基于 AJAX 的 Client Puzzle 协议框架的研究与实现

The Application and Research of CPP Protocol to Defend Against Resource – Depletion DoS Attack

邱科宁　（仲恺农业技术学院计算机科学与工程学院 510225）

摘要：本文在研究了基于主动式 Client Puzzle 协议的 HTTP 请求分类服务系统模型（CPPWQ）的基础上，结合 AJAX 技术，实现了一个改进的 ACPP 模型。在对该模型的分析和设计基础上，本文给出了它的一个实现原型，并与原有模型做比较。

关键词：AJAX　Client Puzzle 协议　请求分类服务系统模型

1　引言

已有的对 Client Puzzle 协议（以下简称 CPP 协议）研究大多都集中在 IP 层和 TCP 层[1][2][3][4][5][6]，以及传输层上层的少数协议[7]（如 SSL/TLS），然而，在这些层上实现都必须修改操作系统内核，因此它的可扩展性很差。基于此，笔者提出一个防御资源耗尽型 DoS 攻击的基于主动式 CPP 协议的 HTTP 请求分类服务系统模型 CPPWQ。该模型是在应用层把 CPP 协议与分类服务思想结合起来，充分利用分类服务思想的优点和 CPP 协议的有效性。CPPWQ 模型的客户端在实现上主要采用了网关技术，有效地避免了修改内核所带来的巨大工作量，但是，由于每个客户端都需要安装特定的 CPP 网关，因此限制了该模型的广泛应用。

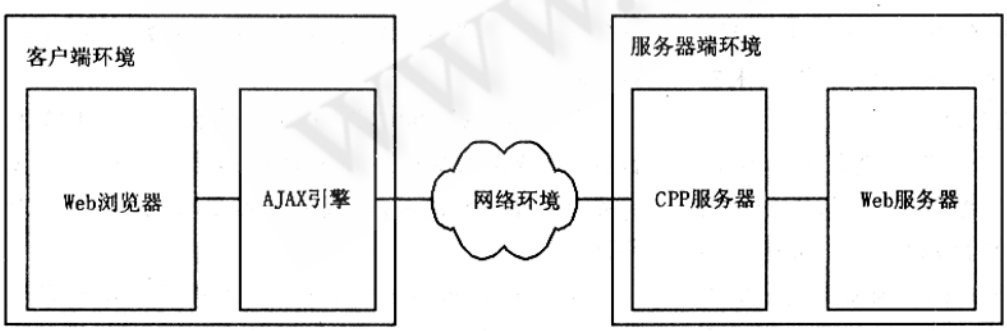

图 1　ACPP 模型的体系结构

AJAX 技术是异步 JavaScript 和 XML 这两项技术的统称，专为描述 JavaScript 的两项强大性能. 这两项性能在多年来一直被网络开发者所忽略，直到最近 Gmail，Google suggest 和 google Maps 的横空出世才使人们开始意识到其重要性。

这两项被忽视的性能是：

无需重新装载整个页面便能向服务器发送请求；

对 XML 文档的解析和处理。

本文提出基于 AJAX 的 Client Puzzle 协议框架 ACPP，该框架充分利用了 AJAX 技术的优点，在客户端用 JavaScript 代替了原有的 CPPWQ 模型中的网关的角色，为 CPP 协议的大规模应用打下坚实的基础。

2　ACPP 体系结构和功能组成

2.1　ACPP 体系结构

图 1 所示为 ACPP 模型的体系结构。

该模型由 Web 浏览器，AJAX 引擎，CPP 服务器和 Web 服务器组成。与普通的浏览器/服务器架构相比，该模型增加了 AJAX 引擎，CPP 服务器两部分。AJAX 引擎和 CPP 服务器是模型的核心，负责 CPP 协议的通信，其中，AJAX 引擎位于客户端，负责 CPP 协议的建立与运作过程，CPP 服务器位于服务器端，负责 CPP 协议的验证，

并把通过验证的 HTTP 请求转发给 Web 服务器做进一步的处理。

2.2 ACPP 功能组成

图 2 所示为 ACPP 模型的功能结构，以及它们之间的通信消息。

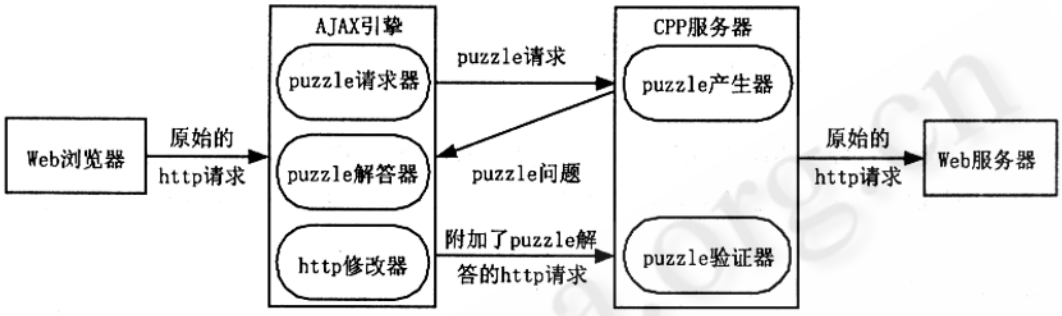

图 2　ACPP 的功能结构图

从图中可以看出，AJAX 引擎接收到从 Web 浏览器来的 HTTP 请求后，启动 CPP 协议与 CPP 服务器通信：

- AJAX 引擎把 HTTP 请求缓存起来。
- AJAX 引擎向 CPP 服务器发送 puzzle 问题的请求。
- CPP 服务器通信根据 Web 服务器的当前负载决定 puzzle 的难度系数，并构建对应的 puzzle，以 puzzle 问题的形式返回给 CPP 网关。
- AJAX 引擎接收到 puzzle 问题后解答之，从缓冲区中取出 HTTP 请求，并把 puzzle 解答附加在该 HTTP 请求上，最后向 CPP 服务器发送该请求。
- CPP 服务器对接收的附加了 puzzle 解答的 HTTP 请求进行验证，如果 puzzle 解答有误，则丢弃该 HTTP 请求；如果 puzzle 解答正确，则接纳该 HTTP 请求并转发给 Web 服务器作最终的处理。

(1) AJAX 引擎。puzzle 请求器接收到 Web 浏览器来的 http 请求，负责选择恰当的时机发送 puzzle 请求。通常选择在服务器遭受到 puzzle 攻击时发送 puzzle 请求，而在正常场合则正常访问。判断服务器是否受到攻击是当前的请求是否被拒绝，如果当前的请求没有被拒绝，即有正常的响应，则不发送 puzzle 请求，否则，发送 puzzle 请求。

puzzle 解答器对 puzzle 问题进行求解，通常对 puzzle 问题的求解会耗费自身的部分资源，puzzle 的难度系数越大，需要耗费自身的资源就越多，因此，DoS 攻击的发起者就会因为需要同时解答大量 puzzle 问题而耗光自身的资源，达到限制其自身攻击能力的目的。

在解答了 puzzle 问题后得到 puzzle 答案，ajax 引擎从从缓冲区中取出 HTTP 请求，并调用 http 修改器把 puzzle 解答附加在该 HTTP 请求上。

(2) CPP 服务器。CPP 服务器包含 puzzle 产生器和 puzzle 验证器两个模块，puzzle 产生器根据客户端中 puzzle 请求器的 puzzle 请求和自身的资源负荷情况决定 puzzle 的难度系数。自身的资源负荷越重，则 puzzle 的难度系数越高，而自身的负荷可以由当前对客户请求的服务线程数来近似衡量。

在接收到附加了 puzzle 解答的 http 请求后，puzzle 验证器首先从该 http 请求中提取出 puzzle 解答，然后验证该解答是否正确，如果正确，则服务该 http 请求，否则，拒绝该 http 请求。

3　ACPP 模型性能分析

对于 WEB 服务器的压力测试也可以转化为一次 DoS 攻击。本文采用的就是这种方式，用到的压力测试工具为 jmeter。除了发送访问请求外，jmeter 还可以监控服务器的性能，包括负载，内存使用率，活动线程数等。

Web 服务器：tomcat5.5，提供的服务程序为样本容量 828 MB 的搜索工具 luceneweb，并用 jmeter 对服务器端性能进行测试，设置 Thread Delay 设置为 1000 毫秒。

客户端：用 jmeter 模拟 WWW 浏览器对 Web 服务器进行访问，其 Loop Count 设置为 Forever，Ramp－Up Period in seconds 设置为 0，Number of Threads 设置为 75。

在没有 CPPWQ 对 Web 服务器保护，有 CPPWQ 对 Web 服务器保护且难度系数(x)分别为 5、10、15 这四

种情况下，在服务器端观测到访问流量分布情况分别如图 3、图 4、图 5 和图 6 所示。

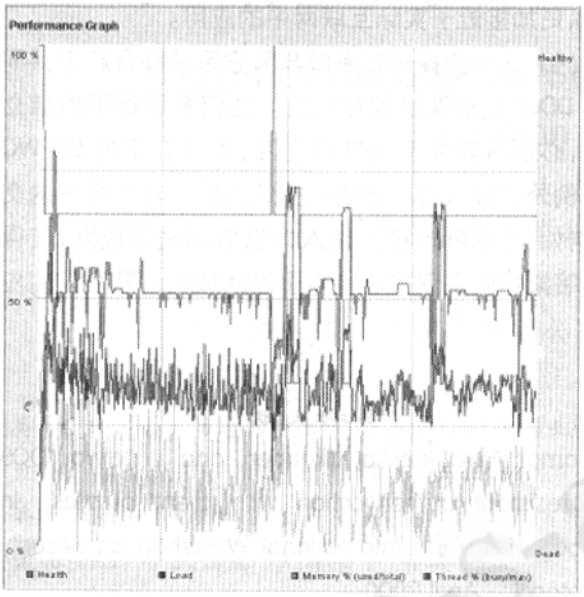

图 3　没有 CPPWQ

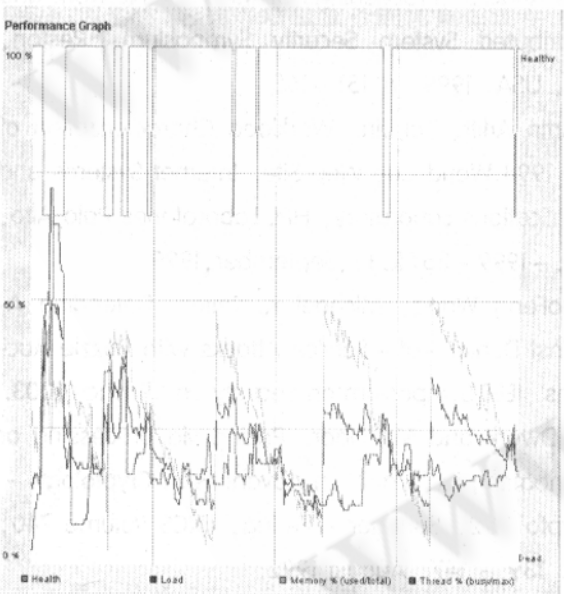

图 4　有 CPPWQ，x = 5

从上面四个图可以看出，一方面，在没有 CPPWQ 对 Web 服务器保护的情况下，服务器端的性能退化比有 CPPWQ 的情况下严重，这说明在同样攻击强度下，CPPWQ 可以很好地保护 Web 服务器，使服务器的性能有良好的退化，可以提供更多的对外服务。另一方面，在同一攻击强度下，难度系数越大，服务器端性能的影响就越小，这是因为难度系数越大，成功发起每次请求所需要的客户端资源就越多，从而客户端能够发动攻击的频率就越小，最终获得服务器服务的请求就越小。

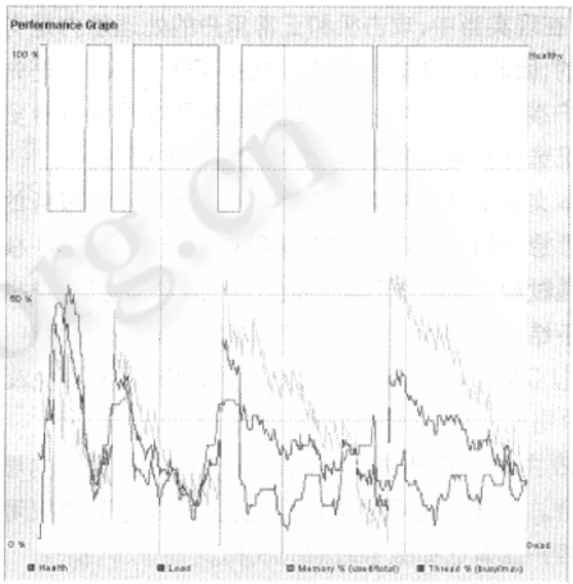

图 5　有 CPPWQ，x = 10

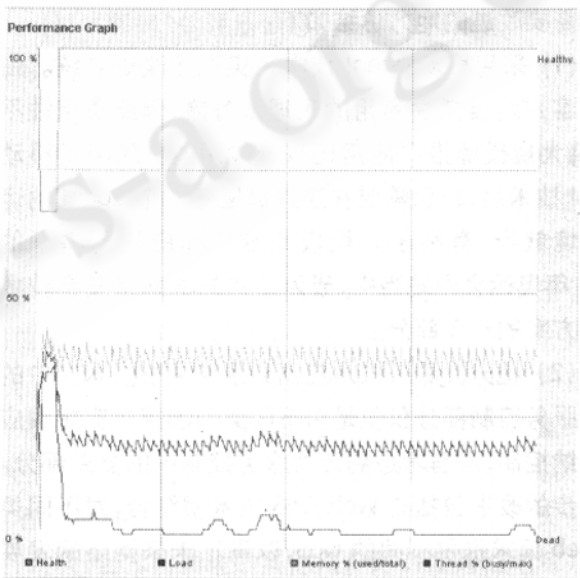

图 6　有 CPPWQ，x = 15

4 总结与展望

4.1 总结

传统的被动式 CPP 协议是由服务器端先发起的，在服务器端看来，客户端的处理能力是没有差别的。然而，在现实当中，攻击机和正常用户的处理能力是有差别的，因此，本文提出了主动式的 CPP 协议，它是先由客户端发起的，并且每个客户端都可以根据自身资源状况选择恰当的参数，获得相应级别的服务。

本文以主动式 CPP 协议为核心，结合分类服务的基本思想，设计并实现了支持 CPP 协议的 CPPWQ 模型。通过理论探讨、模型分析、设计、实现和验证，得出了以下结论：

① CPP 协议可以很好地保护 Web 服务器免受 DoS 攻击。

② 主动式 CPP 协议可以很好地跟分类服务的思想结合。与被动式 CPP 协议相比，主动式 CPP 协议可以给客户端有选择服务级别的权利。

4.2 展望

本文仅仅提出了 CPPWQ 的基本思想并实现了一个简单的原型系统，没能对 CPPWQ 作进一步优化，将来还需要改进的地方总结如下：

(1) 采用移动 agent 技术来实现和改进系统。由于在客户端要搭建专用的代理服务器，因此通用性不高，离大规模地推广应用还有一段距离。采用了移动 agent 技术后，仅仅需要在客户端搭建一个 agent 的运行环境就行，有利于大规模的推广和应用到将来的 2008 年奥运会网站当中，成为该网站的网络安全技术解决方案的一个部分。

(2) 把分类服务技术应用到系统中去。本文中的分类服务控制部分仅仅是对不同类别的请求实施相应的接纳控制，没有考虑到分类服务控制中的复杂部分。进一步的改进包括与 Web QoS 技术相结合，对不同类的 web 请求提供不同的 QoS，以满足未来网络的发展方向。

(3) 在复杂的 Internet 环境下对系统性能进行测评。由于本文是在局域网条件下评测系统性能，距离真正应用于 Internet 上还有一段距离，因此，进一步的研究还包括搭建一个复杂的模拟 Internet 环境或试验床，以更加接近于实际互联网中的应用。

(4) 进一步探讨该系统与其它系统联合起来共同防御 DDoS 的必要性和可行性。任何系统都不能孤立存在，必须与其他系统进行交互，本文介绍的 CPPWQ 也不例外，因此，进一步的研究还包括与其他系统的交互，例如，可以考虑在网络入口出增加过滤规则，使得在网络繁忙时仅仅接收特定于已经加入 CPP 协议的应用。

参考文献

1 Tuomas Aura, Pekka Nikander, and J. Leiwo. DOS – resistant authentication with client puzzles. In Proc. of the 8th International Workshop on Security Protocols, April 2000.

2 Juels, A. and Brainard, J. Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks. Proceedings of the 1999 Network and Distributed System Security Symposium; Reston, VA, USA, 1999. p. 151 – 165.

3 Martin Arlitt, Tai Jin. Workload Characterization of the 1998 World Cup Web Site. Internet Systems and Applications Laboratory, HPL Laboratories Palo Alto, HPL – 1999 – 35(R.1), September, 1999.

4 XiaoFeng Wang, Michael K. Reiter. Defending Against Denial – of – Service Attacks with Puzzle Auctions. IEEE Symposium on Security and Privacy 2003.

5 C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In Advances in Cryptology – Crypto '92, Springer – Verlag, LNCS volume 740, pp. 139 – 147, August 1992.

6 崔筠、周大水，用 Puzzle 方法积极防御 DoS 攻击，通信技术，No 9，2002.

7 Drew Dean and Adam Stubblefield. Using client puzzles to protect TLS. In Proceedings of 10th Annual USENIX Security Symposium, 2001.