基于日志监视主动防御 HTTP 泛洪攻击①

袁 志

(广州大学 华软软件学院，广州 510990)

摘 要：模仿正常访问行为的 HTTP 泛洪攻击较为隐蔽，在消耗网站服务器资源的同时还带来信息安全隐患，提出了一种主动防御方法。用 URL 重写的方法使 Web 日志记录 HTTP 请求的 CookieId 和 SessionId；定时分析 Web 日志，利用 CookieId 和 SessionID 识别用户，根据请求时间特征来识别傀儡主机；对 HTTP 请求进行预处理，拦截傀儡主机的请求。该方法成本低、便于实施，实践证明了其有效性。

关键词：分布式拒绝服务攻击；HTTP 泛洪；网站安全；Web 日志分析；主动防御

Proactive Defense Against HTTP Flood Attacks Based on Watching Web Log

YUAN Zhi

(South China Institute of Software Engineering, Guangzhou University, Guangzhou 510990, China)

Abstract: HTTP flood attacks mimicking normal access behavior are difficult to discovered, it consumes web server's resources and brings hidden danger on information security, a method of proactive defense against HTTP floods is provided. Rewrite URL to record CookieId and SessionId of HTTP requests into Web log; analysis Web log at regular time, identify user according CookieId and SessionId, indentify puppet computers using request time characteristic; process HTTP requests in advance to keep out the requests from the puppet computers. This method is low cost and easy to implement, practice proved its validity.

Key words: distributed denial of service; HTTP flood; website security; Web log analysis; proactive defense

HTTP 泛洪是分布式拒绝服务攻击(Distributed denial of service, DDoS)的一种常见攻击形式，其攻击对象主要是 Web 服务器。当前的 HTTP 泛洪攻击的趋势是模仿正常访问行为，傀儡主机在脚本的控制下向服务器大量发送貌似合法的请求，消耗服务器的 CPU、内存、数据库等资源，同时带来信息安全问题[1]。

当前区分傀儡主机与人的方法可归类为四种：图形迷题法[2]、超链接诱饵法[3]、语义模型法和请求时间特征法[4]。图形迷题法成本较低，但会对正常的用户访问造成困扰；超链接诱饵法和语义模型法可以检测遍历式攻击，但难以检测经过刻意训练的机器请求；请求时间特征法有比较高的成功率，但目前文献没有提供具体的实施方案。

本文采用请求时间特征法，结合 ASP.NET 的技术特点，提出一种基于 Web 日志监视的主动防御方法，该方法成本低、便于实现，通过实验和应用检验，验证了其有效性。

1 防御方法及其实现

本方法包括两个主要环节：一是用 Web 日志监视器定时分析日志，筛选出可疑用户（傀儡主机）；二是设计一个 HTTP 模块，该模块对用户请求进行预处理，采用 URL 重写技术将 CookieId 和 SessionID 写入 Web 日志，并拦截来自傀儡主机的请求。

1.1 Web 日志监视器的设计

Web 日志监视器的作用是通过 Web 日志的定时分析，发现用户请求时间特征，从而识别傀儡主机。

1.1.1 监视器的工作原理

① CookieId 或 SessionId 作为身份标示。Cookie 和 SessionID 都是由服务器端分发给客户端的，客户端

① 收稿时间:2011-08-21;收到修改稿时间:2011-09-15

无法修改，适合用于身份标示[5]。CookieId 相同的请求，认为来自同一用户，SessionID 相同的请求，认为来自同一用户的一个连续时间段。

② 根据请求时间特征识别傀儡主机。由机器发起的请求，其特征是网页停留时间短，刷新频率快，显然不同于人。从日志中可以提取具有这些特征的用户标示（CookieId 和 SessionId），凡持有这些标示的请求将被拦截。

③ 根据 Agent 识别网络蜘蛛。傀儡主机中可能包含著名搜索引擎的网络蜘蛛，网络蜘蛛都会在 Agent 字段中标明蜘蛛身份，通过 Agent 字段检查，避免对网络蜘蛛的错误过滤。

1.1.2 监视器对日志各字段的处理

Web 日志记载用户的请求记录，为分析用户行为提供了基础。在默认情况下，Web 日志中的字段内容很少，本方法需要对 Web 日志定制字段，使其包含（但不限于）如下字段：

#Fields: time cs-uri-query sc-status cs(User-Agent)

以上字段在本方法中的作用：time 字段用于分析用户请求的时间特征；cs-uri-query 字段包含 cookie 和 sessionId，用于用户识别；sc-status 字段用于数据清理，本方法只处理状态码为 200（即请求被正常响应）的记录；cs(User-Agent)字段用于识别网络蜘蛛。

1.2.3 监视器的工作流程

日志监视器在服务器的后台运行，使用一个 Access 数据库来存放分析结果，数据库包含三个表：TLog（请求记录表）、TSpan(每次请求与上次请求间隔时间表)、TDefent(傀儡主机身份标示表)。监视器的工作流程如下：

① 设定分析时间间隔参数 T、傀儡主机特征参数(分析周期内的请求次数上限 MaxCount、平均间隔时间下限 MinSpan)；

② 读取日志文件格式；

③ 依次读日志中的每条记录，将状态码为 200 且与当前时间间隔小于 T 的记录的写入到 TLog 数据表中；

④ 将 TLog 中的记录按 Cookie 和 SessionId 排序，经处理得到用户在时间 T 内的每次请求与上次请求的间隔时间，写入 TSpan 数据表中；

⑤ 读 TSpan 表，统计访问次数与平均间隔时间，将符合傀儡主机特征的 CookieId 和 SessionID 写入 TDefent 数据表中；

⑥ 线程停止时间 T，清空 TLog 和 TSpan，转③。

1.2 HTTP 模块的设计与引用

在 ASP.NET 框架中，HttpModule 是 HTTP 请求的“必经之路”，HttpModule 负责监听请求，在请求被处理之前附加一些信息或者终止请求。ASP.NET 有内置的 HTTPModule，可以通过自定义一个实现接口 IHttpModule 的类，来取代该 HttpModule。

1.2.1 HttpModule 的设计

本方法中，自定义的 HTTPModule 实现两项功能，第一是进行 URL 重写，在 URL 后添加查询参数，使之包含 CookieId 和 SessionId 信息；第二是检查请求上下文中的 CookieId 和 SessionId 与第 2 节中的 TDefent 表进行匹配，如果匹配成功，则拦截该请求。

自定义的 HttpModule 代码结构如下：

```
namespace MyHttpModule
{ public class HttpModule : System. Web. IHttpModule
  { public HttpModule() {}
    public void Dispose(){}
    public void Init(HttpApplication context)
    { context.BeginRequest += new EventHandler (Application_BeginRequest);
      context.AcquireRequestState += new Event Handler(context_AcquireRequestState);}
    public void Application_BeginRequest(object sender, EventArgs e)
    { //如果请求上下文中的 CookieId 和 SessionId 在屏蔽表中，拦截请求
      //如果 URL 查询参数中的 SessionId 在屏蔽表中，拦截请求
      //如果用户是第一次访问网站，请求上下文中的 CookieId 为空，则生成一个唯一的 CookieId，有效期为一个月，发送到客户端，然后放行请求；否则，如果请求上下文中已经含有 CookieId,而 URL 查询参数中不包含 CookieId,则在查询参数中附加 CookieId。}
    public void context_ AcquireRequestState(object sender, EventArgs e)
    { //如果请求上下文中还没有 SessionID，说明是某用户在某时间段内的第一次访问，放行请
```

求；否则，如果请求上下文中已经有 SessionID，而 URL 查询参数中没有，则将 SessionID 附加到查询参数}

}}

1.2.2 HTTPModule 的引用

在网站应用程序中，添加上述模块的引用，然后在 Web.Config 中添加以下配置项：

<httpModules>

<add name="HttpModule" type="MyHttpModule.HttpModule,MyHttpModule"/>

</httpModules>

经过以上配置，所有去往该网站的 HTTP 请求，都要经过 HTTPModule 的处理，傀儡主机的请求被拦截在网站的处理核心之外，网站服务器的 CPU、内存和数据库资源得到保护。

2 实验与应用检验

2.1 实验

设计一个测试网页，用一个按钮实现 Post 测试，用一个超链接实现 Get 测试。日志监视器的 T 设置为 5 分钟，MaxCount 设置为 50，MinSpan 设置为 5 秒。在两个客户端上向网站发送请求，先以低频率发送，均能得到有效响应，后都以高频率发送，5 分钟后都得到请求被终止的反馈。此时查看 Web 日志（摘录如表 1）。

表 1 Web 日志中的数据摘录

Time	Get	cs-uri-query	sc-status
04:01:12	Get	CookieId=634493398364062500 &sid=nj5grsyehyn5x4zj3de3hdj2	200
04:01:36	Post	CookieId=634493925902968750 &sid=mxf0ss55nav31l55z1nbib45	200

从表 1 看到，对于 Get 和 Post 发送的请求，日志中都记载了 CookieId 和 SessionId。查看 Access 数据库，可以看到符合傀儡机器特点的 CookieId 与 SessionId 被计入 TDefent 表。

2.2 应用检验

本方案在一个教育培训机构网站上实施，该网站用 ASP.NET 开发，服务器托管在 ISP 机房，网站的主要价值在于提供了一个论坛，借助 Web2.0 技术使大量的信息由访问者贡献，并从中衍生出大量实际客户，网站上的部分收费信息也为公司贡献了收益。该网站曾因遭到 HTTP 泛洪攻击而多次停机。

按照本文方案，结合网站的具体技术环境，在该网站上进行实施。对照方案实施前 10 天和后 1 个月的服务器性能数据(如表 2)，发现方案实施后，服务器的负载更为平稳，各种资源占用峰值以及峰值持续时间都有大幅降低。

表 2 防御方案实施前后服务器性能比较

	CPU 使用峰值	CPU 使用超 60%时间	内存占用峰值	内存占用超 40%时间	网卡带宽占用峰值	带宽占用超 30%时间
实施前	98%	792 分钟	71%	654 分钟	48%	227 分钟
实施后	65%	32 分钟	34%	63 分钟	16%	0 分钟

3 结语

本文设计了一种主动防御 HTTP 泛洪攻击的方法。该方法的 HTTP 请求预处理模块是结合 ASP.NET 的技术特点实现的，对于其他技术，如 Java 或 PHP，原理上也可用相同的思路来实现。该方法成本低、便于实施，在中小型企业网站上进行了应用检验，实践证明了其有效性。

参考文献

1 孙长华,刘斌.分布式拒绝服务攻击研究新进展综述.电子学报,2009,37(7):1562−1569.

2 Kandula S, Katabi D, Jacob M, Berger A. Surviving organized DDos attacks that mimic flash crowds. USENIX Symposium on Network Systems Design and Implementation, May 2005.

3 Gavrilis D, Chatzis I, Dermatas E. Flash crowd detection using decoy hyperlinks. 2007 IEEE International Conference on Networking, Sensing and Control. April 2007: 466−470.

4 Oikonomou G, Mirkovic J. Modeling Human Behavior for Defense Against Flash-Crowd Attacks. Proc. of IEEE International Conference on Communications. June 2009: 14−18.

5 肖惠,王立华.Web 日志挖掘中的用户识别算.计算机系统应用,2011,20(5):223−226.