目录

OpenWRT 智能路由器的恶意流量拦截技术的研究与实现

Lin Yang

2017 年 9 月 15 日

摘要

家用路由器作为家庭网络流量的汇集点，是现代家庭接入网络的重要设备。人们在无意识地访问钓鱼网站、下载恶意软件时，这些恶意流量也必然会经过家用路由器。如果没有相应安全保护措施，有可能会造成不必要的财产损失。面对这种情况，本文提出了一种路由器级的流量拦截策略。该策略应用在 OpenWRT 嵌入式 Linux 操作系统上，通过开发基于 Netfilter 框架的钩子函数，实现对恶意网络流量的识别和拦截。由于恶意流量种类繁多，本文将以应用广泛的 HTTP 协议为目标对象，在 OpenWRT 操作系统下，实现对钓鱼 URL 的识别和拦截。

1 绪论

1.1 研究背景

家用路由器已经普及，钓鱼网站依旧猖獗，防范钓鱼网站，除了意识上的，还要有技术上的实现。

1.2 国内外研究现状

针对基于 Netfilter 框架的内核流量过滤，国内袁方方【2012】提出了一种内容过滤系统的设计与实现，实现了对特定 URL 及相关网页内容的过滤，但是其对恶意 URL 拦截策略是“串行”的，即只有在判定完 URL 的恶意性后，才能够使得该请求继续进行，延长的一次 HTTP 请求的时间，降低了用户体验，对于突发的大量 HTTP 请求有可能会出现丢包的状况。

鲍娟【2009】中提出了一种嵌入式 Linux 的网络流量监测系统，仅仅实现了流量监测，并未实现流量的拦截功能。

1.3 研究的内容和目的

在上一节国内网研究现状中我们看到，对恶意流量的过滤均采用一种“串行”的方式，即从用户发起请求，到进入流量过滤模块，到继续进行正常的网络通信，这三个步骤是依次进行的。这种方式使得每次的网络请求便都至少增加了一个流量过滤模块的处理延迟，降低了用户体验，如果流量过滤模块没有得到很好的设计，甚至会导致内核崩溃。面对这种情况，本文提出了一种“并行”的解决方案，力求减少路由器设备对流经网络流量的操作：用户在发起 HTTP 请求时，我们仅仅在内核进行“登记”，真正的拦截将发生在 HTTP 响应包上，而在请求和响应的时间差内，我们对该请求 URL 的恶意性进行判断。这样便引出了本文要研究的几个问题：

1. **如何识别 HTTP 请求中的 URL**
   通过在 POST 点和 PRE 点上，注册定制的 Hook 函数，对 HTTP 请求头中的 host 字段中的内容进行记录。

2. **如何判断识别出的 URL 的恶意性**
   通过 Netlink 技术，将提取到的 URL 等信息传递到用户空间程序，用户空间程序可根据本地黑白名单，对该 URL 的恶意性作出判断。

3. **如何将 HTTP 响应包与 HTTP 请求包关联起来**
   对内核 conntrack 结构体进行轻微改造，添加一个自定义指针，该指针指向我们在内核中开辟的内存区域，通过定制的关联策略，将 HTTP 请求和响应对应起来。

4. **如何实现恶意流量的拦截**
   根据之前的恶意性判断，用户空间进程会发送 Netlink 消息给内核，内核根据消息内容，找到之前的自定义指针，根据指针指向的地址，及预设好的结构找到相应 HTTP 请求和响应包的位置，进行标记。当流量走过钩子点时，钩子函数会根据这个自定义指针找到相应标记位，根据标记位的值，对流经该钩子点的包裹采取相应的动作（通过或拦截）。而且这种拦截无需拦截这个 TCP 链接上的所有 TCP 包，只需将包含 HTTP 响应头的 TCP 包拦截掉，则该 HTTP 响应包便无法成组。

5. **如何将这些功能应用到 OpenWRT 智能路由器上**
   通过对 OpenWRT 操作系统的研究，实现对内核模块的交叉编译，开发针对 OpenWRT 平台的内核模块，实现上述所讲功能。

综上所述，本文除了继承了之前研究工作的模块化程序设计优势，还将“串行”的解决方案，通过一定策略设计，转变为一种“并行”的解决方案。使

得用户的正常网络请求，与对该请求恶意性的判断并行地进行，力求降低延迟，提升用户体验。

1.4 论文组织结构

本文包含以下几个部分：

第一节主要阐述了本文研究的意义和背景，国内外研究动态，论文研究的内容和目的，最后介绍了论文的结构

第二节主要讲解了恶意流量过滤所涉及的技术。其中有，Netfilter 框架下的相关技术介绍，包括对流量进行识别和拦截操作的 Hook 函数的开发，实现用户空间和内核空间通信的 Netlink 技术，以及实现对每一个 TCP 连接进行跟踪的 Conntrack 技术等。还包括对 HTTP 协议的介绍，以辅助策略设计。为后面的恶意流量过滤提供了理论基础。

第三节主要讲解了在 OpenWRT 操作系统上的恶意流量过滤系统的设计与实现。

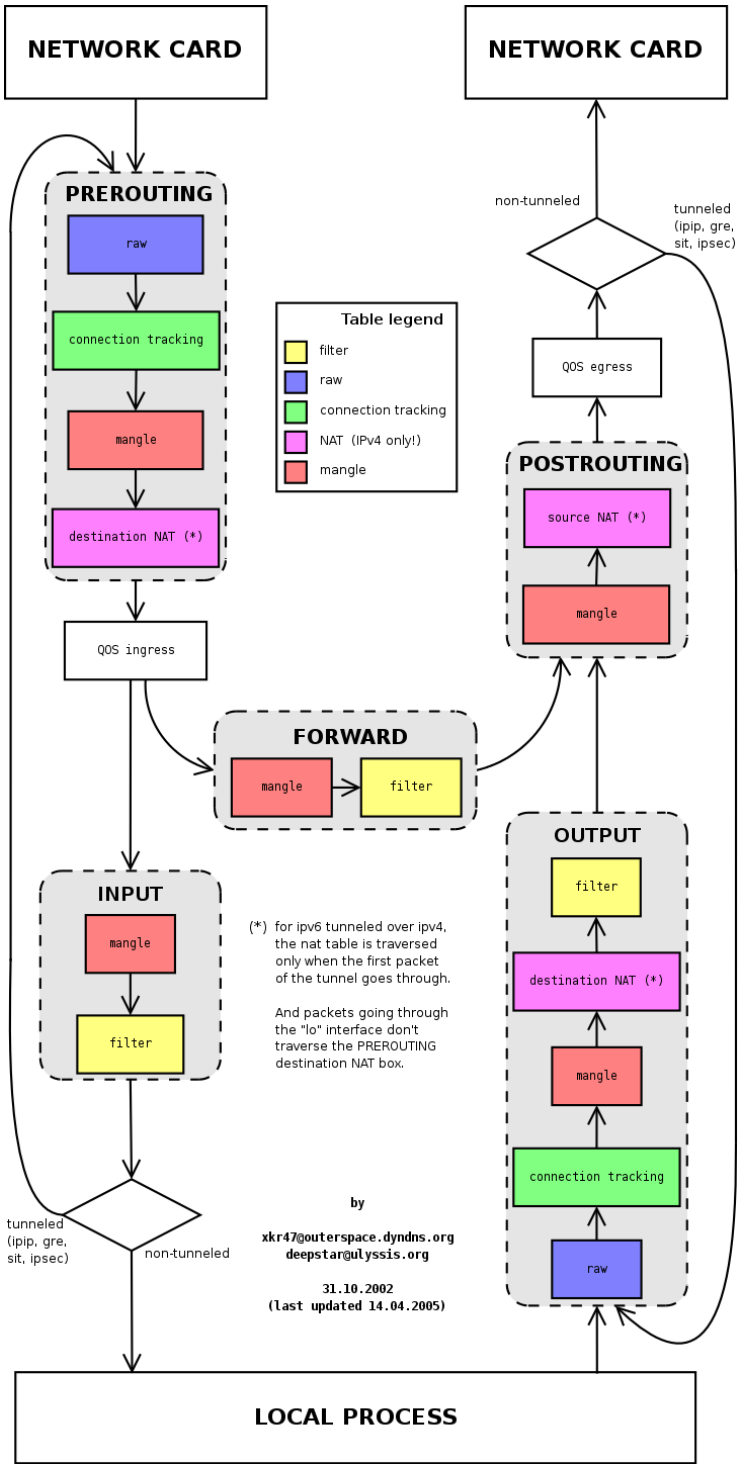
第四节主要讲解了基于 OpenWRT 的恶意流量过滤系统环境的搭建，测试和分析。

第五节对本文的工作进行了总结和展望。

2 恶意流量过滤技术相关技术介绍

2.1 Netfilter 框架

Netfilter 是 Linux2.4.x 版本引入的一个子系统，它作为一个通用的、抽象的框架，提供了一整套 hook 函数管理机制，使得网络地址转换（NAT）、数据包过滤和基于协议类型的连接跟踪成为了可能。Netfilter 框架对数据包的处理流程如下图：


NETWORK CARD
NETWORK CARD
PREROUTING
raw
connection tracking
mangle
destination NAT (*)
QOS ingress
Table legend
filter
raw
connection tracking
NAT (IPv4 only!)
mangle
non-tunneled
tunneled (ipip, gre, sit, ipsec)
QOS egress
POSTROUTING
source NAT (*)
mangle
FORWARD
mangle
filter
OUTPUT
filter
destination NAT (*)
mangle
connection tracking
raw
INPUT
mangle
filter
(*) for ipv6 tunneled over ipv4, the nat table is traversed only when the first packet of the tunnel goes through.
And packets going through the "lo" interface don't traverse the PREROUTING destination NAT box.
by
xkr47@outerspace.dyndns.org
deepstar@ulyssis.org
31.10.2002
(last updated 14.04.2005)
tunneled (ipip, gre, sit, ipsec)
non-tunneled
LOCAL PROCESS

2.1.1 Hook 函数开发

2.1.2 Netlink 通信机制

2.1.3 Conntrack 技术

2.2 HTTP 协议介绍

2.2.1 HTTP 协议基本介绍

2.2.2 HTTP 连接管理

HTTP definitive Guide Chap4

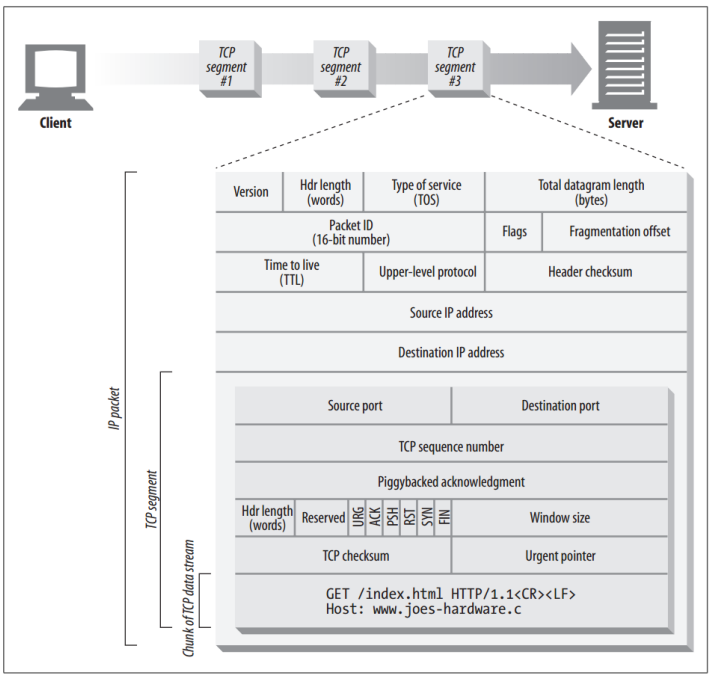


Figure 4-4. IP packets carry TCP segments, which carry chunks of the TCP data stream

3 OpenWRT 恶意流量过滤系统的设计与实现

3.1 OpenWRT 系统及开发介绍

3.2 URL 请求识别系统的设计

3.3 Conntrack 系统拓展设计

4 环境搭建、测试与分析

5 总结与展望