn.net/wangyangzhizhou。

致谢

首先，感谢读者，你的阅读让本书更加有价值。

其次，感谢在本书编写过程中帮助过我的人，感谢公司提供的平台让我得到了很多学习和成长的机会，还要感谢人民邮电出版社的傅道坤编辑，根据他的建议我对本书内容进行了多处改进，使内容更加丰富，结构更加清晰。

最后，感谢一直鼓励我、支持我的家人，特别是我的爱妻，挺着身孕仍然孜孜不倦地帮我审稿，你们让我的世界更丰富多彩。同时也将本书献给我即将出生的孩子。

前　言

Tomcat 作为一款免费的开源应用服务器，凭借技术先进、性能稳定、体积小巧、扩展性好等优势，深受开发者和软件开发商认可。鉴于 Tomcat 是一款较轻量级的应用服务器，它广泛使用在中小型系统中，并且是一个很流行的 Web 服务器。那么，如此优秀的 Tomcat 是怎样创造出来的呢？它的架构是怎样的呢？内部到底又是怎样运作的呢？需要哪些技术来支撑呢？有很多疑问都需要我们去研究和探索，作者试图在本书中阐明 Tomcat 内部的秘密。

虽然 Tomcat 已经广泛使用了很长时间，市面上也有很多相关图书，但多数关于 Tomcat 的图书基本都停留在如何使用 Tomcat、如何在 Tomcat 服务器上进行 Web 应用开发等方面。本书将从 Web 服务器基础知识开始讲起，循序渐进，让读者不仅能了解 Tomcat 内核的设计，还能掌握 Web 服务器的原理，体会到一个工业级的 Web 服务器是如何设计的。本书可以帮助读者快速建立 Tomcat 的内部运作模型。

重复发明轮子不是我们提倡的，本书并不鼓励读者重复开发轮子，而是鼓励大家去研究开源软件，学习其中的优秀架构，从中借签优秀的设计理念，看看这些优秀开源产品的过人之处，从而提高自己的软件素养。

本书具备如下特点。

- 所探讨的 Tomcat 基于 Tomcat 7 版本。
- 通篇大量采用图解，方便读者理解。
- 对各个设计要点都做深入剖析，读者可以体会到其中为什么要这样设计，原来工业级软件要考虑的如此多、如此细。
- 脉络结构比较清晰，由整体到部分，由浅到深，循序渐进，知识点的连贯性比较强，对于基础知识有补充说明，避免读者读到一半无法继续阅读。

组织结构

本书旨在剖析 Tomcat 的内核设计及其原理，全书共分为 22 章，主要内容如下。

- **第 1 章：Web 服务器机制**，介绍 Web 相关的基础知识，如 HTTP、套接字通信及服务器模型等。
- **第 2 章：Servlet 规范**，介绍 Java 体系 Web 容器的 Servlet 规范。
- **第 3 章：Tomcat 的启动与关闭**，介绍 Tomcat 启动、关闭的批处理及相关的变量。

- **第 4 章：从整体预览 Tomcat**，先从整体介绍 Tomcat 内部结构以及请求处理的整个过程，让读者能从整体了解 Tomcat 结构，为后面深入介绍各个组件做铺垫。
- **第 5 章：Server 组件与 Service 组件**，介绍 Server 和 Service 组件，以及 Tomcat 中对内存泄漏的监听检查。
- **第 6 章：Connector 组件**，介绍 Tomcat 包含的 HTTP 和 AJP 两种协议的连接器，以及它们不同的 I/O 模式，如 BIO 模式、NIO 模式和 APR 模式。
- **第 7 章：Engine 容器**，介绍 Engine 容器。
- **第 8 章：Host 容器**，介绍 Host 容器及其包含的内部组件。
- **第 9 章：Context 容器**，介绍 Context 容器及其包含的内部组件。
- **第 10 章：Wrapper 容器**，介绍 Wrapper 容器及 Servlet 的种类和工作机制，以及 Comet 模式的实现、WebSocket 协议的实现和异步 Servlet 的实现。
- **第 11 章：生命周期管理**，介绍 Tomcat 的生命周期管理机制及其事件监听机制。
- **第 12 章：日志框架及其国际化**，介绍 Tomcat 的日志系统及日志的国际化，同时还有 Tomcat 的访问日志的设计及使用介绍。
- **第 13 章：公共与隔离的类加载器**，介绍 Tomcat 内部的类加载器结构，如何达到多个 Web 应用既能共用某些类库又能互相隔离。
- **第 14 章：请求 URI 映射器 Mapper**，介绍 Tomcat 对请求 URI 处理的原理，以及局部路由和全局路由两种 Mapper。
- **第 15 章：Tomcat 的 JNDI**，介绍 Tomcat 内部对 JNDI 的集成支持，以及在 Tomcat 中如何使用 JNDI。
- **第 16 章：JSP 编译器 Jasper**，介绍 JSP 的语法及 Tomcat 如何对其进行解析，介绍从 JSP 到 Servlet，再从 Servlet 到 Class 的整个编译过程。
- **第 17 章：运行、通信及访问的安全管理**，介绍 Tomcat 内部运行时的安全管理，Tomcat 通信信道的安全实现，以及客户端访问认证机制。
- **第 18 章：处理请求和响应的管道**，介绍 Tomcat 中对请求和响应处理的管道模式的设计，以及在 Tomcat 中如何定制阀门。
- **第 19 章：多样化的会话管理器**，介绍 Tomcat 内部的会话管理机制，以及标准会话管理器、持久化会话管理器、集群增量会话管理器和集群备份会话管理器的实现机制及原理。
- **第 20 章：高可用的集群实现**，介绍 Tomcat 如何实现集群的高可用性，Tomcat 从单机模式到集群模式的会话管理，以及 Tomcat 的 Cluster 组件。
- **第 21 章：集群通信框架**，介绍 Tomcat 的集群通信框架 Tribes，剖析 Tribes 的原理机制，以及 Tomcat 如何使用 Tribes 进行会话同步和集群部署。
- **第 22 章：监控与管理**，介绍了 Tomcat 如何实现自身内部的监控及其管理。

读者对象

- 假如你对浏览器上单击某个按钮后发生的事情感兴趣，那么这本书适合你。
- 假如你想深入了解 Web 服务器原理，那么这本书适合你。
- 假如你想深入了解 Tomcat 核心架构的原理及 Tomcat 内组件的工作原理，那么这本书适合你。
- 假如你想设计开发一个类似 Tomcat 的中间件，那么这本书适合你。
- 假如你想基于 Tomcat 做二次开发，自定义 Tomcat，那么这本书适合你。

反馈

在本书交稿时，我仍在担心本书是否遗漏了某些知识点，其中的内容是否翔实齐备，是否能让读者有更多收获，是否会因为自己理解的偏差而误导读者。由于写作水平和写作时间所限，本书中难免存在谬误，恳请读者评判指正。

读者可将任何意见及建议发送到邮箱 wyzz8888@foxmail.com，本书相关的勘误也会发布到我的个人博客 blog.csdn.net/wangyangzhizhou 上。欢迎读者通过邮件或博客与我交流。

目　录

第1章　Web服务器机制

所有的 Web 服务器都根据规定好的协议机制进行不同的实现及扩展。有的 Web 服务器只能处理静态资源，而有的可以完成动态处理。有的 Web 服务器用 C++语言实现，而有的用 Java 语言实现。但不管 Web 服务器具体如何实现及扩展，它都必须要遵循基本的协议规定。在深入研究 Tomcat 之前很有必要先了解 Web 服务器的一些机制。

本章分别从通信协议、Socket 通信、Web 服务器模型三方面对 Web 服务器机制进行介绍。

1.1 通信协议

1.1.1 HTTP/HTTPS

HTTP 是 HyperText Transfer Protocol（超文本传输协议）的缩写。HTTP 协议是用于从 Web 服务器传输超文本到本地浏览器的协议，它能使浏览器更加高效，使网络传输减少，保证计算机正确快速地传输超文本文档。现在我们普遍使用的版本是 HTTP1.1。

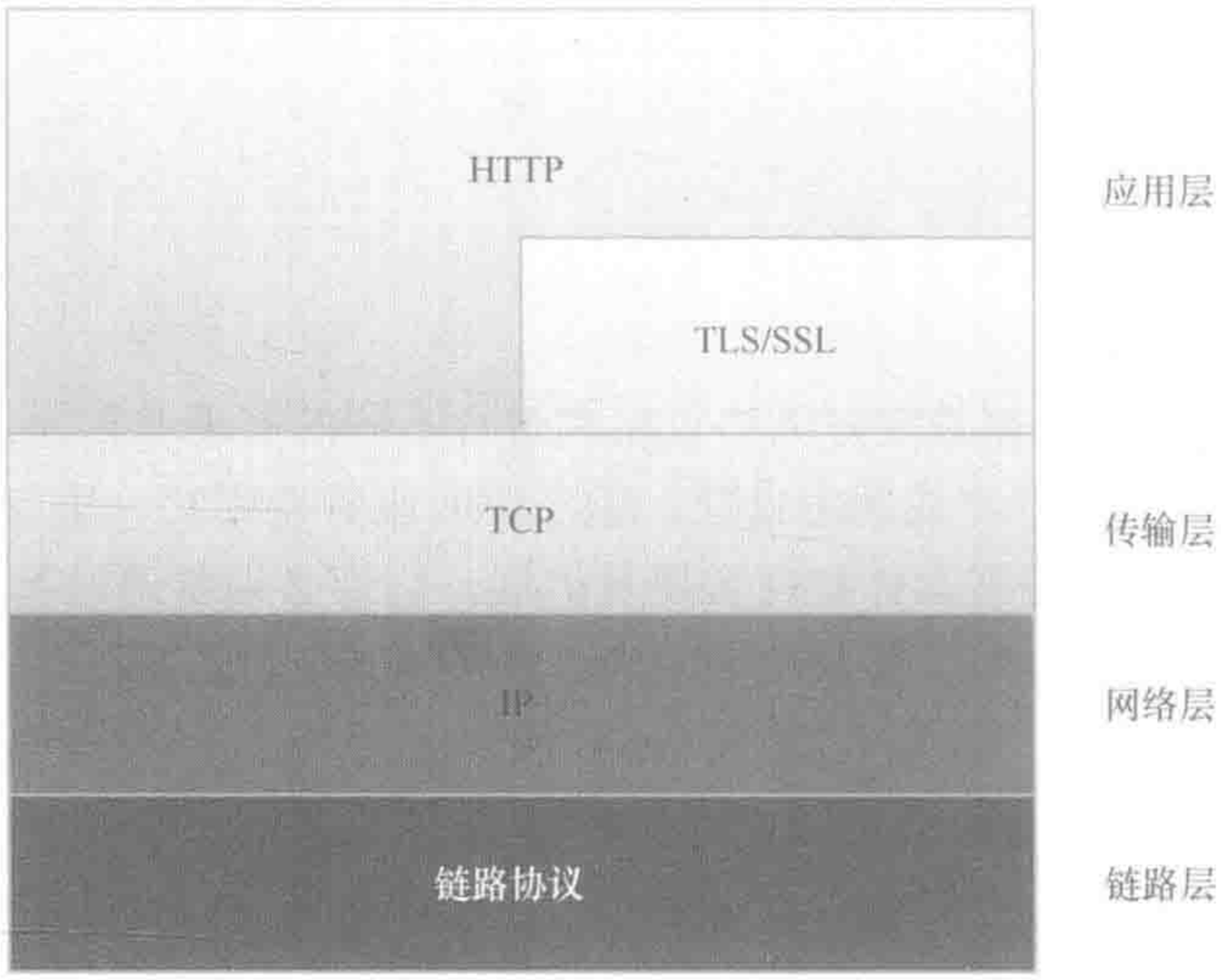
HTTP 是一个应用层协议，它由请求和响应组成，是一个标准的 B/S 模型。同时，它也是一个无状态的协议，即同一个客户端上，此次请求与上一次请求是没有对应关系的。

而 HTTPS 简单地说就是 HTTP 的安全版。通常，在安全性要求比较高的网站（例如银行网站）上会看到 HTTPS，它本质上也是 HTTP 协议，只是在 HTTP 增加了一个 SSL 或 TLS 协议层。如图 1.1 所示，如果在 TCP 协议上加一层 SSL 或 TLS 协议，就构成 HTTPS 协议了。SSL/TLS 协议提供了加解密的机制，所以它比 HTTP 明文传输更安全。从图 1.1 中可以看出，HTTP 可以直接进入 TCP 传输层，也可以在 TCP 层上加一层 SSL/TLS 层，这样就先经过 SSL/TLS 再进入 TCP 传输层。这两种方式便是 HTTP 与 HTTPS。一般 HTTP 的端口号为 80，而 HTTPS 的端口号为 443。

简单地说，SSL/TLS 协议层主要的职责就是借助下层协议的信道安全地协商出一份加密密钥，并且用此密钥来加密 HTTP 请求响应报文。它解决了以下三个安全性方面的议题。

- 提供验证服务，验证本次会话实体身份的合法性。

- 提供加密服务，强加密机制能保证通信过程中的消息不会被破译。
- 提供防篡改服务，利用 Hash 算法对消息进行签名，通过验证签名保证通信内容不被篡改。



▲图 1.1　HTTP 与 HTTPS

HTTPS 运用越来越广泛，而且在安全场景中它是一个很好的解决方案，一般作为解决安全传输的首选解决方案。下面深入了解一下 HTTPS 的工作原理及流程。

在理解 HTTPS 工作原理前，先了解一些加密解密算法与 Hash 算法。

- 对称加密。密钥只有一个，加密、解密都是这个密码，加解密速度快，典型的对称加密算法有 DES、AES、RC4 等。
- 非对称加密。密钥成对出现，分别为公钥与私钥，从公钥无法推知私钥，反之，从私钥也不能推知公钥。加密、解密使用不同的密钥，公钥加密需要私钥解密，反之，私钥加密需要公钥解密。非对称加密速度较慢，典型的非对称加密算法有 RSA、DSA、DSS 等。
- Hash 算法，这是一种不可逆的算法，它常用于验证数据的完整性。

图 1.2 详细描述了 HTTPS 完成一次通信要做哪些事情。因为 HTTPS 是基于 TCP/IP 协议通信的，属于可靠传输，所以它必须要先进行三次握手，完成连接的建立。接着是 SSL 的握手协议，此协议非常有效地让客户和服务器之间完成相互之间的身份验证及密钥协商。

① 客户端浏览器向服务器发送 SSL/TLS 协议的版本号、加密算法的种类、产生的随机数，以及其他需要的各种信息。

② 服务器从客户端支持的加密算法中选择一组加密算法与 Hash 算法，并且把自己的证书（包含网站地址、加密公钥、证书颁发机构等）也发送给客户端。

③ 浏览器获取服务器证书后验证其合法性，验证颁发机构是否合法，验证证书中的网址是否与正在访问的地址一致，通过验证的浏览器会显示一个小锁头，否则，提示证书不受信。

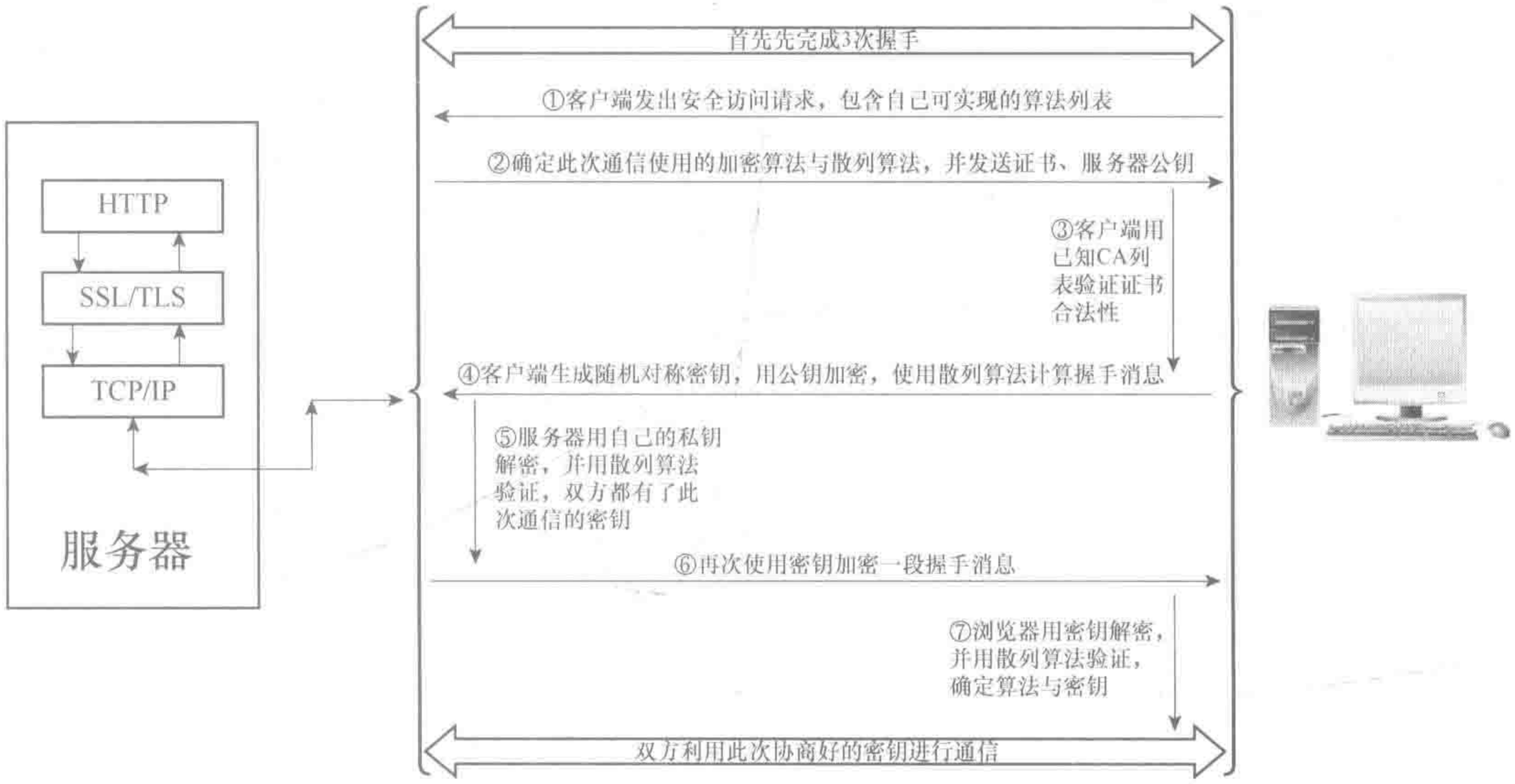
④ 客户端浏览器生成一串随机数并用服务器传来的公钥加密，再使用约定好的 Hash 算法计算握手消息，发送到服务器端。

⑤ 服务器接到握手消息后用自己的私钥解密，并用散列算法验证，这样双方都有了此次通信的密钥。

⑥ 服务器再使用密钥加密一段握手消息，返回给客户端浏览器。

⑦ 浏览器用密钥解密，并用散列算法验证，确定算法与密钥。

完成以上 7 步后双方就可以利用此次协商好的密钥进行通信。



▲图 1.2 HTTPS 的工作原理及流程

1.1.2 HTTP 请求/响应模型

从某种意义上来说，HTTP 协议永远都由客户端发起请求，由服务器进行响应并发送回响应报文。如果没有客户端进行请求或曾经请求过，那么服务器是无法将消息推送到客户端的。HTTP 采用了请求/响应模型，一个 HTTP 请求与响应一般如图 1.3 所示，客户端向服务器发送一个请求，请求头包含请求方法、URI、协议版本、请求修饰符、客户信息，以及类似于 MIME 结构的消息内容。服务器以一个状态行作为响应，内容包括消息协议版本、成功（或失败）编码、服务器信息、实体元信息及一些实体内容。这样就完成了一个请求/响应过程。

通常，一个 HTTP 请求/响应的工作流程大概可以用以下 4 步来概括。

① 客户端浏览器先要与服务器建立连接，即通过三次握手建立连接。在浏览器上最常见的场景就是单击一个链接，这就触发了连接的建立。

② 连接建立后，客户端浏览器发送一个请求到服务器，这个过程其实是组装请求报文的过程，详细的报文格式与解析会在下一节介绍。