文章编号:2095-6134(2017)06-0759-12

一种基于无线路由器的 IoT 设备轻量级防御框架*

严志涛[1,2],方滨兴[3,4],刘奇旭[1,2†],崔 翔[1,2,3]

(1 中国科学院大学网络空间安全学院,北京 100049; 2 中国科学院信息工程研究所,北京 100093;
3 北京邮电大学,北京 100876; 4 东莞电子科技大学电子信息工程研究院,广东 东莞 523808)

(2016 年 11 月 29 日收稿; 2017 年 2 月 22 日收修改稿)

Yan Z T, Fang B X, Liu Q X, et al. A wireless router-based lightweight defense framework for IoT devices[J]. Journal of University of Chinese Academy of Sciences, 2017,34(6):759-770.

摘 要 目前 IoT(Internet of things, 物联网)设备安全问题很多,然而由于 IoT 设备自身限制(嵌入式系统,资源紧张),传统 PC 的保护手段已经不再适用。提出一种基于无线路由器的 IoT 设备轻量级防御框架 WRGuardian(wireless router guardian),利用家用无线路由器在网络流量的掌控能力和拓扑结构优势,从被动防御和主动防御两个方面入手,及时监测并阻断目前针对 IoT 设备的主要攻击行为,同时定期扫描检测安全问题并修复。该框架无需外部硬件或者修改设备原有系统,降低了部署难度和成本,有利于后期推广。实验结果显示 WRGuardian 能够有效对抗针对 IoT 设备弱口令、命令注入等主要攻击手段,且能排查修复已知风险,是一种低成本可行的轻量级防护方案。

关键词 无线路由器; IoT 设备; 安全防护; 网络流量

中图分类号:TP393 **文献标志码**:A **doi**:10.7523/j.issn.2095-6134.2017.06.013

A wireless router-based lightweight defense framework for IoT devices

YAN Zhitao[1,2], FANG Binxing[3,4], LIU Qixu[1,2], CUI Xiang[1,2,3]

(1 *School of Cyber Security, University of Chinese Academy of Sciences, Beijing* 100049, *China*;
2 *Institute of Information Engineering, Chinese Academy of Sciences, Beijing* 100093, *China*;
3 *Beijing University of Posts and Telecommunications, Beijing* 100876, *China*;
4 *Institute of Electronic and Information Engineering, Dongguan University of Electronic Science and Technology, Dongguan* 523808, *Guangdong, China*)

Abstract It is well known that IoT (Internet of things) devices are vulnerable and can be easily intruded by attackers. However, traditional protection methods for PCs are no longer suitable for IoT devices. In this work, we design a router-based lightweight defense framework WRGuardian (wireless router guardian) which uses the router's network traffic controllability and computing capacity to protect IoT devices. It will monitor and block the attack behaviors to IoT devices, and it will detect and fix the security issues by simulating attacks. Because there is no requirement of additional security hardware for the IoT devices, this protection framework has a low cost, and it is

* 国家重点研发计划(2016YFB0801604)、国家自然科学基金(61303239)和广东省产学研合作项目(2016B090921001)资助
† 通信作者,E-mail:liuqixu@iie.ac.cn

convenient to deploy and beneficial for promotion. Our experimental results show that WRGuardian is feasible and protects IoT devices from main attacks. It is an effective lightweight solution.

Keywords wireless router; IoT device; protection; network traffic

随着IoT设备的发展,各大厂商发布了越来越多的IoT设备来丰富我们的生活。智能路由器、智能摄像头、智能手环、智能插座等各种智能化设备的大量推出给我们的生活带来了极大便利,Gartner公布的预测数据显示[1],到2020年,全球联网设备将达到250亿部,远超过那时的全球人口总数。然而,IoT设备的高速发展也伴随着大量的安全问题,弱口令、漏洞层出不穷,影响着IoT设备的安全。

当前,通过IoT设备的各种安全问题,攻击者们已经开始了恶意代码大规模控制的尝试,2012年3至12月份,匿名安全研究者通过扫描各种网络设备并植入恶意代码,非法入侵全球大约42万台联网设备,获取抽样数据,发表了一份名为"2012互联网普查"的报告[2],并将此僵尸网络称为"Carna"僵尸网络。2014年12月,一个名叫"Lizard Squad"的组织使用基于路由器构建的僵尸网络[3]向索尼公司和微软公司的在线游戏平台PSN与Xbox Live发动DDoS攻击,导致这两个平台长时间无法访问。2016年10月22日,美国域名服务器管理服务供应商Dyn遭遇DDoS攻击,从而导致包括Twitter、GitHub、PayPal等在内的许多知名网站在美国东海岸地区无法访问[4],被媒体称为"美国东海岸断网事件"。根据Flashpoint公司的调查结论[5],此次攻击与感染IoT设备为主要目标的Mirai僵尸网络有关。Mirai[6]是一款开放源码的僵尸程序,其目标主要是弱口令的IoT设备,包括路由器、机顶盒、摄像头等,主要用于DDoS攻击。与其类似的还有Lightaidra[7],一款在GitHub上开源的僵尸程序,其使用IRC协议[8]作为与C&C(Command and Control,命令与控制)服务器通讯的协议,支持弱口令扫描和渗透以路由器为代表的IoT设备,构建僵尸网络并可支持DDoS攻击操作。

除家用路由器被利用构建僵尸网络外,智能冰箱、智能洗衣机等也可以成为恶意代码攻击的目标。2014年Proofpoint公司发现了可能是史上首个物联网僵尸网络的攻击[9],参与攻击的设备包括智能冰箱和智能电视。

这些感染IoT设备的僵尸程序往往并没有使用高深的漏洞,多数是借助厂商预置的默认弱口令轻松地获取设备的控制权,例如已经开源的Mirai僵尸程序中就内置了60余组扫描口令用户暴力破解获取设备的控制权。Krebsonsecurity整理了这些口令组合对应的厂商情况[10],部分内容如表1所示。

表1 Mirai源码中内置的部分口令组合情况

Table 1 Usernames and passwords included in the Mirai source code

用户名	密码	相关厂商产品
admin	123456	ACTi IP Camera
root	anko	ANKO Products DVR
root	pass	Axis IP Camera, et. al
root	vizxv	Dahua Camera
root	hi3518	HiSilicon IP Camera
root	admin	IPX-DDK Network Camera

由于IoT设备多数是嵌入式设备,看似简单的默认弱口令问题修复十分不便,需要厂商发布固件更新后用户自行刷新固件来完成。除更新操作不便外,新固件的制作还将影响到厂商新产品的发布计划,增加开发测试难度,获得的收益却几乎为零,多数厂商固件更新动作迟缓甚至不为老产品发布更新。为推动IoT设备厂商重视产品的安全,在"美国东海岸断网事件"发生后的2016年11月15日,美国国土安全部发布《保护物联网策略准则》[11]呼吁厂商在设计生产IoT设备时肩负起保障安全的责任。

综上所述,当前市场上的IoT设备安全问题众多,然而防护手段却极其有限,只能等待厂商发布新版本的固件解决安全问题。由于IoT设备资源性能限制、部署成本等原因,在设备内部安装防护软件或者在外部部署额外的安全硬件等防护方案都不具有可行性或者可推广性,而设备厂商发布固件更新动作迟缓,IoT设备亟需一款轻量级的可推广的防护方案。

因此,针对以上IoT设备的安全困境,本文提出一种轻量级的防御框架来保护IoT设备。在无需加装外部硬件设备或者修改IoT设备系统设置的前提下,利用IoT设备广泛接入的家用无线路

由器在网络流量的掌控能力和拓扑结构优势,来保护接入的各类设备。

本文的主要贡献如下:

1) 设计了一个基于无线路由器的 IoT 设备轻量级防御框架 WRGuardian,在无需外部硬件或者修改设备原有系统的前提下利用无线路由器的特点从环境入手保护 IoT 设备,从被动防御和主动修复两方面对抗攻击者主要使用的弱口令和命令注入等漏洞的攻击。

2) 实现了防御框架的原型系统,针对多种不同的 IoT 设备,设置相应的实验环境,在功能上测试验证了该防御框架对抗主流攻击和检测修复已知风险的能力,在性能上测试了该框架对原有网络吞吐能力的影响情况。

1　相关工作

目前学术界和工业界并没有明确提出针对 IoT 设备的专用保护方案,IoT 设备往往和 PC 一起被归为一类进行处理保护。主要相关工作如表 2 所示。

表 2　相关工作比较

Table 2　Related works

研究方向	保护方案	作者单位	存在问题
流量检测防御	基于 SDN 的在线流量异常检测方法[12]	解放军理工大学	对部署设备的资源、性能要求较高,不适合在 IoT 设备上部署
	基于特征选择的轻量级入侵检测系统[13]	中国科学院计算所	
	基于流量信息结构的异常检测[14]	清华大学	
	软件防火墙	—	多数 IoT 设备无法安装第三方软件
	硬件 IDS/IPS	—	部署成本高,难以推广
漏洞扫描修复	Acunetix Web Vulnerability Scanner[15]	Acunetix	主要针对 Web 漏洞,且只提示漏洞信息,无自动尝试修复功能
	IBM Security AppScan[16]	IBM	
	Nessus Vulnerability Scanner[17]	Tenable Network Security	只提示漏洞信息,无自动尝试修复功能

在网络流量检测防御方面,国内外学者提出诸多理论和方法。文献[12]提出一种基于软件定义网络的在线流量异常检测方法,使用主成分分析方法检测网络中异常的流量。文献[13]根据 3 种特征模式分类比较基于特征选择的入侵检测系统,总结其优缺点及适用条件。文献[14]提出基于流量信息结构支持向量机的异常检测算法。此外,工业界推出软件防火墙、硬件 IDS/IPS 等安全产品对 PC、服务器等设备提供通用性的保护。

在漏洞扫描修复方面,工业界开发实现的产品相对较多。Acunetix 公司推出的 Acunetix Web Vulnerability Scanner[15]可以扫描检测目标设备上的 Web 漏洞,提出漏洞信息,但无修复功能。IBM 公司推出的 IBM Security AppScan[16]可以扫描 Web 应用和手机应用的安全漏洞,提出修复建议。Tenable Network Security 公司推出的 Nessus Vulnerability Scanner[17]可以扫描目标设备的系统漏洞,生成检测报告,提示修复建议。

在此前的研究中,各类保护方案主要是面向攻击方式提出针对性的解决方案,主要是由 PC、服务器上的防护软件或者是专用外部安全硬件提供防御功能。若采用在 IoT 设备上安装第三方防护软件的方案,由于目前的 IoT 设备多数采用嵌入式系统,资源性能有限,在没有厂商配合的前提下安装第三方防护软件基本不可能。此外由于平台、型号众多,安装第三方防护软件开发部署成本也让此方案不具可行性。若采用专用外部安全硬件的方案,由于 IoT 设备的使用场景大多在家庭领域,多数家庭用户出于成本上的考虑并不会购买专用外部安全硬件,而目前曝光的 IoT 设备被大规模恶意利用的案例中被利用的设备往往却是家庭领域的设备,此方案部署成本较高,推广难度较大。

2　WRGuardian 设计思路

WRGuardian 防御框架的设计工作平台是有一定计算资源富余的家用无线路由器(内存 64 MB 以上,闪存 8 MB 以上,一般百元价位的无线路由器即可满足此条件)。此类价位无线路由器往往带有简单的安全防护机制(如端口过滤、MAC 地址过滤等),但只能方便在用户发现威胁后进行简单的事后防护,没有办法自动发现并响应威胁。目前 IoT 设备的大规模恶意利用的渗透方式以厂商预置弱口令暴力尝试为主,恶意利用以 DDoS 攻击和发送垃圾邮件为主。以上 IoT 设备的安全威胁仅仅依赖用户及时发现并在无线路由器中设置是不现实的。基于以上 IoT 设备安全问题现状,结合无线路由器的性能情况,我们设计了 WRGuardian 防御框架。考虑到家用无线路由器的资源与性能限制,防御框架的设计追求轻量

化,主要体现在主程序文件数量和体积小、运行不依赖其他程序、方便部署与移植等。在功能上,防御框架利用无线路由器从被动防御和主动防御两个方面保护接入的 IoT 设备。被动防御机制主要是针对攻击行为发生的时候及时发现并阻断,让攻击操作失败;而主动防御主要是从攻击行为发生之前考虑,定期扫描接入设备的安全情况,若发现问题可尝试修复,无法修复则通知风险预警模块及时提醒用户处理。

WRGuardian 防御框架的主要模块包括流量处理模块、风险预警模块、主动防御模块等。其中,流量处理模块主要通过 MitM(Man-In-the-Middle)中间人技术对引入的 IoT 设备网络流量进行分析判断,判定其是否符合设置的要求或者是否存在恶意代码,进行处置后将清洗过的流量从原出口输出;主动防御模块主要负责对接入无线路由器的设备进行已知漏洞的扫描处置工作;风险预警模块主要负责在收到其他模块的风险消息后及时采取多种方式向用户发出预警,提示用户及时处理安全问题。整个框架模型如图 1 所示。

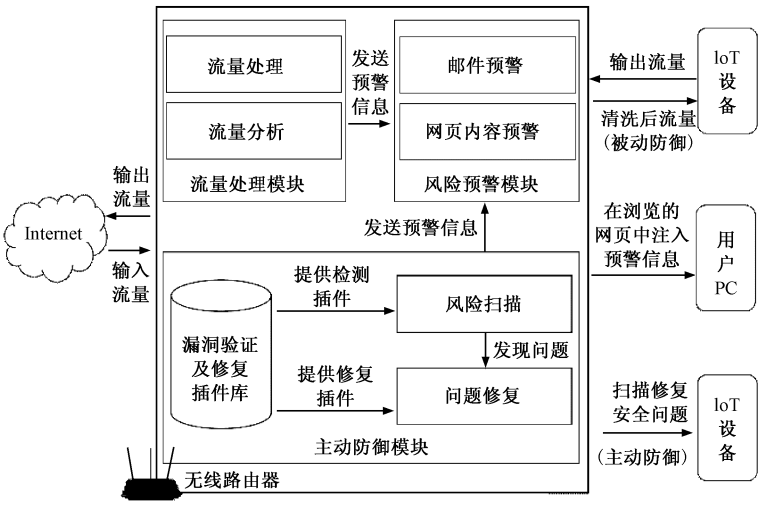

图 1 WRGuardian 防御框架模型

Fig. 1 Model of WRGuardian protection framework

WRGuardian 防御框架分别从攻击行为处置和 IoT 设备安全风险排查两个角度入手,提供被动防御和主动防御两个保护机制保护接入的 IoT 设备。

被动防御主要是面向攻击行为进行监测,对流经路由器与被保护设备相关的网络流量进行监测,由于无线路由器的性能限制,加之目前主流的 IoT 设备与外界交流的协议较为简单,因此可对被保护设备配置流量访问规则,时刻监测其流量情况,若发现协议异常或者流量异常可立即阻断后续网络流量,并向用户预警安全风险。网络流量监测流程如图 2 所示。

主动防御主要是从 IoT 设备安全风险排查方面入手,通过从漏洞验证(PoC, Proof of Concept)及修复方案插件库提取检测处置插件,定期对接入的 IoT 设备进行安全风险扫描,发现安全风险后将尝试修复,无法修复的则立即向用户发出预警,告知该设备的安全风险提醒用户注意。具体的工作流程图如图 3 所示。

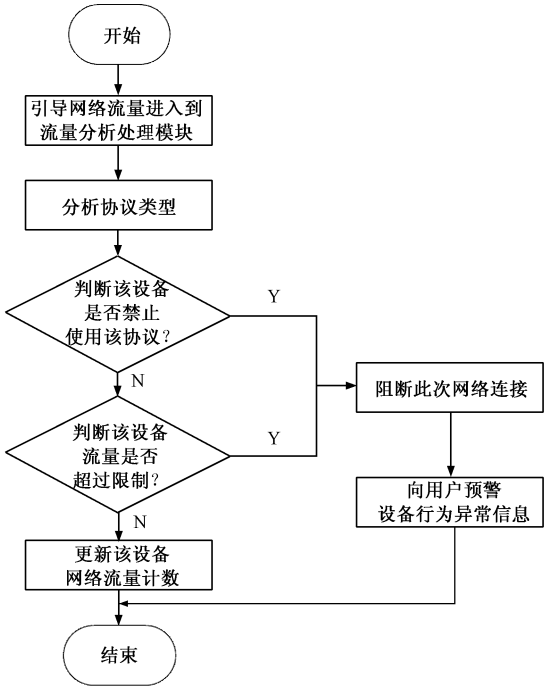

图 2 网络流量监测流程图

Fig. 2 Process of network traffic monitoring

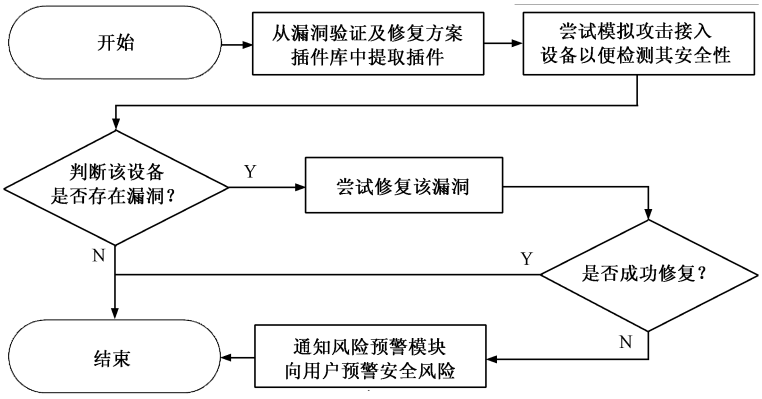

图 3　主动防御工作流程图

Fig. 3　Process of active defense

3　WRGuardian 模块实现

基于上文提出的设计思路,我们实现了 WRGuardian 防御框架原型系统。原型系统采用模块化的设计思路进行开发实现,主要由流量处理模块、主动防御模块、风险预警模块等组成。

3.1　开发语言及相关技术

WRGuardian 防御框架各模块开发语言及第三方库使用情况如表 3 所示。

表 3　各模块开发语言及第三方库使用情况

Table 3　Programing languages and libraries used in modules

	模块	开发语言	第三方库
1	流量处理模块	C	L7-filter[18] 正则表达式库
2	主动防御模块	C, Lua[19]	Lua 5. 2. 4 LuaSocket 3. 0-rc1[20]
3	风险预警模块	C	—

其中,防御框架主体采用 C 语言开发,使用 uClibc-0. 9. 33. 2[21] 交叉编译成相应平台的二进制程序。流量处理模块引入 L7-filter[18] 软件的正则表达式库帮助识别流量的应用层协议信息。

L7-filter 是一款 Linux 下应用层数据包协议识别分类软件,其使用正则表达式匹配特征的方式对数据包应用层协议进行识别分类。L7-filter 提供一个协议特征模式库,根据各个协议的流量特征结合 RFC[22] 说明,总结归纳出各个协议的特征正则表达式(20090528 版本支持识别 114 种协议),存放在该模式库中。由于 L7-filter 的安装使用需要对 Linux 系统底层进行改造(给 Linux 内核和 iptables 安装补丁),且对内核版本等有一定的要求,不适合本文讨论的无线路由器环境,因此流量处理模块只采用其成熟的正则表达式库进行应用层协议的识别工作。

主动防御模块的插件系统引入开源的 Lua-5. 2. 4[19] 解释器引擎,为方便检测处置插件的开发,同时集成开源的 LuaSocket-3. 0-rc1[20] 作为 Lua 解释器的扩展库,该扩展库封装了常见的网络操作,降低了插件开发的难度。

Lua 是巴西里约热内卢天主教大学的研究小组设计开发的一个轻量化的脚本语言,主要用于嵌入到已有应用程序之中为其提供灵活的扩展能力,其解释器由标准 C 语言编写,可以灵活地编译运行在多种指令集和操作系统之上,并可轻松地与 C/C + +代码相互调用。相较于功能强大的 Python 语言,Lua 解释器的体积与运行速度更加有优势,更加适合 WRGuardian 防御框架的运行环境(嵌入式设备、资源紧张)。

3.2　流量处理模块

流量处理模块主要通过 MitM 中间人技术对引入的 IoT 设备网络流量进行分析判断,判定其是否符合设置的要求或者是否存在恶意代码,进行处置后将清洗过的流量从原出口输出。目前针对主流 IoT 设备的攻击大多是利用 IoT 设备的网络资源和计算资源获取利益,因此,流量分析处理模块的主要工作是监控接入设备的网络通讯情况,发现异常协议或者异常流量及时阻断,让攻击者无法利用。这就需要对网络流量进行协议分析和流量统计工作,评估安全风险,判断是否需要对该流量进行处理。

由于当前无线路由器资源和性能的限制,本模块目前采取基于自定义规则的流量分析处理方法,即在每个设备接入路由器之时,系统自动给其

初始化一个默认网络行为规则,将其允许使用的网络协议和禁止使用的网络协议明确在规则中,同时设置其单位时间内网络流量最大使用量,此外,用户也可以根据具体情况更改这些规则。基于以上设置的设备网络行为规则,流量处理模块即可监测设备的异常网络行为,及时处理。

流量处理模块主要由数据包预处理子模块、协议识别子模块、设备规则评估子模块、数据包内容评估处理子模块4部分组成。网络数据包首先将进入数据包预处理子模块进行预处理,重组出较为完整的应用层数据,然后进入到协议识别子模块中,识别出其应用层协议情况,获取到协议信息后,设备规则评估子模块将根据该设备的规则设置情况判断是否放行数据包,若放行,相关数据包将交给数据包内容评估处理子模块,该子模块将会根据情况判断是否需要对内容进行处理,最后,数据包将会转发给接收设备。

1)数据包预处理子模块

数据包预处理子模块主要负责重组缓存多个数据包以便还原其中的应用层数据,同时统计相应设备的网络流量。目前大量应用层数据由于体积较大等原因会分割成多个数据包发送,应用层协议特征可能分布在多个数据包内,直接识别准确性较低。

为了能够提高识别准确度,需要还原出较为完整的应用层数据内容再进行识别操作,具体做法是对于每个连接的前几个数据包进行缓存,然后重组其中的应用层数据,再交给协议识别子模块进行下一步的操作。

2)协议识别子模块

协议识别子模块主要负责识别出数据包中的应用层协议。该模块采用L7-filter的协议识别思想,即基于正则表达式(Regular Expression)匹配数据中的特征识别出所使用的应用层协议。

在接收到预处理后的应用层数据后,协议识别子模块将使用不同协议的特征正则表达式去匹配,一旦匹配到,就标记其协议,并将识别结果发送给设备规则评估子模块进行接下来的操作。如匹配HTTP协议的正则表达式如下所示:

```
http/(0\.9|1\.0|1\.1)[1-5][0-9][0-9][\x09-\x0d-~]*(connection:|content-type:|content-length:|date:)|(get|post|head|put|delete)[\x09-\x0d-~]*http/[01]\.[019]
```

该表达式根据HTTP协议头的特点匹配了协议头部的关键词HTTP、状态码、Connection、Content-type、Content-length等,若匹配成功即可大概率判断其应用层协议为HTTP协议。

3)设备规则评估子模块

设备规则评估子模块主要负责检查该流量相关设备设置的规则,根据协议识别结果判断是否允许该协议流量通过。

4)数据包内容评估处理子模块

数据包内容评估处理子模块主要负责数据包内容的修改处理操作。该子模块在收到相关指令后将可以对即将放行的数据包内容进行修改。

3.3 主动防御模块

主动防御模块主要负责对接入无线路由器的设备进行已知漏洞的扫描处置工作。通过使用主动防御模块对接入设备模拟攻击操作,检测操作的成功性,以判断其是否存在安全风险;同时,在掌握攻击原理之后,相应的防御修复方案也可快速提出,做到从攻击入手提升防护能力的效果。

结合以上以模拟攻击验证安全问题并修复的思想,主动防御模块采用插件化的模式进行构建,将收集到的漏洞根据原理编写成包含漏洞验证(PoC)代码及修复方案的检测处置插件,然后使用这些插件完成检测处置操作。当有新的设备漏洞被发现,即可将相应的PoC代码及修复方案封装成插件,追加到漏洞验证及修复方案插件库中。

该模块首先从漏洞验证及修复方案插件库提取检测处置插件,然后使用插件扫描检测接入的设备是否存在安全问题,若存在则主动尝试修复该问题,无法修复的则通知风险预警模块向用户发出安全预警。

整个主动防御模块工作效果的好坏主要是由漏洞验证及修复方案插件库的插件质量来决定,插件库需要能够定期更新扩展,以便覆盖到最新的漏洞。为此,主动防御模块采用插件化的设计思想,引入Lua解释器,提供Lua脚本的解释执行能力,为插件库的扩展提供基础。在Lua解释器的支持下,每个检测处置插件只需要封装成Lua脚本的形式,即可被主动防御模块解释执行,而不需要修改主动防御模块的代码。插件库的插件需要提供一个入口函数action,作为主动防御模块的起始调用点。为增强插件对多种环境的适应能力,入口函数将提供一个参数args,将各种调用参数以空格符分隔整合成一个字符串作为该参数传

入,入口函数首先将 args 参数中的各个调用参数取出,然后执行相应操作。

一个 Telnet 弱口令检测处置插件的入口函数示例。

输入:*args*,一个将目标 IP、目标端口、弱口令字典以空格符分隔的字符串。

输出:*return_msg*,检测处置结果的字符串。

```
    //Entry point
1   Function action(args)
2       If args is not null Then
3           args_list ← string.split(args, "")
4           target_ip ← args_list[1]
5           target_port ← args_list[2]
6           pwd_list ← args_list[3:]
7       End If
8   return_msg ← "No vulnerability"
9   For i from 1 to len(pwd_list)
10          res = test_telnet_weak_password(target_ip, target_port, pwd_list[i])
                //test_telnet_weak_password() tests telnet service using weak password and return the result
11      If res is true Then
12          fix_res = try_to_fix_vulnerability(target_ip, target_port, pwd_list[i])
                //try_to_fix_vulnerability() tries to fix the vulnerability and return the result
13          If fix_res is true Then
14              return_msg ← "Found vulnerability and fixed"
15          Else
16              return_msg ← "Found vulnerability and failed to fix"
17          End If
18          Break
19      End If
20  End for
21      Return return_msg
22  End
```

3.4　风险预警模块

风险预警模块主要负责在收到其他模块的风险消息后及时采取多种方式向用户发出预警,提示用户及时处理安全问题。风险预警模块的目标是让用户尽可能早地得知当前环境中存在安全风险的设备,以便尽早处理消除安全隐患,为此,该模块提供 2 种预警方式。

第 1 种方式是当风险预警模块收到其他模块的风险消息后向指定邮箱发送风险预警邮件,提示用户某设备存在安全风险,请用户尽快修复。此方式作为传统的风险提示方式实现简单,但是由于目前用户使用电子邮件的方式还是以定时查看为主,对大部分用户而言,邮件通知无法实时被获取查看,预警效果较差。

第 2 种方式是利用中间人技术修改用户访问的网页内容,将预警信息加入到该网页中,直到用户完成修复操作。此方式需要风险预警模块和流量处理模块联动,在接收到风险消息后,风险预警模块将会向流量处理模块发送处理请求,流量处理模块将会对用户访问的网页进行修改,在页面中加入预警信息,让用户能够第一时间知晓设备安全隐患,及时处理。

此方式主要针对使用 HTTP 协议的 Web 页面进行修改,由于目前大量设备使用基于 HTTP 协议的自定义协议作为通讯协议,为了不影响设备的正常通讯功能,漏洞预警模块将让流量处理模块在确保是 Web 页面的前提下修改网页内容,在 < title > 标签中修改网页标题或者在 < body > 标签中加入预警信息。

为了在 HTTP 页面中插入预警信息,还需要对 HTTP 报文常用的分块传输编码(Chunked Transfer Encoding)和 Gzip[23] 压缩等技术进行预处理和还原。其中 Chunked 编码是 HTTP1.1 协议中引入的编码方式,定义在 RFC2616[24] 中,其允许 HTTP 服务器将传输的数据分成多个块,然后以一个或多个块分块发送,不像之前的 HTTP 响应包需要定义字段 Content-Length 来表明数据长度,采用分块传输编码技术的服务器可以在不知道需要发送数据的总大小的前提下先发送数据,适合动态生成内容的 Web 服务器使用。

由于以上技术的存在,HTTP 报文内容修改需要进行相应处理,具体流程如下:

1)从数据流中拦截 TCP 数据包,判断是否包含 HTTP 响应报文;

2)从报文中获取 HTTP 协议头信息,判断该报文是否采用分块传输编码、Gzip 压缩等技术;

3)根据 HTTP 协议头的 Content-Length 字段或者分块传输的结束标记,判断是否需要继续接

收数据包直到完整的 HTTP 报文被接收到;

4）在报文内容中搜索 HTML 协议特征标签,判断 HTTP 报文内容是否是 Web 页面,确认后在 body 标签后插入预警信息;

5）根据编码技术的使用情况,修正 HTTP 报文协议头的 Content-Length 或者所在分块的数据长度,然后根据压缩技术的使用与否判断是否重新压缩修改后的报文内容;

6）放行修改后的数据包。

4 实验与结果分析

为测试验证 WRGuardian 防御框架的实际效果和相关的性能情况,我们选取不同厂商、不同型号的无线路由器部署运行 WRGuardian 防御框架,部署运行的方式是先开启无线路由器的 Telnet 或 SSH 管理功能,使用 Shell 指令将框架主程序上传到无线路由器中运行,然后再关闭路由器的 Telnet 或 SSH 功能。实验所使用的无线路由器信息及运行情况如表 4 所示。框架部署运行后,我们进行了一系列实验,分为功能测试和性能测试两个部分。其中,功能测试主要是测试评估防御框架能否对目前主流的 IoT 设备攻击方式进行防御拦截,性能测试主要是测试防御框架的部署对原有网络吞吐能力的影响情况。

表 4 WRGuardian 防御框架在不同型号路由器中部署运行结果

Table 4 Results of WRGuardian running on different platforms

	品牌	型号	指令集	系统	运行情况
1	ASUS	RT-AC66U	MIPS32 74 K series	BusyBox 1.17.4	✓
2	PHICOMM	PSG 1208	MIPS32 24 K/E series	BusyBox 1.12.1	✓
3	NETGEAR	WNDR 4300	MIPS32 74 K series	OpenWRT 15.05.1	✓
4	—	Virtual Machine	X86	OpenWRT 15.05	✓

4.1 实验环境

功能测试和性能测试的实验环境如图 4 所示。

其中,无线路由器 NETGEAR WNDR4300(安装系统为 OpenWRT 15.05.1)作为防御框架部署的平台,安装了 WRGuardian 防御框架原型系统。被保护的 IoT 设备将通过有线或无线的方式接入到该无线路由器上。一台 PC(操作系统为 Linux Mint 17.2 Cinnamon 64 bit, CPU 为 Intel Core i7-3770 3.40 GHz, 内存为 8 GB)接入到该无线路由器上作为实验操作平台。

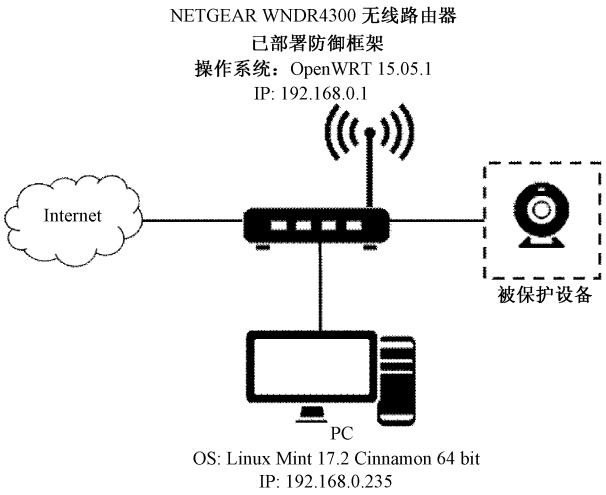

图 4 实验环境拓扑图

Fig. 4 Topology of the experiments

4.2 功能测试

功能测试部分主要测试 WRGuardian 防御框架对目前主流 IoT 设备攻击方式的应对情况,根据防御框架的功能特点,功能测试部分主要分为以下 2 个方面:

1）被动防御能力测试

对被保护设备相关攻击行为的防御能力;

2）主动防御能力测试

主动防御模块的漏洞排查修复能力。

4.2.1 被动防御能力测试

对被保护设备相关攻击行为主要可分为外界对设备的渗透、设备被渗透后发动攻击 2 种。因此,对此类攻击行为的防御能力测试也从这 2 个方面展开。

为了让实验测试场景更接近真实的案例,我们使用开源的两个僵尸网络程序(Lightaidra[7] 和 Mirai[6])模拟发动攻击,防御框架运行在安装了 OpenWRT 15.05.1 系统的 NETGEAR WNDR4300 无线路由器上,让被保护设备接入到该路由器上。

同时考虑到 IoT 设备系统平台、版本众多,每项实验的被保护设备尽可能选取不同型号或者不同系统版本的设备,增大防御框架保护设备的覆盖面,让测试更加全面。具体测试结果如表 5 所示。

实验 1 ~ 3 使用弱口令扫描和漏洞利用渗透两种方式来模拟针对 IoT 设备的渗透攻击(被外

表 5　对被保护设备相关攻击行为的防御能力测试结果

Table 5　Results of protection from attack behaviors using WRGuardian framework

	被保护设备 运行平台	协议	渗透/对外 攻击方式	成功 处置
1	Atsmart 微插座 1.2.5.0	Telnet	Lightaidra 弱口令尝试渗透	✓
2	HIKVISION IP 摄像机 DS-2CD3Q10FD	HTTP	弱口令尝试渗透	✓
3	PHICOMM PSG1208 智能路由器	HTTP	命令注入漏洞	✓
4		IRC	Lightaidra 僵尸网络通讯	✓
5	ASUS RT-AC66U	TCP	Lightaidra 僵尸程序 DDoS	✓
6	智能路由器	HTTP	Mirai 僵尸程序 DDoS	✓
7		SMTP	大量发送垃圾邮件	✓

界攻击),测试防御框架的处置情况。目前恶意程序使用的主要是弱口令尝试渗透和利用漏洞两种,而由于 IoT 设备型号众多,系统版本指令集各异,漏洞方式的普适性不高,大规模利用较少使用,目前被曝光的 IoT 设备大规模恶意利用的案例也是以弱口令尝试渗透为主,为此,这些实验已经可以覆盖到主流的 IoT 设备渗透攻击方式。

实验 4 ~ 7 模拟设备被渗透感染沦为僵尸网络一员后的对外攻击操作,测试防御框架的处置情况。实验选取僵尸网络通讯、僵尸程序对外 DDoS 操作、僵尸程序对外大量发送垃圾邮件等场景,较为全面地覆盖到 IoT 设备被渗透后恶意利用的场景。

表 5 的实验结果显示,在以上不同的实验场景之下,WRGuardian 防御框架均能成功做出响应处置。下面将选取开源的 Lightaidra 僵尸网络对外通讯的场景介绍一下防御框架的响应处置情况。

实验中,被保护设备是 ASUS 的 RT-AC66U(固件版本 3.0.0.4.378 _9313),在未启动防御框架时,当使用 Telnet 连接到被保护设备上下载运行编译的 Lightaidra 僵尸程序后,我们搭建的 C&C 服务器上收到了该僵尸程序的上线信息,可以正常下发指令并执行,而当启动防御框架后,防御框架检测到 IRC 协议并不属于该设备正常通讯所使用的协议,连接被阻断,C&C 服务器上该僵尸程序下线,无法正常接收指令,成功处置。实验过程截图如图 5 所示。

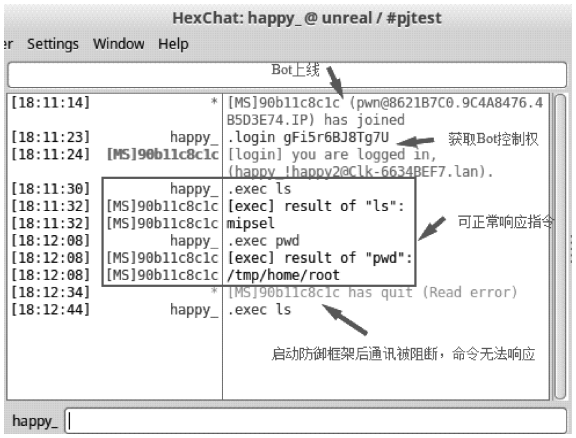

图 5　Lightaidra 僵尸程序对外通讯的场景实验截图

Fig. 5　Screenshot of Lightaidra bot communication experiment

4.2.2　主动防御能力测试

主动防御模块的漏洞排查修复功能主要是采取模拟攻击的方式探测设备是否存在漏洞并尝试修复加固操作。为了更全面地反映主动防御模块的漏洞排查修复能力,我们选取多种不同类型的漏洞设备,编写相应的漏洞检测处置插件并放入漏洞验证及修复方案插件库中。由于主动防御模块是由防御框架定期启动运行的,为加快实验速度,减少等待时间,实验中我们关闭了防御框架的定期启动功能,转而使用人工启动主动防御模块的方式。具体使用的漏洞情况和处置结果如表 6 所示。

表 6　主动防御模块的漏洞排查修复能力实验结果

Table 6　Results of detecting and fixing the security issues using active defense module

	被保护设备	漏洞编号	漏洞信息	成功处置
1	Atsmart 微插座	—	默认弱口令	✓
2	TP-Link TL-WDR4300	CVE-2015-3035	目录遍历漏洞	✓
3	TP-Link TL-WDR4300	CVE-2014-4727	存储型 XSS 漏洞	✓
4	ASUS RT-AC66U	CVE-2013-6343	缓冲区溢出	✓
5	TP-Link TL-WR840 N	CVE-2014-9510	CSRF 漏洞	✓

4.3　性能测试

由于 WRGuardian 防御框架需要对网络流量进行检测处理,在正常的网络传输中增加了一个环节,因此需要测试防御框架的部署对原有网络吞吐能力的影响情况。

为测试防御框架的影响情况,我们在 PC 上

模拟了 4 种用户常用的网络操作。

1）Ping 操作测试响应时间

Ping www. qq. com 测试网络响应时间；

2）HTTP 下载文件

使用 Wget 1. 15 程序下载文件

http://dldir1. qq. com/qqfile/qq/QQ8. 7/19113/QQ8. 7. exe

大小:59 006 272 bytes(56 M)；

3）FTP 下载文件

使用系统自带 ftp 程序下载文件

ftp://ftp. mirrorservice. org/sites/sourceware. org/pub/gcc/releases/gcc-2. 95/gcc-2. 95. tar. gz

大小:12 864 284 bytes(12. 3 M)；

4）打开网页

使用 Firefox 38. 0 的开发者工具分别强制刷新无缓存打开简单网页 www. baidu. com(21 个请求,539. 68 KB)和复杂网页 www. qq. com(254 个请求,4 007. 16 KB),观察记录页面加载时间。

具体测试结果如表 7 所示。

表 7 性能测试结果

Table 7 Results of performance test

测试项目	平均完成时间	
	未启用	启用
Ping www. qq. com	6. 133 ms	6. 542 ms
打开简单网页 www. baidu. com	0. 85 s	0. 96 s
打开复杂网页 www. qq. com	16. 01 s	17. 99 s
FTP 下载文件(12. 3 M)	14. 28 s (879. 5 KB/s)	16. 18 s (776. 3 KB/s)
HTTP 下载文件(56 M)	69 s (839 KB/s)	85 s (682 KB/s)

根据测试结果绘制各实验项目所用时间折线图(图 6),从图可以看出 WRGuardian 防御框架的部署对原有的网络吞吐能力有轻微的影响,在小流量环境下所用时间与未部署时相差不大,基本可以忽略不计。而在大文件下载环节才显示出一定的影响,由于防御框架目前可设置成只对 IoT 设备的流量进行转发处理,且 IoT 设备的大流量下载操作场景并不多,此类情形出现的几率不高,也在可以接受的范围内,后续可以进一步优化。

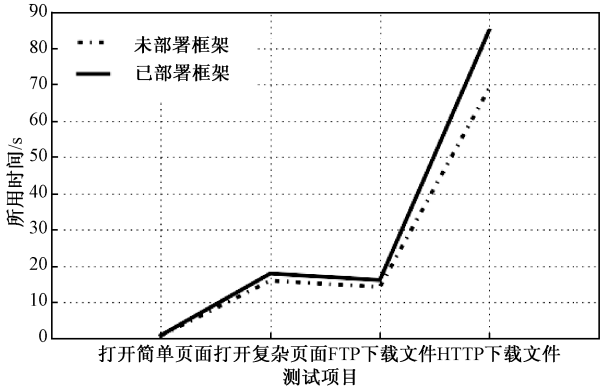

图 6 各个实验项目所用时间比较

Fig. 6 Comparison of elapsed time among different experiments

5 讨论

目前学术界和工业界并没有明确提出针对 IoT 设备的专用保护方案,每当某一型号 IoT 设备出现安全问题,最常用的解决方案往往是等待厂商发布修复版本的固件让用户自行更新,从源头入手让攻击失效。此方案面临以下 3 个问题。

1）厂商提供固件更新不及时甚至不提供；

2）用户自行更新固件难度大；

3）用户无法被及时提醒去更新固件。

第二种解决方案是在网络环境中部署相关的保护设备(如 IPS、IDS 等),从攻击攻击行为入手及时发现阻断。由于目前主流的 IoT 设备通常无法在内部安装第三方保护软件,只能采取从外部环境保护的方案。此方案对 DDoS 攻击等通用化的攻击行为有较好的保护能力,但属于治标不治本的方案,设备的安全漏洞仍然存在,安全问题的源头没有解决。此外,此类外部保护设备部署成本相对较高,通常被大型企业的所采用,一般家庭用户和 SOHO(Small office/home office, 小型企业或家庭企业)用户由于成本和设备数量较少等原因不会选择部署此类设备。而从目前曝光的几个 IoT 设备被大规模恶意利用的案例中可以发现这些案例中的被控设备往往是家庭和 SOHO 用户的设备,大型企业的 IoT 设备占比较少,甚至有部分恶意程序会主动避开特定企业的设备(如 Mirai 僵尸网络将会避免扫描攻击特定企业或者政府部门的设备[25])。

本文提出的 WRGuardian 防御框架利用无线路由器的特点从环境入手保护 IoT 设备,通过监测处理与 IoT 设备相关的网络流量应对目前针对 IoT 设备主流攻击行为,达到治标的效果,同时通过主动防御模块的模拟攻击操作,及时检测、修复已知漏洞,处理安全问题的源头,达到治本的效果。此外,由于防御框架只需部署在无线路由器之上,并不需要外部硬件或者修改设备原有系统,部署难度和成本较低,有利于家庭和 SOHO 用户

使用。

为方便框架的后续推广部署,本框架设计开发之时采取了多种措施(减少使用第三方库、静态编译等)增强框架程序的对不同平台的适应与移植能力,生产厂商的部署成本低。实验验证结果显示本框架可在多个品牌无线路由器(原生系统或者第三方开源系统)中正常运行,可以在一定程度上保护 IoT 设备且对原有的网络吞吐能力影响较小,让无线路由器生产厂商可以在无需更改硬件设计的前提下增强产品的功能。厂商可以直接在固件中集成本框架,或者采用在其路由器应用商店中以应用插件的形式提供本框架,无需改动固件,更加灵活且更易被厂商与用户接受。

当然,WRGuardian 防御框架也存在着改进和优化的空间,如流量监测的工作性能不高,受限于无线路由器的计算性能,有着很大的优化改进空间。另外随着无线路由器性能的增强,可考虑将机器学习算法引入到 IoT 设备的流量特点学习上,在发现设备异常流量后及时处理,以取代现有的根据流量协议情况判断是否是异常流量的方法,增强异常流量识别的准确度。此外,部分 IoT 设备并不直接接入无线路由器,而是通过 ZigBee、蓝牙等网络技术接入到配套的中转网关,再由中转网关接入到无线路由器从而接入互联网。由于此类设备流量并未直接经过无线路由器,影响了 WRGuardian 防御框架的保护能力。因此,能否将防御框架的工作场景扩展到这些中转网关中来增强对此类设备的防护能力,这也是本文下一步的一个研究方向。

6　结束语

随着智能家居这一理念的深入人心,IoT 设备的使用场景将不断扩大,使用规模和联网设备数量将保持高速增长,相关的安全问题将更加突出。由于各个厂商为了让自家设备尽快占领市场,往往只重视 IoT 设备的功能和研发速度,忽视设备的安全问题,这就导致 IoT 设备行业安全问题频发,预置弱口令、安全漏洞等问题频频被曝光;不仅如此,厂商对安全问题的不重视也导致各种修复固件迟迟不推出,事后补救也无法及时做到;即便推出修复固件,而各类 IoT 设备主要面向家庭用户,他们的安全意识和专业知识相对欠缺,IoT 设备的固件更新操作也不容易完成。

针对以上问题,本文提出的 WRGuardian 轻量级防御框架从被动消极防御和主动积极防御 2 个方面入手,对安全问题引发的攻击行为及时监测阻断,同时定期模拟攻击检测安全问题并修复,且无需外部硬件或者修改设备原有系统,减少了部署难度,降低了部署成本。经实验测试,能够对目前主流的 IoT 设备攻击方式进行防御拦截,希望能为 IoT 设备的安全防御提供思路和理论支持。

参考文献

[1] Gartner. Gartner says the Internet of things installed base will grow to 26 billion units by 2020 [EB/OL]. (2013-12-12) [2016-11-18]. http://www.gartner.com/newsroom/id/2636073.

[2] Anonymous. Internet census 2012 [EB/OL]. (2012-12) [2016-11-18]. http://internetcensus2012.bitbucket.org/paper.html.

[3] Paganini P. Lizard stresser hacking tool relies on compromised home routers [EB/OL]. (2015-01-10) [2016-11-18]. http://securityaffairs.co/wordpress/32022/cyber-crime/lizard-stresser-hacking-tool.htmlg.

[4] Krebs B. DDoS on Dyn impacts twitter, spotify, reddit [EB/OL]. (2016-10-21) [2016-11-18]. https://krebsonsecurity.com/2016/10/ddos-on-dyn-impacts-twitter-spotify-reddit/.

[5] Flashpoint. Mirai botnet linked to Dyn DNS DDoS attacks [EB/OL]. (2016-10-21) [2016-11-18]. https://www.flashpoint-intel.com/mirai-botnet-linked-dyn-dns-ddos-attacks/.

[6] Gamblin J. Leaked Mirai source code for research/IoC development purposes [EB/OL]. (2016-10-31) [2016-11-18]. https://github.com/jgamblin/Mirai-Source-Code.

[7] Fazzi F. IRC-based mass router scanner/exploiter [EB/OL]. (2015-6-19) [2016-11-18]. https://github.com/eurialo/lightaidra.

[8] Oikarinen J, Reed D. Internet relay chat protocol [EB/OL]. (1993-05) [2016-11-18]. https://tools.ietf.org/rfc/rfc1459.txt.

[9] Proofpoint. Proofpoint uncovers Internet of things (IoT) cyberattack [EB/OL]. (2014-01-16) [2016-11-18]. http://investors.proofpoint.com/releasedetail.cfm?ReleaseID=819799.

[10] Krebs B. Who makes the IoT things under attack [EB/OL]. (2016-10-03) [2016-11-18]. https://krebsonsecurity.com/2016/10/who-makes-the-iot-things-under-attack/.

[11] DHS. Strategic principles for securing the Internet of things [EB/OL]. (2016-11-16) [2016-11-18]. https://www.dhs.gov/sites/default/files/publications/Strategic_Principles_for_Securing_the_Internet_of_Things-2016-1115-FINAL_v2-dg11.pdf.

[12] 左青云,陈鸣,王秀磊,等. 一种基于 SDN 的在线流量异常检测方法[J]. 西安电子科技大学学报, 2015, 42(1): 155-160.

[13] 陈友,程学旗,李洋,等. 基于特征选择的轻量级入侵检测系统[J]. 软件学报, 2007, 18(7): 1 639-1 651.

[14] 朱应武,杨家海,张金祥. 基于流量信息结构的异常检测[J]. 软件学报, 2010, 21(10): 2 573-2 583.

[15] Acunetix. Web application security with Acunetix Vulnerability Scanner [EB/OL]. (2016-11) [2016-11-18]. http://www.acunetix.com/vulnerability-scanner/.

[16] IBM. IBM security AppScan [EB/OL]. (2016-11) [2016-11-18]. http://www-03.ibm.com/software/products/en/appscan.

[17] Tenable. Nessus vulnerability scanner [EB/OL]. (2016-01-01) [2016-11-18]. http://www.tenable.com/products/nessus-vulnerability-scanner.

[18] Levandoski J, Sommer E, Strait M. Application layer packet classifier for Linux [EB/OL]. (2009-01-07) [2016-11-18]. http://l7-filter.sourceforge.net/.

[19] Tecgraf. The programming language Lua [EB/OL]. (2016-10-14) [2016-11-18]. http://www.lua.org/.

[20] Nehab D. Network support for the Lua language [EB/OL]. (2016-07-23) [2016-11-18]. https://github.com/diegonehab/luasocket.

[21] Andersen E. A C library for embedded Linux [EB/OL]. (2012-05-15) [2016-11-18]. https://uclibc.org/.

[22] IETF Working Group. Request for comments (RFC) [EB/OL]. (2016-10-03) [2016-11-18]. http://www.ietf.org/rfc.html.

[23] Gailly J, Adler M. The gzip home page [EB/OL]. (2003-07-27) [2016-11-18]. http://www.gzip.org/.

[24] Fielding R, UC Irvine, Gettys J, et al. Hypertext transfer protocol: HTTP/1.1 [EB/OL]. (1999-06) [2016-11-18]. http://www.ietf.org/rfc/rfc2616.txt.

[25] Herzberg B, Bekerman D, Zeifman I. Breaking down Mirai: an IoT DDoS Botnet analysis [EB/OL]. (2016-10-10) [2016-11-18]. https://www.incapsula.com/blog/malware-analysis-mirai-ddos-botnet.html.