

<<HTTP权威指南>>

图书基本信息

书名：<<HTTP权威指南>>

13位ISBN编号：9787115281487

10位ISBN编号：7115281483

出版时间：2012-9

出版单位：人民邮电出版社

作者：David Gourley,Brian Totty

页数：694

译者：陈涓,赵振平

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<HTTP权威指南>>

内容概要

《HTTP权威指南》是HTTP及其相关核心Web技术方面的权威著作，主要介绍了Web应用程序是如何工作的，核心的因特网协议如何与架构构建块交互，如何正确实现因特网客户和服务器等。

《HTTP权威指南》适合所有想了解HTTP和Web底层结构的人阅读。

<<HTTP权威指南>>

作者简介

David Gourley是Endeca的首席技术官（Chief Technology Officer），负责Endeca产品的研究及开发。

Endeca开发的因特网及内部网络信息访问解决方案为企业级数据的导航及研究提供了一些新的方式。在到Endeca工作之前，David是Inktomi基础工程组的一员，他在那儿帮助开发了Inktomi的因特网搜索数据库，是Inktomi的Web缓存产品的主要开发者。

David在加州大学伯克利分校获得了计算机科学的学士学位，还拥有Web技术方面的几项专利。

Brian

Totty最近出任了Inktomi公司（这是1996年他参与建立的一家公司）研发部副总裁，在公司中他负责Web缓存、流媒体及因特网搜索技术的研发工作。

他曾是Silicon

Graphics公司的一名科学家，他在那儿为高性能网络和超级计算机系统设计软件并对其进行优化。在那之前，他是苹果计算机公司高级技术组的一名工程师。

Brian在伊利诺伊大学Urbana-Champaign分校获得了计算机科学的博士学位，在MIT获得了计算机科学及电子工程的学士学位，在那里他获得了计算机系统研究的Organick奖。

他还为加州大学扩展系统开发并讲授了一些屡获殊荣的因特网技术方面的课程。

Marjorie

Sayer在Inktomi公司负责编写Web缓存方面的软件。

在加州大学伯克利分校获得了数学硕士和博士学位之后，一直致力于数学课程的改革。

从1990年开始致力于能量资源管理、并行系统软件、电话和网络方面的写作。

Sailu Reddy目前在Inktomi公司负责嵌入式的性能增强型HTTP

代理的开发。

Sailu从事复杂软件系统的开发已经有12年了，从1995年开始深入Web架构的研发工作。

他是Netscape第一台Web服务器、Web

代理产品，以及后面几代产品的核心工程师。

他具备HTTP应用程序、数据压缩技术、数据库引擎以及合作管理等方面的技术经验。

Sailu在亚里桑那大学获得了信息系统的硕士学位并握有Web技术方面的多项专利。

Anshu

Aggarwal是Inktomi公司的工程总监。

他领导着Inktomi公司Web缓存产品的协议处理工程组，从1997年就开始参与Inktomi的Web技术设计工作。

Anshu在科罗拉多大学Boulder分校获得了计算机科学的硕士和博士学位，从事分布式多处理器的内存一致性技术研究。

他还拥有电子工程的硕士和学士学位。

Anshu撰写了多篇技术论文，还拥有两项专利。

<<HTTP权威指南>>

书籍目录

- 第一部分 HTTP : Web的基础

- 第1章 HTTP概述 3

- 1.1 HTTP——因特网的多媒体信使 4

- 1.2 Web客户端和服务端 4

- 1.3 资源 5

- 1.3.1 媒体类型 6

- 1.3.2 URI 7

- 1.3.3 URL 7

- 1.3.4 URN 8

- 1.4 事务 9

- 1.4.1 方法 9

- 1.4.2 状态码 10

- 1.4.3 Web页面中可以包含多个对象 10

- 1.5 报文 11

- 1.6 连接 13

- 1.6.1 TCP/IP 13

- 1.6.2 连接、IP地址及端口号 14

- 1.6.3 使用Telnet实例 16

- 1.7 协议版本 18

- 1.8 Web的结构组件 19

- 1.8.1 代理 19

- 1.8.2 缓存 20

- 1.8.3 网关 20

- 1.8.4 隧道 21

- 1.8.5 Agent代理 21

- 1.9 起始部分的结束语 22

- 1.10 更多信息 22

- 1.10.1 HTTP协议信息 22

- 1.10.2 历史透视 23

- 1.10.3 其他万维网信息 23

-

- 第2章 URL与资源 25

- 2.1 浏览因特网资源 26

- 2.2 URL的语法 28

- 2.2.1 方案——使用什么协议 29

- 2.2.2 主机与端口 30

- 2.2.3 用户名和密码 30

- 2.2.4 路径 31

- 2.2.5 参数 31

- 2.2.6 查询字符串 32

- 2.2.7 片段 33

- 2.3 URL快捷方式 34

- 2.3.1 相对URL 34

- 2.3.2 自动扩展URL 37

<<HTTP权威指南>>

- 2.4 各种令人头疼的字符 38

- 2.4.1 URL字符集 38

- 2.4.2 编码机制 38

- 2.4.3 字符限制 39

- 2.4.4 另外一点说明 40

- 2.5 方案的世界 40

- 2.6 未来展望 42

- 2.7 更多信息 44

-

- 第3章 HTTP报文 45

- 3.1 报文流 46

- 3.1.1 报文流入源端服务器 46

- 3.1.2 报文向下游流动 47

- 3.2 报文的组成部分 47

- 3.2.1 报文的语法 48

- 3.2.2 起始行 50

- 3.2.3 首部 53

- 3.2.4 实体的主体部分 55

- 3.2.5 版本0.9的报文 55

- 3.3 方法 56

- 3.3.1 安全方法 56

- 3.3.2 GET 56

- 3.3.3 HEAD 57

- 3.3.4 PUT 57

- 3.3.5 POST 58

- 3.3.6 TRACE 58

- 3.3.7 OPTIONS 60

- 3.3.8 DELETE 60

- 3.3.9 扩展方法 61

- 3.4 状态码 62

- 3.4.1 100 ~ 199——信息性状态码 62

- 3.4.2 200 ~ 299——成功状态码 63

- 3.4.3 300 ~ 399——重定向状态码 64

- 3.4.4 400 ~ 499——客户端错误状态码 68

- 3.4.5 500 ~ 599——服务器错误状态码 69

- 3.5 首部 70

- 3.5.1 通用首部 71

- 3.5.2 请求首部 72

- 3.5.3 响应首部 74

- 3.5.4 实体首部 75

- 3.6 更多信息 77

-

- 第4章 连接管理 79

- 4.1 TCP连接 80

- 4.1.1 TCP的可靠数据管道 80

- 4.1.2 TCP流是分段的、由IP分组传送 81

- 4.1.3 保持TCP连接的正确运行 82

<<HTTP权威指南>>

- 4.1.4 用TCP套接字编程 84

- 4.2 对TCP性能的考虑 85

- 4.2.1 HTTP事务的时延 86

- 4.2.2 性能聚焦区域 87

- 4.2.3 TCP连接的握手时延 87

- 4.2.4 延迟确认 88

- 4.2.5 TCP慢启动 89

- 4.2.6 Nagle算法与TCP_NODELAY 89

- 4.2.7 TIME_WAIT累积与端口耗尽 90

- 4.3 HTTP连接的处理 91

- 4.3.1 常被误解的Connection首部 91

- 4.3.2 串行事务处理时延 92

- 4.4 并行连接 94

- 4.4.1 并行连接可能会提高页面的加载速度 94

- 4.4.2 并行连接不一定更快 95

- 4.4.3 并行连接可能让人“感觉”更快一些 95

- 4.5 持久连接 96

- 4.5.1 持久以及并行连接 96

- 4.5.2 HTTP/1.0+ keep-alive连接 97

- 4.5.3 Keep-Alive操作 98

- 4.5.4 Keep-Alive选项 98

- 4.5.5 Keep-Alive连接的限制和规则 99

- 4.5.6 Keep-Alive和哑代理 100

- 4.5.7 插入Proxy-Connection 102

- 4.5.8 HTTP/1.1持久连接 104

- 4.5.9 持久连接的限制和规则 104

- 4.6 管道化连接 105

- 4.7 关闭连接的奥秘 106

- 4.7.1 “任意”解除连接 106

- 4.7.2 Content-Length及截尾操作 107

- 4.7.3 连接关闭容限、重试以及幂等性 107

- 4.7.4 正常关闭连接 108

- 4.8 更多信息 110

- 4.8.1 HTTP连接 110

- 4.8.2 HTTP性能问题 110

- 4.8.3 TCP/IP 111

-

- 第二部分 HTTP结构

-

- 第5章 Web服务器 115

- 5.1 各种形状和尺寸的Web服务器 116

- 5.1.1 Web服务器的实现 116

- 5.1.2 通用软件Web服务器 117

- 5.1.3 Web服务器设备 117

- 5.1.4 嵌入式Web服务器 118

- 5.2 最小的Perl Web服务器 118

- 5.3 实际的Web服务器会做些什么 120

<<HTTP权威指南>>

- 5.4 第一步——接受客户端连接 121

- 5.4.1 处理新连接 121

- 5.4.2 客户端主机名识别 122

- 5.4.3 通过ident确定客户端用户 122

- 5.5 第二步——接收请求报文 123

- 5.5.1 报文的内部表示法 124

- 5.5.2 连接的输入/输出处理结构 125

- 5.6 第三步——处理请求 126

- 5.7 第四步——对资源的映射及访问 126

- 5.7.1 docroot 127

- 5.7.2 目录列表 129

- 5.7.3 动态内容资源的映射 130

- 5.7.4 服务器端包含项 131

- 5.7.5 访问控制 131

- 5.8 第五步——构建响应 131

- 5.8.1 响应实体 131

- 5.8.2 MIME类型 132

- 5.8.3 重定向 133

- 5.9 第六步——发送响应 134

- 5.10 第七步——记录日志 134

- 5.11 更多信息 134

-

- 第6章 代理 135

- 6.1 Web的中间实体 136

- 6.1.1 私有和共享代理 136

- 6.1.2 代理与网关的对比 137

- 6.2 为什么使用代理 138

- 6.3 代理会去往何处 143

- 6.3.1 代理服务器的部署 144

- 6.3.2 代理的层次结构 144

- 6.3.3 代理是如何获取流量的 147

- 6.4 客户端的代理设置 148

- 6.4.1 客户端的代理配置：手工配置 149

- 6.4.2 客户端代理配置：PAC文件 149

- 6.4.3 客户端代理配置：WPAD 150

- 6.5 与代理请求有关的一些棘手问题 151

- 6.5.1 代理URI与服务器URI的不同 151

- 6.5.2 与虚拟主机一样的问题 152

- 6.5.3 拦截代理会收到部分URI 153

- 6.5.4 代理既可以处理代理请求，也可以处理服务器请求 154

- 6.5.5 转发过程中对URI的修改 154

- 6.5.6 URI的客户端自动扩展和主机名解析 155

- 6.5.7 没有代理时URI的解析 155

- 6.5.8 有显式代理时URI的解析 156

- 6.5.9 有拦截代理时URI的解析 157

- 6.6 追踪报文 158

- 6.6.1 Via首部 158

<<HTTP权威指南>>

- 6.6.2 TRACE方法 162

- 6.7 代理认证 164

- 6.8 代理的互操作性 165

- 6.8.1 处理代理不支持的首部和方法 166

- 6.8.2 OPTIONS : 发现对可选特性的支持 166

- 6.8.3 Allow首部 167

- 6.9 更多信息 167

-

- 第7章 缓存 169

- 7.1 冗余的数据传输 170

- 7.2 带宽瓶颈 170

- 7.3 瞬间拥塞 171

- 7.4 距离时延 172

- 7.5 命中和未命中的 173

- 7.5.1 再验证 173

- 7.5.2 命中率 175

- 7.5.3 字节命中率 176

- 7.5.4 区分命中和未命中的情况 176

- 7.6 缓存的拓扑结构 177

- 7.6.1 私有缓存 177

- 7.6.2 公有代理缓存 177

- 7.6.3 代理缓存的层次结构 179

- 7.6.4 网状缓存、内容路由以及对等缓存 180

- 7.7 缓存的处理步骤 181

- 7.7.1 第一步——接收 181

- 7.7.2 第二步——解析 182

- 7.7.3 第三步——查找 182

- 7.7.4 第四步——新鲜度检测 182

- 7.7.5 第五步——创建响应 182

- 7.7.6 第六步——发送 183

- 7.7.7 第七步——日志 183

- 7.7.8 缓存处理流程图 183

- 7.8 保持副本的新鲜 183

- 7.8.1 文档过期 184

- 7.8.2 过期日期和使用期 185

- 7.8.3 服务器再验证 185

- 7.8.4 用条件方法进行再验证 186

- 7.8.5 If-Modified-Since:Date再验证 187

- 7.8.6 If-None-Match : 实体标签再验证 189

- 7.8.7 强弱验证器 190

- 7.8.8 什么时候应该使用实体标签和最近修改日期 190

- 7.9 控制缓存的能力 191

- 7.9.1 no-Store与no-Cache响应首部 191

- 7.9.2 max-age响应首部 192

- 7.9.3 Expires响应首部 192

- 7.9.4 must-revalidate响应首部 192

- 7.9.5 试探性过期 193

<<HTTP权威指南>>

- 7.9.6 客户端的新鲜度限制 194

- 7.9.7 注意事项 194

- 7.10 设置缓存控制 195

- 7.10.1 控制Apache的HTTP首部 195

- 7.10.2 通过HTTP-EQUIV控制HTML缓存 196

- 7.11 详细算法 197

- 7.11.1 使用期和新鲜生存期 198

- 7.11.2 使用期的计算 198

- 7.11.3 完整的使用期计算算法 201

- 7.11.4 新鲜生存期计算 202

- 7.11.5 完整的服务器——新鲜度算法 202

- 7.12 缓存和广告 204

- 7.12.1 发布广告者的两难处境 204

- 7.12.2 发布者的响应 204

- 7.12.3 日志迁移 205

- 7.12.4 命中计数和使用限制 205

- 7.13 更多信息 205

-

- 第8章 集成点：网关、隧道及中继 207

- 8.1 网关 208

- 8.2 协议网关 210

- 8.2.1 HTTP/*：服务器端Web网关 211

- 8.2.2 HTTP/HTTPS：服务器端安全网关 212

- 8.2.3 HTTPS/HTTP客户端安全加速器网关 212

- 8.3 资源网关 213

- 8.3.1 CGI 215

- 8.3.2 服务器扩展API 215

- 8.4 应用程序接口和Web服务 216

- 8.5 隧道 217

- 8.5.1 用CONNECT建立HTTP隧道 217

- 8.5.2 数据隧道、定时及连接管理 219

- 8.5.3 SSL隧道 219

- 8.5.4 SSL隧道与HTTP/HTTPS网关的对比 220

- 8.5.5 隧道认证 221

- 8.5.6 隧道的安全性考虑 221

- 8.6 中继 222

- 8.7 更多信息 224

-

- 第9章 Web机器人 225

- 9.1 爬虫及爬行方式 226

- 9.1.1 从哪儿开始：根集 226

- 9.1.2 链接的提取以及相对链接的标准化 227

- 9.1.3 避免环路的出现 228

- 9.1.4 循环与复制 228

- 9.1.5 面包屑留下的痕迹 229

- 9.1.6 别名与机器人环路 230

- 9.1.7 规范化URL 230

<<HTTP权威指南>>

- 9.1.8 文件系统连接环路 231

- 9.1.9 动态虚拟Web空间 232

- 9.1.10 避免循环和重复 233

- 9.2 机器人的HTTP 236

- 9.2.1 识别请求首部 236

- 9.2.2 虚拟主机 236

- 9.2.3 条件请求 237

- 9.2.4 对响应的处理 238

- 9.2.5 User-Agent导向 239

- 9.3 行为不当的机器人 239

- 9.4 拒绝机器人访问 240

- 9.4.1 拒绝机器人访问标准 241

- 9.4.2 网站点和robots.txt文件 242

- 9.4.3 robots.txt文件的格式 243

- 9.4.4 其他有关robots.txt的知识 246

- 9.4.5 缓存和robots.txt的过期 246

- 9.4.6 拒绝机器人访问的Perl代码 246

- 9.4.7 HTML的robot-control元标签 249

- 9.5 机器人的规范 251

- 9.6 搜索引擎 254

- 9.6.1 大格局 255

- 9.6.2 现代搜索引擎结构 255

- 9.6.3 全文索引 255

- 9.6.4 发布查询请求 257

- 9.6.5 对结果进行排序, 并提供查询结果 258

- 9.6.6 欺诈 258

- 9.7 更多信息 258

-

- 第10章 HTTP-NG 261

- 10.1 HTTP发展中存在的问题 262

- 10.2 HTTP-NG的活动 263

- 10.3 模块化及功能增强 263

- 10.4 分布式对象 264

- 10.5 第一层——报文传输 264

- 10.6 第二层——远程调用 265

- 10.7 第三层——Web应用 265

- 10.8 WebMUX 265

- 10.9 二进制连接协议 266

- 10.10 当前的状态 267

- 10.11 更多信息 267

-

- 第三部分 识别、认证与安全

-

- 第11章 客户端识别与cookie机制 271

- 11.1 个性化接触 272

- 11.2 HTTP首部 273

- 11.3 客户端IP地址 274

<<HTTP权威指南>>

- 11.4 用户登录 275

- 11.5 胖URL 277

- 11.6 cookie 278

 - 11.6.1 cookie的类型 278

 - 11.6.2 cookie是如何工作的 279

 - 11.6.3 cookie罐：客户端的状态 280

 - 11.6.4 不同站点使用不同的cookie 282

 - 11.6.5 cookie成分 283

 - 11.6.6 cookies版本0(Netscape) 284

 - 11.6.7 cookies版本1(RFC 2965) 285

 - 11.6.8 cookie与会话跟踪 288

 - 11.6.9 cookie与缓存 290

 - 11.6.10 cookie、安全性和隐私 291

- 11.7 更多信息 292

-

- 第12章 基本认证机制 293

 - 12.1 认证 294

 - 12.1.1 HTTP的质询/响应认证框架 294

 - 12.1.2 认证协议与首部 295

 - 12.1.3 安全域 296

 - 12.2 基本认证 297

 - 12.2.1 基本认证实例 298

 - 12.2.2 Base-64用户名/密码编码 298

 - 12.2.3 代理认证 299

 - 12.3 基本认证的安全缺陷 300

 - 12.4 更多信息 301

-

- 第13章 摘要认证 303

 - 13.1 摘要认证的改进 304

 - 13.1.1 用摘要保护密码 304

 - 13.1.2 单向摘要 306

 - 13.1.3 用随机数防止重放攻击 307

 - 13.1.4 摘要认证的握手机制 307

 - 13.2 摘要的计算 308

 - 13.2.1 摘要算法的输入数据 308

 - 13.2.2 算法H(d)和KD(s,d) 310

 - 13.2.3 与安全性相关的数据(A1) 310

 - 13.2.4 与报文有关的数据(A2) 310

 - 13.2.5 摘要算法总述 311

 - 13.2.6 摘要认证会话 312

 - 13.2.7 预授权 312

 - 13.2.8 随机数的选择 315

 - 13.2.9 对称认证 315

 - 13.3 增强保护质量 316

 - 13.3.1 报文完整性保护 316

 - 13.3.2 摘要认证首部 317

 - 13.4 应该考虑的实际问题 317

<<HTTP权威指南>>

- 13.4.1 多重质询 318

- 13.4.2 差错处理 318

- 13.4.3 保护空间 318

- 13.4.4 重写URI 319

- 13.4.5 缓存 319

- 13.5 安全性考虑 320

- 13.5.1 首部篡改 320

- 13.5.2 重放攻击 320

- 13.5.3 多重认证机制 320

- 13.5.4 词典攻击 321

- 13.5.5 恶意代理攻击和中间人攻击 321

- 13.5.6 选择明文攻击 321

- 13.5.7 存储密码 322

- 13.6 更多信息 322

-

- 第14章 安全HTTP 323

- 14.1 保护HTTP的安全 324

- 14.2 数字加密 326

- 14.2.1 密码编制的机制与技巧 326

- 14.2.2 密码 327

- 14.2.3 密码机 328

- 14.2.4 使用了密钥的密码 328

- 14.2.5 数字密码 328

- 14.3 对称密钥加密技术 330

- 14.3.1 密钥长度与枚举攻击 330

- 14.3.2 建立共享密钥 332

- 14.4 公开密钥加密技术 332

- 14.4.1 RSA 333

- 14.4.2 混合加密系统和会话密钥 334

- 14.5 数字签名 334

- 14.6 数字证书 336

- 14.6.1 证书的主要内容 336

- 14.6.2 X.509 v3证书 337

- 14.6.3 用证书对服务器进行认证 338

- 14.7 HTTPS——细节介绍 339

- 14.7.1 HTTPS概述 339

- 14.7.2 HTTPS方案 340

- 14.7.3 建立安全传输 341

- 14.7.4 SSL握手 341

- 14.7.5 服务器证书 343

- 14.7.6 站点证书的有效性 344

- 14.7.7 虚拟主机与证书 345

- 14.8 HTTPS客户端实例 345

- 14.8.1 OpenSSL 346

- 14.8.2 简单的HTTPS客户端 347

- 14.8.3 执行OpenSSL客户端 350

- 14.9 通过代理以隧道形式传输安全流量 351

<<HTTP权威指南>>

- 14.10 更多信息 353

- 14.10.1 HTTP安全性 353

- 14.10.2 SSL与TLS 353

- 14.10.3 公开密钥基础设施 354

- 14.10.4 数字密码 354

-

- 第四部分 实体、编码和国际化

-

- 第15章 实体和编码 357

- 15.1 报文是箱子, 实体是货物 359

- 15.2 Content-Length: 实体的大小 361

- 15.2.1 检测截尾 361

- 15.2.2 错误的Content-Length 362

- 15.2.3 Content-Length与持久连接 362

- 15.2.4 内容编码 362

- 15.2.5 确定实体主体长度的规则 362

- 15.3 实体摘要 364

- 15.4 媒体类型和字符集 364

- 15.4.1 文本的字符编码 365

- 15.4.2 多部分媒体类型 365

- 15.4.3 多部分表格提交 366

- 15.4.4 多部分范围响应 367

- 15.5 内容编码 368

- 15.5.1 内容编码过程 368

- 15.5.2 内容编码类型 369

- 15.5.3 Accept-Encoding首部 369

- 15.6 传输编码和分块编码 371

- 15.6.1 可靠传输 371

- 15.6.2 Transfer-Encoding首部 372

- 15.6.3 分块编码 373

- 15.6.4 内容编码与传输编码的结合 375

- 15.6.5 传输编码的规则 375

- 15.7 随时间变化的实例 375

- 15.8 验证码和新鲜度 376

- 15.8.1 新鲜度 377

- 15.8.2 有条件的请求与验证码 378

- 15.9 范围请求 380

- 15.10 差异编码 382

- 15.11 更多信息 385

-

- 第16章 国际化 387

- 16.1 HTTP对国际性内容的支持 388

- 16.2 字符集与HTTP 389

- 16.2.1 字符集是把字符转换为二进制码的编码 389

- 16.2.2 字符集和编码如何工作 390

- 16.2.3 字符集不对, 字符就不对 391

- 16.2.4 标准化的MIME charset值 391

<<HTTP权威指南>>

- 16.2.5 Content-Type首部和Charset首部以及META标志 393

- 16.2.6 Accept-Charset首部 393

- 16.3 多语言字符编码入门 394

- 16.3.1 字符集术语 394

- 16.3.2 字符集的命名很糟糕 395

- 16.3.3 字符 396

- 16.3.4 字形、连笔以及表示形式 396

- 16.3.5 编码后的字符集 397

- 16.3.6 字符编码方案 399

- 16.4 语言标记与HTTP 402

- 16.4.1 Content-Language首部 402

- 16.4.2 Accept-Language首部 403

- 16.4.3 语言标记的类型 404

- 16.4.4 子标记 404

- 16.4.5 大小写 405

- 16.4.6 IANA语言标记注册 405

- 16.4.7 第一个子标记——名字空间 405

- 16.4.8 第二个子标记——名字空间 406

- 16.4.9 其余子标记——名字空间 407

- 16.4.10 配置和语言有关的首选项 407

- 16.4.11 语言标记参考表 407

- 16.5 国际化的URI 408

- 16.5.1 全球性的可转抄能力与有意义的字符的较量 408

- 16.5.2 URI字符集合 408

- 16.5.3 转义和反转义 409

- 16.5.4 转义国际化字符 409

- 16.5.5 URI中的模态切换 410

- 16.6 其他需要考虑的地方 410

- 16.6.1 首部和不合规范的数据 410

- 16.6.2 日期 411

- 16.6.3 域名 411

- 16.7 更多信息 411

- 16.7.1 附录 411

- 16.7.2 互联网的国际化 411

- 16.7.3 国际标准 412

-

- 第17章 内容协商与转码 413

- 17.1 内容协商技术 414

- 17.2 客户端驱动的协商 415

- 17.3 服务器驱动的协商 415

- 17.3.1 内容协商首部集 416

- 17.3.2 内容协商首部中的质量值 417

- 17.3.3 随其他首部集而变化 417

- 17.3.4 Apache中的内容协商 417

- 17.3.5 服务器端扩展 418

- 17.4 透明协商 419

- 17.4.1 进行缓存与备用候选 419

<<HTTP权威指南>>

- 17.4.2 Vary首部 420

- 17.5 转码 422

- 17.5.1 格式转换 422

- 17.5.2 信息综合 423

- 17.5.3 内容注入 423

- 17.5.4 转码与静态预生成的对比 423

- 17.6 下一步计划 424

- 17.7 更多信息 424

-

- 第五部分 内容发布与分发

-

- 第18章 Web主机托管 429

- 18.1 主机托管服务 430

- 18.2 虚拟主机托管 431

- 18.2.1 虚拟服务器请求缺乏主机信息 432

- 18.2.2 设法让虚拟主机托管正常工作 433

- 18.2.3 HTTP/1.1的Host首部 437

- 18.3 使网站更可靠 438

- 18.3.1 镜像的服务器集群 438

- 18.3.2 内容分发网络 440

- 18.3.3 CDN中的反向代理缓存 440

- 18.3.4 CDN中的代理缓存 440

- 18.4 让网站更快 441

- 18.5 更多信息 441

-

- 第19章 发布系统 443

- 19.1 FrontPage为支持发布而做的服务器扩展 444

- 19.1.1 FrontPage服务器扩展 444

- 19.1.2 FrontPage术语表 445

- 19.1.3 FrontPage的RPC协议 445

- 19.1.4 FrontPage的安全模型 448

- 19.2 WebDAV与协作写作 449

- 19.2.1 WebDAV的方法 449

- 19.2.2 WebDAV与XML 450

- 19.2.3 WebDAV首部集 451

- 19.2.4 WebDAV的锁定与防止覆写 452

- 19.2.5 LOCK方法 453

- 19.2.6 UNLOCK方法 456

- 19.2.7 属性和元数据 456

- 19.2.8 PROPFIND方法 457

- 19.2.9 PROPPATCH方法 459

- 19.2.10 集合与名字空间管理 460

- 19.2.11 MKCOL方法 460

- 19.2.12 DELETE方法 461

- 19.2.13 COPY与MOVE方法 462

- 19.2.14 增强的HTTP/1.1方法 465

- 19.2.15 WebDAV中的版本管理 466

<<HTTP权威指南>>

- 19.2.16 WebDAV的未来发展 466

- 19.3 更多信息 467

-

- 第20章 重定向与负载均衡 469

- 20.1 为什么要重定向 470

- 20.2 重定向到何地 471

- 20.3 重定向协议概览 471

- 20.4 通用的重定向方法 474

- 20.4.1 HTTP重定向 474

- 20.4.2 DNS重定向 475

- 20.4.3 任播寻址 480

- 20.4.4 IP MAC转发 481

- 20.4.5 IP地址转发 482

- 20.4.6 网元控制协议 484

- 20.5 代理的重定向方法 485

- 20.5.1 显式浏览器配置 485

- 20.5.2 代理自动配置 485

- 20.5.3 Web代理自动发现协议 487

- 20.6 缓存重定向方法 492

- 20.7 因特网缓存协议 496

- 20.8 缓存阵列路由协议 497

- 20.9 超文本缓存协议 500

- 20.9.1 HTCP认证 502

- 20.9.2 设置缓存策略 503

- 20.10 更多信息 504

-

- 第21章 日志记录与使用情况跟踪 505

- 21.1 记录内容 506

- 21.2 日志格式 507

- 21.2.1 常见日志格式 507

- 21.2.2 组合日志格式 508

- 21.2.3 网景扩展日志格式 509

- 21.2.4 网景扩展2日志格式 510

- 21.2.5 Squid代理日志格式 512

- 21.3 命中率测量 515

- 21.3.1 概述 515

- 21.3.2 Meter首部 516

- 21.4 关于隐私的考虑 517

- 21.5 更多信息 518

-

- 第六部分 附录

- 附录A URI方案 521

- 附录B HTTP状态码 529

- 附录C HTTP首部参考 533

- 附录D MIME类型 557

- 附录E Base-64编码 603

- 附录F 摘要认证 607

<<HTTP权威指南>>

附录G 语言标记 615

附录H MIME字符集注册表 641

索引 661

章节摘录

版权页：插图：7.6.4 网状缓存、内容路由以及对等缓存 有些网络结构会构建复杂的网状缓存（cache mesh），而不是简单的缓存层次结构。

网状缓存中的代理缓存之间会以更加复杂的方式进行对话，做出动态的缓存通信决策，决定与哪个父缓存进行对话，或者决定彻底绕开缓存，直接连接原始服务器。

这种代理缓存会决定选择何种路由对内容进行访问、管理和传送，因此可将其称为内容路由器（content router）。

网状缓存中为内容路由设计的缓存（除了其他任务之外）要完成下列所有功能。

根据URL在父缓存或原始服务器之间进行动态选择。

根据URL动态地选择一个特定的父缓存。

前往父缓存之前，在本地缓存中搜索已缓存的副本。

允许其他缓存对其缓存的部分内容进行访问，但不允许因特网流量通过它们的缓存。

缓存之间这些更为复杂的关系允许不同的组织互为对等（peer）实体，将它们的缓存连接起来以实现共赢。

提供可选的对等支持的缓存被称为兄弟缓存（siblingcache）（参见图7—10）。

HTTP并不支持兄弟缓存，所以人们通过一些协议对HTTP进行了扩展，比如因特网缓存协议（Internet Cache Protocol, ICP）和超文本缓存协议（HyperText Caching Protocol, HTCP）。

我们将在第20章讨论这些协议。

7.7缓存的处理步骤 现代的商业化代理缓存相当地复杂。

这些缓存构建得非常高效，可以支持HTTP和其他一些技术的各种高级特性。

但除了一些微妙的细节之外，Web缓存的基本工作原理大多很简单。

对一条HTTP GET报文的基本缓存处理过程包括7个步骤（参见图7—11）。

（1）接收——缓存从网络中读取抵达的请求报文。

（2）解析——缓存对报文进行解析，提取出URL和各种首部。

（3）查询——缓存查看是否有本地副本可用，如果没有，就获取一份副本（并将其保存在本地）。

（4）新鲜度检测——缓存查看已缓存副本是否足够新鲜，如果不是，就询问服务器是否有任何更新

（5）创建响应——缓存会用新的首部和已缓存的主体来构建一条响应报文。

（6）发送——缓存通过网络将响应发回给客户端。

（7）日志——缓存可选地创建一个日志文件条目来描述这个事务。

<<HTTP权威指南>>

编辑推荐

《HTTP权威指南》由古尔利所著，本书详细解释了HTTP协议，包括它是如何工作的，如何用它来开发基于Web的应用程序，同时还探讨了HTTP有效工作所依赖的所有其他核心因特网技术。尽管HTTP是本书的中心内容，但本书的本质是理解Web的工作原理，以及如何将这些知识应用到Web编程和管理之中，主要涵盖HTTP的技术运作方式、产生动机、性能和目标以及一些相关技术问题。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>