IoT 智能设备安全威胁及防护技术综述

王雅哲[1,2,3], 张城毅[1,2,3], 霍冬冬[1,2], 李佳琳[1,2,3]

[1] 中国科学院信息工程研究所 信息安全国家重点实验室, 北京 中国 100093
[2] 互联网智能设备信息安全北京市工程实验室, 北京 中国 100093
[3] 中国科学院大学 网络空间安全学院, 北京 中国 100049

摘要 伴随着物联网的产生和发展, IoT 智能设备越来越多地出现, 其大规模普及的同时, 也给用户个人资产安全与隐私保护带来了极大地冲击和挑战。本文围绕智能设备, 基于智能设备终端、云服务端和用户控制终端三端系统架构, 综述目前智能设备安全威胁的主要来源和技术攻击手段, 并针对性地梳理已有防护技术和安全研究现状。然后, 针对现有 IoT 智能设备安全防护体系缺失和安全设计不足的问题, 本文讨论提出了全生命周期的 IoT 智能设备系统防护模型设计思路。

关键词 智能设备安全; 安全威胁和防护综述; 系统防护模型设计
中图法分类号 TP309.2 DOI 号 10.19363/j.cnki.cn10-1380/tn.2018.01.004

A Survey of Security Threats and Defending Technologies on IoT Smart Devices

WANG Yazhe[1,2,3], ZHANG Chengyi[1,2,3], HUO Dongdong[1,2], LI Jialin[1,2,3]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
[2] Engineering Laboratory of the Internet Smart Device Information Security, Beijing 100093, China
[3]School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China

Abstract With the emergence and development of Internet of things, IoT smart devices increasingly appear and its large-scale popularity also has brought great impact and challenges on the user's personal assets security and privacy protection. In this paper, based on the smart device terminal, cloud server and user control terminal this three-terminal system architecture, we first summarize the main sources and technical attack methods of smart device security threats and combs the corresponding protection technologies and security research. Then, aiming at the problem of the lack of security protection system and the lack of security design on the existing IoT smart devices, this paper discusses and puts forward full life-cycle IoT smart device protection system model designing.

Key words Smart device security; summary of threat and protection; protection system model designing

1 引言

万物互联(Internet of Things, IoT)是继计算机、互联网与移动通信网之后的又一次信息产业浪潮, 其目标是通过各种信息传感设备与智能通讯系统把全球范围内的物理实体、信息系统和人有机连接起来, 提供更透彻的感知、更全面的互联互通和更深入的智能化服务。近些年来, 伴随着物联网的产生和发展, IoT 智能设备越来越多地出现在市场上, 作为信息空间和物理空间深度融合的代表产物, 已经从面向个人消费的先锋产品快速拓展到经济社会各个领域, 赋予教育、医疗、零售、能源、建筑、汽车等诸多行业新的服务手段, 支撑政府办公、公共安全、交通物流等城市基本职能的提升[1]。预计至 2018 年我国智能设备产品和服务的总体市场规模约 5000 亿元, 至 2020 年可达到万亿元水平[2]。

然而智能设备大规模普及的同时, 也给用户个人资产安全与隐私保护带来了极大地冲击和挑战。现有IoT智能设备侧重于功能实现, 而传统设备厂商安全能力不足, 或者考虑时间和成本等因素, 在系

通讯作者: 张城毅, 硕士研究生, Email: zhangchengyi@iie.ac.cn。
本课题得到中国科学院青年创新促进会(1105CX0105), 信息安全国家标准项目(智能互联设备信息安全技术要求), 国家重点研发计划(2017YFB0801900)资助。
收稿日期: 2017-04-10; 修改日期: 2017-05-18; 定稿日期: 2017-12-05

统设计上普遍忽略安全问题。黑客可以轻易利用设备安全漏洞, 使其成为传统网络攻击的新工具, 如利用 Aidra[3]、Darlloz[4]、Gafgyt[5]、Mirai[6]、Pnscan[7]等恶意代码感染智能设备并发动分布式拒绝服务攻击(Distributed Denial of Service, DDoS)造成网络瘫痪、设备拒绝服务和相关服务下线等严重后果。另外, 智能设备产生、处理以及传递海量来自物理空间和信息空间的安全敏感控制数据和隐私数据, 也极易成为黑客们的攻击对象。目前已经暴露如智能电表远程关闭[8]、智能门锁非法远程开启[9]、智能汽车未授权远程操作[10]等多项安全事件。更有不少黑客在近些年的黑客大会上演示他们攻破家电类智能设备的过程[11], 这些对智能设备的攻击除影响个人隐私安全、财产安全外, 已经证实针对智能设备的非授权指令攻击甚至可以危及用户的人身安全。目前, IoT 智能设备终端与云端深度融合是目前系统部署的主要模式, 在提高其智能化水平与控制便捷度的同时, 也带来了新的攻击面: 黑客可以利用云服务穿透连接物理世界的信息化访问接口, 导致设备恶意控制、大规模数据泄露、设备群体性异常等后果。因此, 如何保障智能设备安全成为当前信息安全领域亟待解决的重要问题之一。

具体来说, IoT 智能设备生态总体分为智能设备终端、云服务端和用户控制终端。本文对目前智能设备面临的安全威胁和已有的防护技术进行了系统地梳理, 发现在智能设备终端, 安全威胁主要来自 IoT 嵌入式操作系统恶意代码攻击、设备敏感数据泄露、设备服务权限验证漏洞攻击和设备接入网络协议安全等方面, 相应的防护技术研究主要关注硬件系统可信执行环境[12]、安全隔离方案[13]、硬件安全认证[14]、硬件完整性认证[15]和硬件端信息流异常检测技术; 在用户控制终端, 安全威胁和防护技术研究主要包括 APP 代码保护、终端对设备操作安全威胁和终端权限管理等方面; 而在云服务端, 一方面有来自传统网络层的安全威胁, 另一方面则有更多来自业务层的安全威胁, 包括业务层接入协议安全漏洞, 业务层设备权限管理漏洞和业务层设备批量攻击威胁。目前云端防护技术主要是信息流的异常检测。然而现有智能设备安全防护技术大多集中于安全事件处理的某个阶段或仅针对具体问题, 缺乏涵盖智能设备安全事件全过程的安全防护体系。

需要指出的是, 目前针对智能设备安全威胁及防护技术, 尚未有系统详细的综述研究。本文在前期工作的基础上, 结合自身的认识和理解, 旨在此领域进行整体论述和分析, 为学术界和相关从业者提供一定的参考。

本文主要有如下两部分的工作和贡献:

(1) 围绕智能设备终端、云服务端和用户控制终端三端架构, 综述目前智能设备安全威胁的主要来源和技术攻击手段, 并针对性地梳理已有防护技术和安全研究现状。

(2) 针对现有 IoT 智能设备安全防护体系缺失和安全设计不足的问题, 本文对 IoT 智能设备系统在安全防护全生命周期下的能力需求进行了讨论, 提出了系统防护模型设计思路。

本文的组织结构如下: 第 2 节描述智能设备相关背景; 第 3 节详细综述智能设备安全威胁以及目前相应的安全防护研究; 第 4 节思考和讨论全生命周期的 IoT 智能设备系统防护模型设计思路; 第 5 节总结全文。

2 研究背景

IoT 智能设备是在万物互联时代产生的一类智能化硬件设备。大多通过软硬件结合, 对传统硬件设备进行改造, 并加入联网模组, 使得设备拥有联网和交互的“智能化”功能。目前智能设备产品从应用场景上划分主要包括智能家居、智能交通、智能医疗类、智能网络设备等。从使用方式上划分还包括可穿戴设备和无人机设备等。在市场和产品最为活跃的智能家居领域, 传统家居产品主要通过增强感知交互、控制和互联网功能实现智能化, 在单品智能基础上, 互联网企业、家电企业通过智能家居平台整合统一的接口、标准和协议, 实现智能家居产品的互联互通。随着连接的产品数量逐渐增多, 开始出现专门用于智能控制的中枢产品, 如智能家居控制中心 SmartThings Hub[17]和 Google Home[18]能够连接其他家居设备, 通过语音识别用户指令, 实现对其他设备的控制。根据研究机构 Research and Markets 发布的最新报告[19]预计, 未来五年全球智能家居设备和服务市场将以每年 8%-10%的速度增长, 到 2018 年市场规模将达到 680 亿美元。而在发展迅速的可穿戴设备领域, 根据 IDC 公布的预测数据[16], 2015-2020 年可穿戴设备出货量在每一年中都实现两位数增长, 到 2020 年出货量将达 2.371 亿。

通过对国内外智能设备云平台, 如 Samsung SmartThings[20]、Apple HomeKit[21]、Google Weave/Brillo[22]、阿里智能[23]、腾讯物联[24]平台以及消费类智能设备厂商云平台的研究分析, 目前大部分智能设备采用了“智能设备终端”<—>“云服务端”<—>

“用户控制终端”的系统架构, 如下图 1 所示。智能设备通过移动蜂窝网络、Wi-Fi、蓝牙等方式直接或者间接地接入云服务端, 上传数据以及执行云服务端下发的指令。云服务端分为厂商私有云、PaaS(Platform as a Service, 平台即服务)云和公有云, 适用于不同的业务场景和设备厂商需求。用户控制终端(如手机、平板上的 APP)通过云服务端对设备进行激活绑定和远程控制。在这种架构中, 用户控制终端与智能设备之间通过云服务建立控制连接, 云服务端还提供用户管理、智能设备生命周期管理, 数据统计等功能。

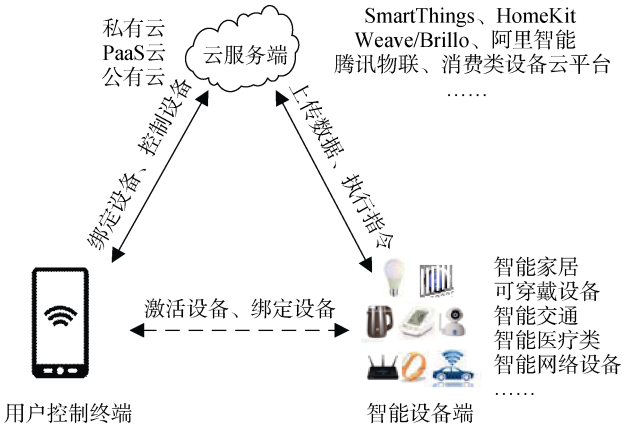

图 1　智能设备系统架构

Figure 1　Intelligent device system architecture

3　智能设备安全威胁及防护技术综述

智能设备在大规模普及的同时, 也给用户资产与隐私保护带来了极大冲击。本章将着眼于 IoT 智能设备安全研究领域, 从智能设备终端、用户控制终端和云服务端, 分别总结安全威胁的主要来源和技术攻击手段, 以及相关防护技术研究现状。此外, 本章还对国内主流互联网厂商和智能家电厂商提供云服务平台进行了详细的调研, 出于安全和隐私的考虑, 用字母表示厂商标识。本节部分内容基于四个有代表性的厂商平台(分别标记为厂商 A、厂商 B、厂商 C 和厂商 D)给出我们的安全分析。

3.1　智能设备终端安全威胁和防护

本小节将针对智能设备终端, 归纳总结安全威胁的主要来源, 包括 IoT 嵌入式操作系统恶意代码攻击威胁、敏感数据泄露安全威胁、服务权限验证安全威胁和设备接入网络协议安全威胁, 然后总结主要的防护技术研究, 包括终端系统加固防护技术、硬件脆弱性防护技术、硬件容错恢复技术和硬件端信息流异常检测技术。

3.1.1　智能设备终端安全威胁

(1) IoT 嵌入式操作系统恶意代码攻击威胁

在当前 IoT 智能设备终端操作系统中, 嵌入式 Linux 系统被广泛使用。常见的智能设备如家电智能设备、网络摄像头、网络路由器等都使用嵌入式 Linux 系统。这些系统具有安全防护措施少、装载设备数量多、具有真实 IP、24 小时在线等特点, 所以近些年来攻击者常常把此类 IoT 智能设备作为首选攻击目标。以嵌入式 Linux 系统为攻击目标的恶性代码在 2008 年第一次被报道[7], 早期的嵌入式 Linux 恶性代码使用 MIPS 对网络路由器进行感染攻击。2012 年发现的 Aidra 蠕虫病毒[3]可以对除了使用 MIPS 的网络路由器以外的其他机顶盒等多种嵌入式 Linux 系统环境进行感染[7]。2016 年 10 月发生了一起利用 IoT 智能设备发起的 DDoS 攻击, 攻击对象是为美国众多公司提供域名解析服务的 Dyn 公司, 攻击导致大半个美国互联网瘫痪[25]。受影响的站点包括 Twitter、Github、Spotify、Paypal、BBC、华尔街日报、纽约日报等大批知名网站[26]。根据安全人员的事后分析, 此次 DDoS 攻击事件来源于名为“Mirai”的恶意代码, 攻击目标大部分为网络路由器, 智能相机等 IoT 智能设备, 攻击涉及的 IP 数量达到千万量级[6]。整体来说, 自 2008 年发现 IoT 恶意代码以来, 对特定物联网智能设备进行感染的恶性代码不断进行变种升级, 主要嵌入式 Linux 恶意代码时间轴[7]如下图 2 所示。下文对其中五种代表性恶性代码

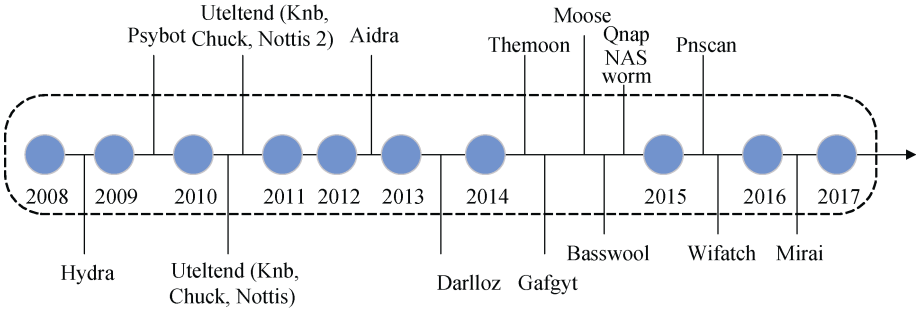

图 2　主要嵌入式 Linux 恶性代码时间轴

Figure 2　Time axis of main embedded Linux code

Aidra[3]、Darlloz[4]、Gafgyt[5]、Mirai[6]和 Pnscan[7]进行介绍。

■ **Aidra 恶意代码**

早在 2013 年, 互联网人口普查项目结束后, 研究人员发现了至少 40 万种嵌入式设备由于厂商提供的默认凭证而被利用, 匿名发布的报告中提到了一种针对 IoT 设备的恶意代码 Aidra[27]。不久, Aidra 的僵尸网络被重新规划和设计, 这些恶意代码还借用了 Torlus/Gayfgt 的 telnet 扫描工具, 并整合 60 余个凭证, 当时在 5 天之内就感染了超过三千台设备。这类恶性代码不仅在 MIPS 程序中传播, 也在 MIPSEL、PowerPC、SuperH 等多种程序中传播[28]。

■ **Darlloz 恶意代码**

Darlloz 恶性代码在 2013 年 10 月被发现, 其主要通过物联网病毒蠕虫感染攻击英特尔 x86、MIPS、Arm、PowerPC 等系统。不同于其他恶性代码一般以 DDoS 的形式进行攻击, 此恶性代码主要以挖掘类似比特币等虚拟货币为主要攻击目标[7]。

■ **Gafgyt 恶意代码**

Gafgyt 于 2014 年 8 月第一次被发现, 特别是 2014 年末在 Lizard Squard(Xbox Live)与 PlayStation Network 的 DDoS 攻击发生后变得更加有名。2015 年 1 月 Gafgyt 的源代码被公开[7], 目前 Gafgyt 也是存在最多变型的恶性代码。2015 年 3 月, 国家计算机网络应急技术处理协调中心对 Gafgyt 恶意代码在我国境内的感染情况进行了监测和分析。经分析, 我国境内互联网上感染该恶意代码的活跃被控 IP 地址中部分可确认为视频监控类智能设备的联网 IP, 涉及包括海康威视在内的至少 5 家国内知名视频监控类设备生产企业[29]。

■ **Mirai 恶意代码**

Mirai 于 2016 年 5 月第一次被发现, 在 2016 年 10 月初公开了其源代码以后, 使用此恶性代码的变型越来越多。2016 年 9 月黑客对某安全博客实施了大规模 DDoS 攻击, 2016 年 10 月对美国域名提供商 Dyn 进行了 DDoS 攻击, 随后 Mirai 变得更加有名。Mirai 恶性代码具有 UDP Flood, Syn Flood, ACK Flood, GRE IP Flood 等多种 DDoS 攻击功能[30]。

Mirai 主要的感染对象是物联网设备, 包括网络路由器、网络摄像头、数字视频录像设备。这些设备主要是 MIPS、ARM 等架构, 具有大规模批量生产、批量部署的特点。在很多应用场景中, 由于集成商、运维人员能力不足, 设备存在默认密码、弱密码、严重漏洞未及时修复等不安全因素, 导致被攻击者植入木马。Mirai 主要攻击过程如下图 3 所示, 针对物联网设备 DDoS 入侵主要通过 Telnet 端口进行密码暴力破解, 或默认密码登录, 如果登录成功, 通过 Telnet 远程登录成功后就尝试利用 busybox 等嵌入式必备的工具进行 wget 下载 DDoS 功能的 bot, 修改可执行属性, 运行控制物联网设备。由于 CPU 指令架构的不同, 在判断了系统架构后僵尸网络可以选择 MIPS、ARM, x86 等架构的样本进行下载, 运行后接收相关攻击指令进行攻击[6]。

■ **Pnscan 恶意代码**

Pnscan 恶性代码于 2015 年 8 月被俄罗斯安全公司 Dr. Web 发现。此恶性代码如果将 ARM、MIPS、PowerPC 系统感染的话就会对 CVE-2013-2678 的漏洞进行攻击[31]。当这些恶意程序渗透进了被入侵的路由器之后, 它们会将自己添加进设备的自动运行列表并在注册表中进行注册。完成之后, 它们就可以通过 IRC 来控制列表中的服务器地址并与服务器建立连接。这些后门程序的功能非常的多, 它们可以发动各种 DDoS 攻击, 还可以执行入侵者所发送的相关命令。根据 Dr.Web 公司发现时的数据, 已经有 1439 种设备感染了上述恶意程序, 从受感染设备的地理位置分布情况看, 绝大多数的设备感染发生在日本, 还有大量的攻击发生在德国, 美国以及中国台湾等地区[32]。

(2) 智能设备敏感数据泄露安全威胁

信息安全中有一项原则叫“代码数据分离原则”, 即为了数据安全, 要将系统运行代码和系统中敏感数据的存储位置隔离。一些缺乏安全意识的智能设备厂商把很多敏感数据(如用户密码, 加密 key)固化在系统程序所在固件中, 一旦固件被逆向分析, 敏感数据将全部被暴力窃取, 造成数据泄露和加密方式泄露[33]。2011 年的 CVE-2011-4859 披露了施耐德公司 PLC 以太网模块固件后门, 漏洞显示该设备远程 ftp 登录的用户名和密码竟然直接固化在设备软件代码中, 攻击者可以无任何混淆地进行读取并对设备进行控制[34]。

此外, 在固件内存溢出和命令执行方面, 由于 flash 的大小限制或是固件代码多样性, 开发者对安全的忽视特别明显, 如 D-LinkDSP-W215 的溢出漏洞[35], 没有检测输入字符长度过长从而造成溢出, 以至攻击者利用溢出攻击获取设备端敏感数据, 甚至获取设备控制权, 造成更大的危害。

(3) 智能设备服务权限验证安全威胁

在智能设备服务权限验证方面, 存在的问题主要有三个方面, 一是嵌入式系统开放调试接口的安全隐患, 二是设备固件升级的安全威胁, 三是控制

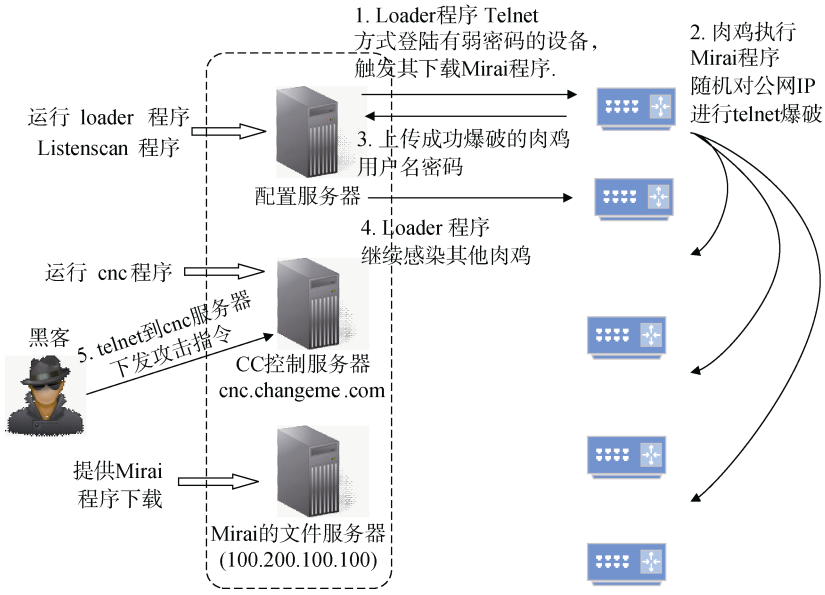

图 3　Mirai 攻击过程示意图

Figure 3　Schematic diagram of the Mirai attack process

指令重放攻击对设备造成的危害。三者均是由设备未对被控制的服务权限进行验证导致。

很多嵌入式系统开放调试接口，在开发者移植现有物联网系统并进行二次开发时，由于对物联网系统的不了解，未关闭物联网系统很多自带的默认功能从而导致安全隐患。如 ftp、telnet 等服务弱身份验证，开启 telnet、tftp 等服务默认口令等。

此外，在固件升级方面，很多固件升级没有验证环节，若黑客刷入恶意固件修改设备软件代码，就可以控制设备并对设备进行非法操作。早在 2013 年，国外安全公司 IOActive 就曾对某款验钞机进行研究，发现其在固件更新时未作签名校验，导致可刷入被恶意篡改的固件，研究人员通过刷入经篡改的固件，使得一张手写假钞被验钞机识别为真钞[36]。

在控制指令重放攻击方面，由于智能设备对服务端的验证普遍较弱，多数情况下无条件地执行服务端下发的指令。如果黑客窃取控制指令进行重放，就会造成设备不正常工作，严重情况下影响用户财产和生命安全，如一些涉及人体健康的生物医学智能设备，特别是心脏起搏器、胰岛素泵等医疗设备，倘若出现安全漏洞，可能就是直接危害生命的问题。

(4) 智能设备接入网络协议安全威胁

智能设备通过接入协议接入到互联网中从而连接到云服务端。目前主要采用的接入协议有 Wi-Fi、蓝牙和 ZigBee 等，接入协议层面的设计缺陷和安全漏洞也是智能设备安全威胁重要来源。

Wi-Fi 接入传输的信息大多为明文数据包，如配网信息、智能设备和用户控制终端内网交互数据包以及智能设备与服务器直接交互 TCP 包。以配网信息为例，如 Easylink[37]协议采用组播的方式传播路由器配网信息，包括 SSID 和配网密码，容易被窃听攻击，使得攻击者连入局域网甚至进入家庭内部网络，并以此作为攻击跳板，攻击家庭局域网内其他设备。我们分析厂商 A 空气质量监测设备，发现在其设备配网环节，使用 EasyLink 协议组播存在热点信息泄露的安全威胁，具体如下图 4 所示，攻击者可以获取到当前局域网路由器的 SSID 和接入密码。

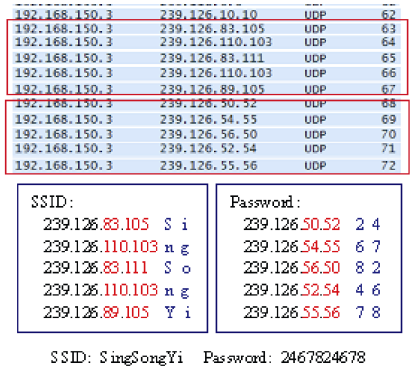

图 4　Wi-Fi 局域网 Easylink 热点信息泄露

Figure 4　Wi-Fi LAN Easylink hotspot information leak

蓝牙这类接入协议，主要应用于可穿戴设备，车联网等近用户距离设备，与用户隐私关系很密切，安全问题的出现也会造成比较严重的后果。2014 年，有安全研究团队就发现某款车联网产品存在漏洞(如图 5 所示)，攻击者可以远程破解蓝牙密码，并实时获取汽车数据[36]，比如车速、温度等等，如果设备硬

件进一步支持写 OBD 接口(车载诊断系统接口), 攻击者还可能远程控制汽车。

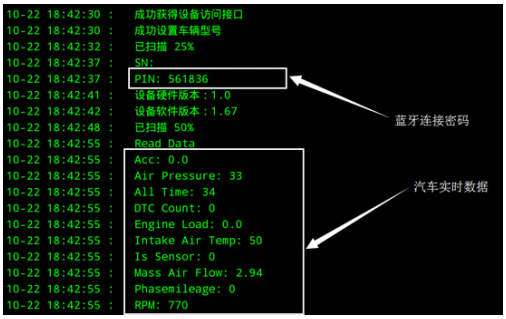

图 5 某安全团队远程破解车联网设备

Figure 5 Remotely cracking car networking equipment

ZigBee 是一种短距离、低功耗的无线通信技术, 最大传输速率为 250 Kbps, 普遍传输范围在 10~100 米[38]。目前已有一些智能家居系统使用到 ZigBee 协议, 被应用于门窗、家电、安防等用途。2015 年初, 小米发布基于 ZigBee 协议的智能家庭套装, 使得基于 ZigBee 协议通讯的智能家居产品又映入人们的眼帘。某安全研究团队对 ZigBee 协议安全性进行研究[36], 发现其主要风险主要在密钥的保密性上, 比如明文传输密钥, 或者将密钥明文写在固件中, 这些都可能直接导致传输的敏感信息被窃取, 甚至伪造设备去控制智能家居产品。

3.1.2 智能设备终端安全防护技术

(1) 智能设备终端系统加固防护技术

目前智能硬件防护模型的构建方案灵感来源于 TrustZone[12]的技术思想, 通过对智能硬件内部电路进行修改, 建立可信执行环境(Trusted Execution Environment, TEE)实现一种对敏感数据以及方法进行单点隔离的系统结构。如 Job Noorman 等人[39]基于对 MCU 硬件以及编译器进行的修改提出 Sancus 方案, 使系统支持为每一个软件划分安全隔离区域, 仅隔离区域内的数据可以互相访问。方案如图 6 左侧所示, 内存中受到保护的模块包含代码常量以及保护数据两部分, 这两部分的起始以及终止地址将存储于受保护存储区中。MCU 通过检查程序计数器 PC 和模块的地址确认访问保护数据的请求是否来自于本模块的代码, 若确定为本模块代码则可以正常访问, 否则将抛出异常。在这一工作的基础上, 文献[40]利用 Sancus 提供的硬件能力, 把可信认证模块(Trust Assessment Module, TAM)安全植入 Contiki 操作系统中, 检测系统软件栈其他模块的完整性。同时, 根据设施提供者以及软件提供者信息设计部署协议建立会话密钥 $K_{N,SP,SM}$, 与软件提供者进行可信通信, 如图 6 右侧所示, 部署协议分为以下几个步骤: 1)设施提供者与设备共享密钥 K_N; 2)设施提供者根据软件提供者代号生成 $K_{N,SP}$ 并与软件提供者共享; 3)软件提供者向设备部署 TAM 模块; 4)设备内部利用 KN 与 TAM 中 SP 信息生成 $K_{N,SP}$; 5)根据 TAM 在系统内部的布局信息, 计算 SM 并生成 $K_{N,SP,SM}$; 6)设备把 TAM 在系统内部的布局信息发送给软件提供者并使其生成 $K_{N,SP,SM}$; 7)此时, TAM 可以与软件提供者进行可信会话, 获取设备内部软件栈完整性报告。

Patrick Koeberl[13]等人针对 MPU 仅支持为代码划定特权级进行安全防护的局限, 对其硬件电路进行了改造, 使 MPU 支持 CPU 输入指令地址和数据地址, 通过程序寄存器 PC 建立起两者的关系, 使 MPU 拥有细粒度的代码执行可认知(Execution-aware, EA)

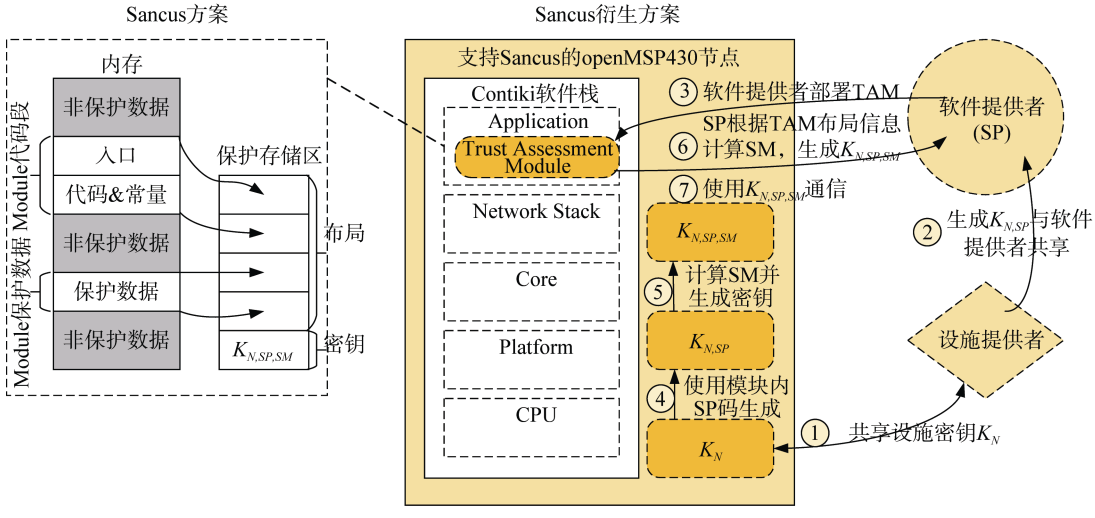

图 6 Sancus 以及其衍生方案

Figure 6 Sancus and its derivatives

能力, 形成基于 EA-MPU 的 TrustLite 方案。方案将判断当前代码的地址(*Curr_IP*, 主体)是否拥有访问待访问数据(*Data*, 客体)所在地址的权限(读/写/执行), 进而决定是否可以访问数据映射的计算资源。这种方式为系统中预设的所有软件模块提供了安全的隔离环境, 使每一个软件模块处于受保护的状态, 保护内存以及外设资源不受到非法访问, 若为非法访问, 则系统抛出内存错误异常, 如图 7 左侧所示。而文献[14]基于 TrustLite 技术提出了一种针对 IoT 设备的安全架构 TyTAN, 为远程认证、安全存储、进程通信代理模块、完整性检测模块等提供了隔离环境, 为系统建立信任根, 并与处于特权级的实时操作系统相分离, 操作系统在无授权情况下也无法对信任根进行访问, 如图 7 右侧所示。与此同时, 该架构弥补了 TrustLite 方案需要预先为模块配置安全属性的不足, 基于 FreeRTOS 环境实现了安全任务的动态加载, 可以根据需要随时建立安全任务。方案对 FreeRTOS 系统的安全能力进行了扩展并为系统内运行的重要任务提供强隔离环境以及实时安全防护。

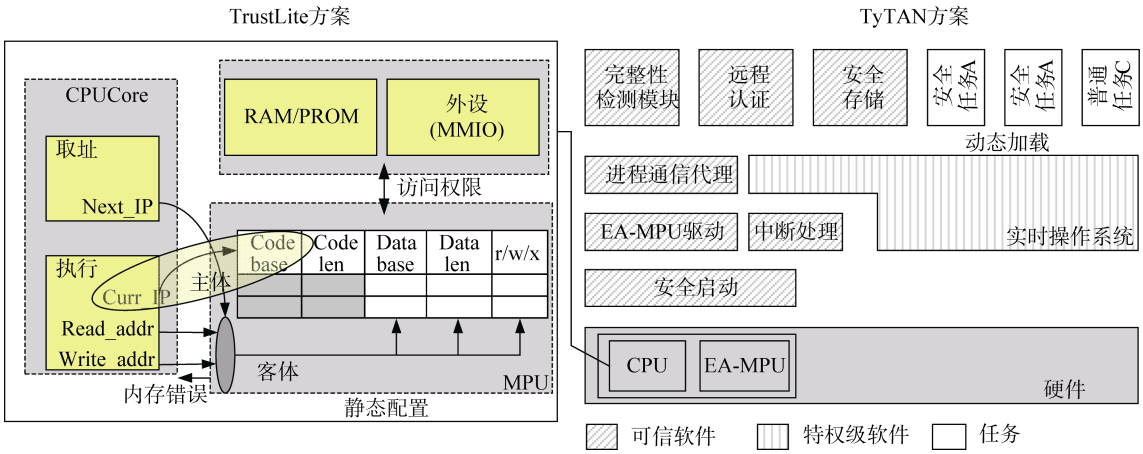

图 7 TrustLite 以及 TyTAN 方案

Figure 7 TrustLite and TyTAN scheme

除此之外, Karim ElDefrawy[15]等人通过改进 MCU, 在 Flash 和 SRAM 空间插入只读的安全代码 SMART, 外部程序无法对其进行修改。在该方案中系统可以利用远程认证的方式触发 SMART 验证系统内任务完整性, 建立动态信任根。不仅在学术界, 产业界也积极推进弱计算能力设备的安全方案, 如日前 ARM 公司推出了一种专门针对 IoT 智能硬件设计的可信环境扩展方案 TrustZone-M[41], 该方案在保证软硬件低消耗的同时还可以提供较好的安全性, 利用安全属性单元(Secure Attribute Unit, SAU)在不额外增加特权级的情况下划分安全域与非安全域, 为敏感任务提供安全的执行环境。产业界和学术界都在积极推动基于硬件的弱设备安全隔离环境的发展, 可见该技术模式将是未来的发展趋势。

(2) 智能设备硬件脆弱性防护技术

随着 PC 和智能手机时代安全技术的发展, 远程完整性认证技术已成为脆弱性防护的主流方案。该技术通过设备搜集软件完整性报告, 利用认证协议发送给可信验证方, 证明设备处于可信状态, 保证智能硬件的正确运行。远程完整性认证技术主要包含认证协议和完整性报告搜集两部分。在认证协议研究方面, 文献[42-44]提出挑战响应式的认证协议, 通过设备与验证者多次数据交互完成密钥的协商与完整性报告的上传, 实现对设备的校验。然而目前智能设备逐渐向规模化趋势发展, 现有集中式认证协议一对一的认证模式难以同时支撑大规模设备。学术界已有利用分布式思想设计群体认证协议的研究, 如 H.Park 等[43]利用多个设备报告相互比较的方式, 确定异常设备; N.Asokan 等[40]为设备群体设计逐跳(hop by hop)方式的验证方案, 传递验证信息到汇聚节点进行整体验证, 但是该方案仅可以验证整个设备群体完整性, 不能准确定位异常设备。从理论研究阶段来看, 针对智能设备认证协议研究工作刚刚起步, 亟待丰富完善。在完整性报告搜集方面, 文献[45-47]提出基于可信软件栈实现报告搜集的方案, 这些方案需要较强安全假设, 要求在执行过程中攻击者不能对设备发起攻击[44], 限制较大。文献[48-50]基于安全硬件模块, 提出安全性更高的方案。但这些方案较为复杂, 成本较高, 无法直接在 IoT 智能硬件平台部署。因此, 设计基于低成本安全硬件模块的完整性报

告采集方案, 将是今后智能硬件认证技术发展的重要趋势。

软件同构化海量部署是 IoT 智能硬件面临的另一个重要脆弱性威胁。同类型智能硬件部署的系统往往拥有相同的内存布局, 同构化严重, 攻击者可以利用这一特点, 将单个硬件的攻击方案复制到全部同构化硬件, 在短时间内造成大范围安全事件。现有终端计算平台解决软件同构化问题的主要思路是通过对物理内存与虚拟内存的多样化映射, 减小大批量同质化设备同时遭到破坏的威胁。如 S.Forrest[51]于 1994 年首次提出建立多样化计算机运行环境的理念, 其通过修改 Gcc 编译器, 为局部变量分配了随机大小的冗余空间, 从而对栈空间进行了一定程度的随机化。Elena Gabriela[52]提出了指令随机化概念模型 RISE, 即程序被映射到虚拟地址空间, 对其二进制指令代码进行加密存储, 在执行时动态解密调用。Monica 等[53]针对恶意攻击中往往需要调用 *exec* 等系统调用来完成的特点, 对系统调用地址进行随机化处理, 阻止攻击者通过硬编码方式定位到相关的系统调用。除上述研究外, 科研界和产业界的研究重点主要集中在内存地址随机化技术(ASLR)[54-57], 该技术通过对库函数入口、栈、堆的位置进行随机化使得攻击者难以定位这些资源, 造成攻击失败。然而上述随机化技术对数据的多样化操作均需要借助于虚拟内存技术, IoT 智能硬件的硬件能力受限, 缺乏硬件辅助如内存管理单元 MMU 的支持, 无法进行内存的虚拟化, 而软件系统架构的多样性使得这些针对操作系统架构特性进行数据随机化的方案缺乏移植性, 现有随机化方案很难在智能硬件上直接使用。

(3) 智能设备硬件容错恢复技术

目前在PC机和智能手机上进行系统容错恢复的主要实现方案是通过对受损模块进行还原操作, 如 Trivedi [58]等利用重启的方法解决了软件系统中很多已知或未知的异常, UCBerkeley 和 Stanford 大学合作的面向恢复计算[59]以及递归重启技术[60], 其他的相关技术还有 N 版编程(N-Version Programming)[61]、恢复块(Recovery Block)[62]、检查点与还原技术(Check pointing and Recovery Technology)[63] 、复制(Replication)[64]技术等。针对 IoT 智能硬件系统恢复技术的研究才刚刚展开, 如文献[65-67]利用空中下载 OTA 技术, 为智能硬件重写固件, 可间接起到系统的功能修复作用。但这些工作不考虑 OTA 过程安全性, 不能根据硬件受损情况灵活配置策略。

强计算能力平台通常将可靠性和完整性放在首位, 设备可以利用更多的计算资源和安全元素对受损模块进行监控和对应处理, 如 N-版编程需要在系统中留有软件模块的多个拷贝, 在执行某一个指令时所有的拷贝任务都要执行, 不断比较结果, 对得出不同结果的模块拷贝进行覆写或重启, 此方式将严重消耗设备的软硬件资源。而 IoT 智能硬件把系统的可用性放在首位[68], 当系统出现安全威胁时, 首先应保证硬件的运行, 其次才会投入有限的计算资源对受损模块进行处置。IoT 智能硬件计算环境无法适配强计算能力平台下的恢复技术, 需要以可用性保障为目标设计相应的智能硬件受损恢复方案。

(4) 智能设备硬件端信息流异常检测技术

信息流在硬件端表征为数据流和控制流。远程控制指令以数据流的形式进入智能硬件, 被系统解析后, 将指导程序选择不同的分支进行执行, 进而形成控制流。针对系统态信息流的恶意攻击将导致指令越界, 敏感数据泄露, 缓冲区溢出等危害。

在针对系统态数据流的安全监测中, 国内外的研究主要通过对系统数据处理架构进行修改的方式进行数据追踪, 如 William Enck[69]提出了一种利用安卓架构实现对重要数据进行实时追踪的架构 TaintDroid, 而文献[70-72]将程序输入标记为污点源, 通过监控程序动态执行过程, 在污点数据被异常使用时产生警告。然而, 这些方案仅适用于强计算能力嵌入式系统(如安卓, Linux), IoT 智能设备的软硬件环境无法支撑其复杂实现逻辑, 且仅针对数据流进行监控的方案, 无法掌握系统内程序的运行状态, 无法检测对控制流的攻击。

在针对系统态控制流的安全监测中, 学术界也已经进行了一些工作, 如文献[73]介绍了一种针对 IoT 智能设备控制流进行完整性检验的方案 C-FLAT。如图 8 所示, 验证者首先对模块 A 中控制流路径进行静态分析, 获取所有执行路径的度量值, 之后向设备发送挑战值, 测试设备在执行相同控制流路径所得出的度量值是否相同。设备端在收到验证指令后, 经利用收集代理动态收集控制流信息, 在安全域度量引擎内部形成控制流报告, 通过远程验证控制流报告正确性以保证系统按照约定的方式执行。Abadi 等[74]在 CFG 中构造控制流转移的合法目标地址集合, 在实际运行时校验转移目标是否越界, 实现对控制流完整性的监控。

然而, 上述工作仅针对于控制流的完整性, 无法对数据攻击进行防御, 有研究者通过构造特异代码片段, 将输出函数(如: *send* 方法)的参数指向敏感数据(如: 存储密钥的结构体), 造成数据泄露[75]。目

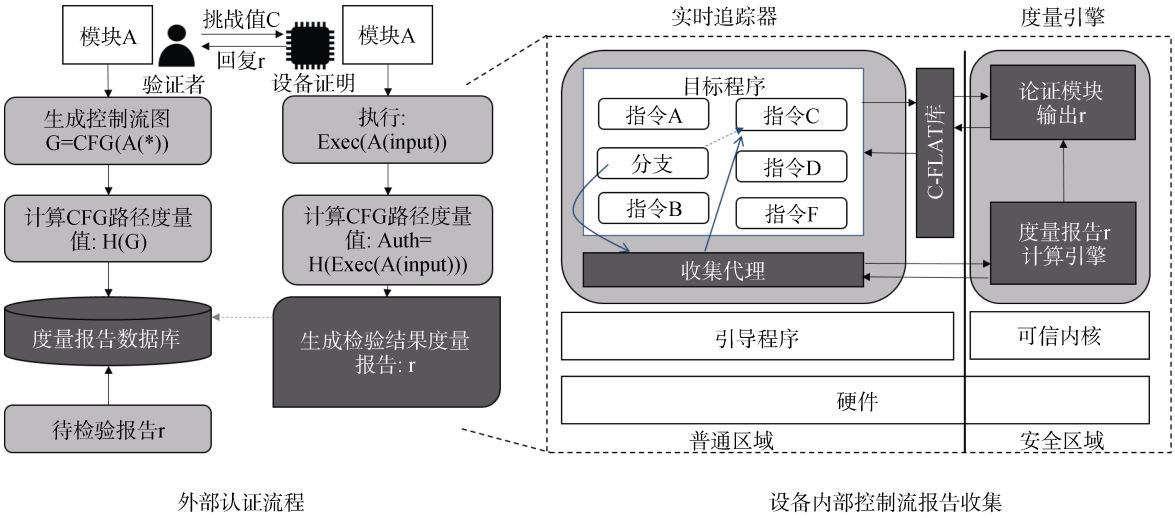

图 8　C-flat 方案

Figure 8　C-flat scheme

前的研究方案仍从数据流或控制流单一角度对远程控制指令进行追踪分析, 无法同时对数据流与控制流的变化进行综合分析, 当数据流和控制流同时受到攻击时, 系统安全将无法得到保证。

3.2　用户控制终端安全威胁和保护

本节从用户控制终端出发, 将从终端代码安全、终端对设备操作安全威胁和终端权限管理三个方面论述安全威胁和保护。此外, 本小节在全面调研和梳理的基础上, 对四个主流智能设备平台厂商系统进行分析, 其结果如右表 1 所示。

3.2.1　用户控制终端代码安全威胁和保护

目前, 智能设备用户控制终端以 APP 为主, 通过 APP 完成用户登录, 设备控制等操作。由于用户控制终端涉及设备控制和云端接入, 其代码安全性需要得到特别的保护。随着智能移动终端设备的快速普及, 移动互联网应用呈井喷式爆发, 每日新开发的各类互联网应用, 都在以几何级数量递增。在开发者辛辛苦苦开发时, 一些不劳而获者瞄准这个潜力巨大的市场, 各类非法利益团体对 APP 进行破解、篡改、二次打包, 严重损害了开发者和设备厂商的利益, 给设备用户带来巨大安全风险。

通过我们研究发现, 目前市面上相当一部分智能设备厂商, 其用户控制终端 APP 的安全隐患严重, 流行的反编译工具, 如 apktool[76]、dex2jar[77]等, 都能反编译得到用户控制终端 APP 大部分源代码。更有甚者, 攻击者通过破解代码分析, 二次打包将 APP

表 1　四个代表性厂商平台安全分析

Table 1　Security analysis of four representative vendors

智能设备厂商标识				A	B	C	D
用户控制终端	代码保护安全威胁			●	●	◎	○
	权限管理问题造成隐私泄露和过度授权			●	●	◎	◎
	对设备操作安全威胁	绑定	设备标识命名规则可猜测	●	●	●	○
			绑定操作无设备参与	●	●	●	●
		控制	可非法控制未绑定设备	●	○	○	○
		授权	重复绑定不提示原绑定者授权	●	●	○	○
云服务端	传统网络层安全威胁			◎	◎	◎	◎
	业务层协议安全威胁		信道可窃听	●	●	◎	◎
			会话令牌不失效	○	●	○	○
			消息不抗重放	●	●	●	◎
	业务层内容安全威胁		单条异常	●	●	●	◎
			批量异常	●	●	◎	○

(注: ● 表示存在且较强, ◎表示存在但较弱, ○表示不存在)

调试接口打开, 能获取几乎智能设备所有控制逻辑和数据信息。上述代码保护的缺乏将造成业务协议泄露、数据传输加密算法破解、数据重放、身份伪造等危害。在我们对智能设备厂商用户控制终端 APP 调研情况来看, 如表 1 所示, 厂商 A 和 B 存在此类安全威胁且问题较大, 厂商 C 也存在此类威胁但是有一定的安全手段对抗反编译。与此同时, 厂商 D 代码保护措施相对较好, 其关键代码写在了 SO 库中, 并给代码复杂的混淆保护, 给逆向破解者提高了破解难度, 使得攻击者利用普通的反编译技术无法得到大部分核心代码。

3.2.2 用户控制终端对设备操作安全威胁和防护

用户控制终端在使用过程中, 会对设备发起一系列的操作, 主要包含绑定、控制和授权等。这些环节的权限管理至关重要, 没有或缺乏相应的安全机制将使得设备被恶意控制, 从而造成极大的安全威胁。下面基于本文分析的四个代表性厂商产品, 论述用户控制终端对设备操作的安全威胁和防护。

(1) 设备绑定操作权限管理安全威胁和防护

正常情况下, 用户在绑定的过程中, 用户控制终端(如 APP)与设备要处于同一个局域网下, 用户控制终端与服务器、用户控制终端与智能设备、智能设备与服务器都会分别进行通信。但是通过对用户控制终端绑定智能设备这一环节的实验发现, 用户控制终端与智能设备的通信仅仅起到写入或获取设备唯一性标识的作用, 智能设备与云服务端通信是为了建立长连接, 真正起作用只有用户控制终端与云服务端的通信, 绑定过程即使没有智能设备参与也可以完成。换言之, 如果攻击者想要进行恶意绑定, 只要构造了正确的请求数据包, 伪造用户控制终端与云服务端通信的绑定请求过程, 即使智能设备并不在身边也可以实现绑定。如表 1 所示, 目前分析的四个智能设备平台在这方面都没有做处理。即使如厂商 C 和厂商 D 做了授权管理控制, 能够防止攻击者把已经绑定的设备“抢”过来, 但并不能防止攻击者绑定未被绑定过的设备。大部分设备的唯一性标识可遍历得到, 就可以实现批量绑定。这样当用户买来一台新设备, 也许会发现自己的设备已经被绑定, 然而自己并没有权限绑定和控制它。如下图 9 所示是我们对厂商 B 某款智能设备进行的攻击测试, 攻击者可以顺利的强绑定多个未经授权的设备(图右侧为抢绑定成功后的列表)。

(2) 设备控制操作权限管理威胁和防护

权限管理属于安全性保证中最基本的一部分。但是通过分析发现某家电厂商 A 在权限管理上没有

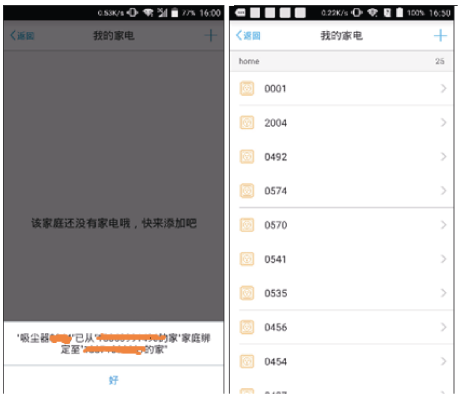

图 9 某厂商智能设备抢绑定攻击测试

Figure 9 Intelligent equipment robbing test

做任何处理, 即只要在设备唯一性标识处填写设备的 MAC 地址, 构造正确的请求数据包, 任何人都可以控制此 MAC 地址对应的设备。相较之下, 其他厂商在权限管理上就做了一定处理。即只有绑定了此设备的用户才有资格对此设备进行控制操作。此外, 我们还发现厂商 A 在云端存在开放接口, 可通过此接口查看其他用户的设备信息, 甚至扫描附近设备信息, 这就给攻击者留下了很大的想象空间。

(3) 设备授权操作权限管理安全威胁和防护

在分析厂商云平台时, 我们发现厂商 A 和厂商 B 由于在授权管理方面存在很大的漏洞, 所以导致恶意绑定较容易实现。攻击者通过构造正确的绑定请求数据包到服务器, 服务器就会将设备原有的绑定关系解除, 重新建立攻击者与设备的绑定关系。设备原绑定者可能会在不知情的情况下被“抢走”设备。而厂商 C 与 D 则在授权方面做了验证保护。即新绑定者必须得到原绑定者的授权, 或原绑定者先主动解除绑定关系, 或通过对设备进行重置操作甚至致电客服核对信息才能实现二次绑定。

3.2.3 用户控制终端内权限管理问题和防护技术

用户控制终端内权限控制不当, 一方面可能造成用户控制终端 APP 自身敏感数据泄露, 另一方面用户控制终端权限的滥用造成对用户隐私数据的窃取和设备的越权访问。严重情况下可能会影响到用户生命财产安全。

通过对相关智能家居设备用户控制终端 APP 的研究发现, 用户的登录信息、密钥信息、业务协议数据或多或少存放在本地文件夹中, 极容易在用户控制终端被恶意程序入侵时造成密钥信息和用户账户信息的泄露。比较安全的做法是将 APP 敏感业务数据存储在云端, 必须存储本地的数据需要加密。

此外, 2016 年以来, 美国密西根大学 Atul

Prakash 团队在信息安全顶级会议 S&P, USENIX, NDSS 上发表多篇研究成果[9,78,79]指出, 针对三星 IoT 智能设备平台 SmartThings, 在用户控制终端 APP 上存在业务过度授权, 敏感数据缺乏保护等漏洞, 在云服务端则存在 API 访问控制不安全等缺陷。作为三星公司智能家居平台的中枢, SmartThings 是一个可扩展的物联网平台, 允许用户购买和添加更多家庭智能设备, 然后通过配套的 SmartThings 应用 SmartApps 或制造商提供的单独 APP 来控制连接的设备。研究人员对该智能设备平台用户控制终端 499 款 SmartApps 和 132 款设备进行了一系列静态代码分析, 发现尽管 SmartThings 部署特权分离模型, 但还是存在提权漏洞, 也就是 SmartApps 能够对设备执行比其原定功能还要多的操作。这些漏洞将允许恶意应用解锁门禁, 远程修改智能门禁密码, 错误触发烟雾警报器以及使智能设备进入假日模式。

针对这些问题, Atul Prakash 团队研究人员首先提出了 FlowFence[78]数据流保护方案, 监测智能设备用户控制终端 APP 对用户隐私数据使用情况。传统关于隐私数据保护的安全方案只支持基于权限的访问控制, APP 获得权限之后如何使用并无有效方案。FlowFence 架构如图 10 所示, 方案通过沙盒隔离模块(*Quarantined Modules, QM*)和污点追踪处理模块(*opaque handles*), 在可信服务内通过 *TrustAPI* 等方法, 基于用户预定义规则(Flow Policy)来阻止恶意业务层数据流。只有符合预定义规则的数据流才被允许, 从而保证 APP 对隐私数据合理使用。

此外, 他们提出了一套基于上下文语义的用户控制终端授权系统 ContextIoT[79], 此方案在运行时给用户提供细粒度的上下文信息提示, 以帮助用户施行有效的访问控制。ContextIoT 将“上下文”定义为程序间控制流和数据流的执行路径。为了帮助用户做出更明智的决策, 此方案还使用污染分析来标记运行时数据的数据源。为了提供向后兼容性, ContextIoT 提供了一个应用程序修补机制, 可将现有的 IoT 应用程序转换为 ContextIoT 兼容的应用程序。

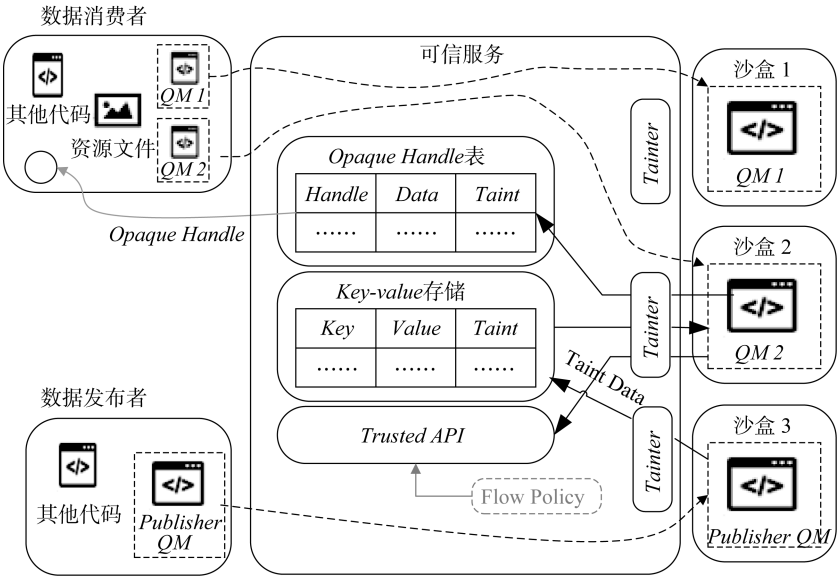

图 10 FlowFence 数据流保护方案

Figure 10 FlowFence data stream protection scheme

ContextIoT 具体案例如下图 11 所示, 每次 APP 尝试针对敏感数据进行操作(如开窗), 请求许可(*Requesting Permission*)会发送到后台, 基于云端的后台授权系统将检查这个上下文操作是否之前被允许或者拒绝过。如果是新的上下文, 系统将把这个请求许可和上下文推送给用户, 由用户进行决策, 之后新决策作为新安全策略添加到后台授权系统。

通过上述分析可以看出, 针对业务层数据流的安全分析已经开始受到关注, 但仅针对隐私泄露和权限控制在用户控制终端内进行检测。IoT 智能设备业务部署和数据通路除了涉及用户控制终端, 还包括硬件端和云服务端, 它们为攻击者在业务层数据上提供更大范围攻击面和漏洞入口, 目前缺乏在全局视图上针对业务层数据攻击的安全研究和分析。

3.3 云服务端安全威胁和防护技术

在智能设备业务生态三端系统架构中, 除了针对智能设备终端和用户控制终端的安全威胁, 在云服务端也存在安全威胁。黑客利用业务层的漏洞, 单

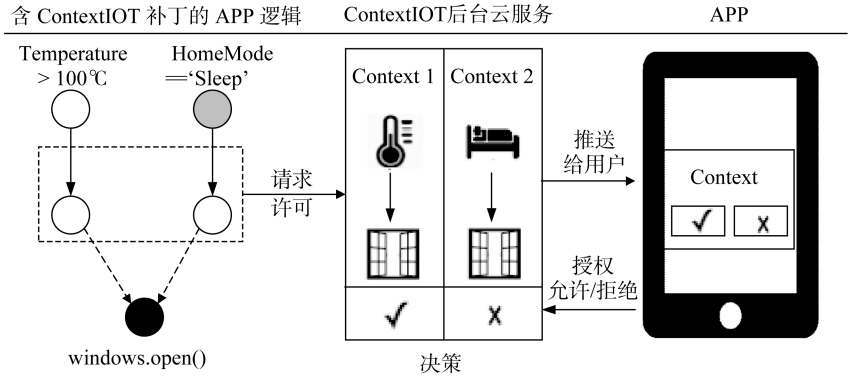

图 11 ContexIoT 实施基于上下文访问控制案例

Figure 11 ContexIoT implementation of a context based access control case

点攻破设备, 批量控制厂商设备等攻击行为正愈演愈烈, 并逐步成为针对智能设备安全威胁的主要来源。调查发现 90%的家电厂商的智能设备与自建的私有云之间没有任何身份认证以及加密协议[80]。赛门铁克公司 2015 年 3 月发布的一项报告显示, 在随机抽样的 50 种智能家居产品中, 大部分存在着类似的基本安全问题[81], 包括脆弱的身份认证和网络攻击控制, 其中的设备包括智能锁, 智能电灯, 智能烟雾警报器, 智能家庭影院等。因此各个智能设备厂商云服务端的安全性问题也日益凸显出来。本章将从传统网络层安全威胁、业务层接入协议安全漏洞、业务层设备权限管理安全威胁和业务层设备批量攻击威胁四个方面总结云服务端面临的安全问题, 之后从安全监测角度综述目前已有安全研究。

3.3.1 云服务端安全威胁

(1) 云平台传统网络层安全威胁

智能设备云服务端接入到互联网之中, 由于物联网对于互联网的天然继承性, 针对互联网的传统网络层安全攻击也自然地蔓延到了物联网之中, 因此云服务端首先面临的就是传统网络层安全威胁。虽然业界和学术界在网络层安全研究深耕多年, 但是攻防相长的态势没有变化, 依然需要引起智能设备厂商以及智能设备安全研究者的关注。

网络层面临的威胁主要包括物理威胁、病毒、非授权访问、蠕虫、系统漏洞、暴力破解、后门(木马程序)、网络嗅探、拒绝服务攻击、内部人员误操作等[82]。常见攻击手段包括恶意程序(如病毒、蠕虫和木马程序)、DDoS、欺骗(如 IP 欺骗[83]、ARP 欺骗[84])、邮件炸弹[85]、口令破解、社会工程[86]等。

根据全球开源安全组织 OWASP 在 2014 年发布的物联网十大安全威胁研究报告[87], 排名第一的物联网安全威胁就是不安全的网络层 Web 接口, 主要包括账户枚举、脆弱的默认密码、暴露在网络流量中的凭据、跨站脚本攻击(XSS)、SQL 注入和长连接 Session 管理等问题。此外, 其他网络层威胁还包括不安全的网络服务(如缓冲区溢出、开放端口和拒绝服务攻击等)、缺乏安全配置能力(如细粒度权限模型的缺乏、缺乏安全监控和缺乏安全日志等)。

(2) 云服务端业务层接入协议安全威胁

目前在 IoT 智能设备云服务端业务层接入协议多种多样, 各大厂商根据自身业务需求直接沿用或简单修改现有互联网通信协议。由于协议开发初期对安全的忽视或是对协议生硬的移植, 使得业务层通信安全变得非常脆弱。常见主流的业务层接入协议有 HTTP/REST[88]、MQTT[89]、XMPP[90]、CoAP[91]等, 这些协议身份认证机制对比如下表 2 所示。下文先分别总结这些协议和安全威胁, 最后归纳业务层接入协议整体的安全威胁。

表 2 四种主流业务层接入协议身份认证机制对比

Table 2 Comparison of four mainstream business layer access protocol identity authentication mechanisms

协议	身份认证机制	分析
HTTP /REST	明文传输基本认证	需要通过 TLS 协议的支持来进行认证和可信服务
MQTT	用户名/密码	MQTT 允许发送用户名和密码进行身份认证, 但都明文发送, 因此安全使用 MQTT 必须采用 TLS
XMPP	多种选项	通过 TLS 来加密通信并通过简单身份验证与安全层(Simple Authentication and Security Layers, SASL)实现身份验证机制
CoAP	preSharedKey rawPublicKey certificate	CoAP 提供设备间多种身份认证方式, 需要使用数据传输层的安全传输协议(D-TLS)

在 HTTP/REST 协议方面, 目前很多厂商云端建设仅仅使用简单的 WEB 程序接口的形式使设备联网,

但设备又必须主动向服务器发送数据, 难以主动向设备推送数据, 因此加入 Websocket 来解决这一问题, 这样云服务端的安全将变的非常脆弱。原始 HTTP 请求报文是明文的, 为了安全一般需要加入传输层安全协议[92](Transport Layer Security, TLS)来保证通信双方的身份认证和信道保密性。通过研究, 我们发现厂商 A 和厂商 B 均直接采用了普通 HTTP 协议, 没有利用 SSL/TLS 加密, 这就导致敏感通信数据极易被拦截、获取和篡改。在数据内容保密性方面, 厂商 A 的 APP 与服务端传输的所有数据内容都是明文, 没有任何加密手段。而厂商 B 设备与云端的通信数据虽然采用了 AES 对称加密使得安全性得到了一定的提高, 但是对用户控制终端 APP 进行反编译就可以轻而易举地从代码中获得 AES 对称密钥, 从而可以对通信数据内容进行破解。另外两家厂商 C 和 D 安全性则相对较好, 采用了 SSL/TLS 协议对 HTTP 协议进行加密保护, 但在 SSL 认证处理上皆采用了单向认证, 即客户端验证服务器身份, 服务器不验证客户端身份的方式, 使得它们仍然不能防止中间人攻击。

MQTT[89]是 IBM 开发的即时通讯协议, 相较于重量级的 HTTP 协议它可以通过很少的代码和带宽远程连接设备, 适合于 CPU 和内存资源受限的嵌入式设备。MQTT 是一种基于代理的发布/订阅的消息协议, 提供一对多的消息分发, 一个发布者可以对应多个订阅者, 当发布者发生变化的时候, 他可以将消息一一通知给所有的订阅者, 这种模式提供了更大的网络扩展性和更动态的网络拓扑[93]。MQTT 基于 TCP 协议, 默认情况通讯并不加密, 出于成本的考虑有些使用了该协议的智能设备产品没有添加额外的安全措施, 这就使得一切信息暴露无遗, 造成极大的安全威胁。因此出于安全的考虑使用 TSL 几乎是必须的选择。

XMPP 作为一种即时聊天协议, 和现有物联网设备通信的模式非常相似。XMPP 包含了针对服务器端的软件协议, 使之能与另一个进行通话, 这使得开发者更容易建立客户应用程序或给一个已经配置好 XMPP 协议的系统添加功能[94]。XMPP 的开放性和易用性, 使开发者能够很快的应用与现有物联网环境, 具有成本低和开放快的特点。但是由于开发者的水平参差不齐, XMPP 协议在传输数据加密处理、云服务端用户鉴权隔离和用户认证信息存储等诸多方面容易造成安全隐患。在 2015 年 Hackpwn 大会上安全人员就展示了对 TCL 洗衣机(使用了 XMPP 协议)的破解过程[95]。TCL 智能洗衣机通过配置直接接入互联网中, 它使用的是“京东微联”平台, 京东微联是京东旗下针对智能设备的公有云平台。通过安全人员的分析, 在京东微联和用户控制终端之间都具备高强度的加密传输协议和身份认证方案, 但是洗衣机和京东微联之间使用的是未加密的 XMPP 协议来实现会话控制和设备长连接, 可以被嗅探者清晰地看到控制指令格式和内容。

CoAP[91] (Constrained Application Protocol)的设计目的在于允许资源相对有限的设备利用 UDP 而非 TCP 通过互联网实现通信, 开发人员可以和任意支持 CoAP 的设备进行交互, 具体方式与采用传统 REST API 的设备完全一致[96]。CoAP 的主要适用场景包括低功耗传感器以及需要通过互联网加以控制的设备。所有 CoAP 消息皆可被标记为“确认”或者“未确认”, 并作为应用层级的 QoS 机制。SSL/TLS 加密无法在 UDP 之上实现, 因此 CoAP 需要使用数据传输层安全作为替代[94]。

通过对主流业务层接入协议的总结和安全威胁阐述, 结合本研究工作前期对智能设备产品业务层接入协议相关调研, 我们归纳得出业务层接入协议整体面临的安全威胁有如下四个部分, 包括窃听攻击、篡改攻击、伪造攻击和重放攻击:

1) 窃听攻击: 攻击者通过窃听网络通信链路上的数据, 获取智能设备终端的隐私数据和敏感数据。例如, 通过被动窃听网络可以获取健康医疗类智能硬件设备的数据, 从而获得用户的隐私数据。

2) 篡改攻击: 攻击者通过主动插入到通信链路上, 对数据的完整性进行攻击, 即篡改数据报文的内容, 例如, 攻击者可以篡改智能设备向业务云提交的业务数据, 向业务云提供错误的数据信息。

3) 伪造攻击: 即一种针对真实性的攻击, 攻击者伪造一些看似来自合法来源的数据报文, 以此诱骗通信的另一方。尤其是在智能设备的控制指令传输过程中, 攻击者可以伪造一些携带控制指令的报文发给智能设备, 从而达到未授权操作的效果。

4) 重放攻击: 从目前的分析来看, 重放攻击主要分为两个方面, 一个是针对未加密的通信协议重放控制指令攻击设备, 另一个是对会话令牌的重复使用而发起的攻击行为。

通过研究, 我们发现厂商 A 没有时间戳机制, 任何信息都可以重放; 厂商 B 和 C 虽然有时间戳, 但服务端并没有对时间戳进行检验, 过时很久的信息依旧可以重放; 厂商 D 在这个问题上做得相对较好, 服务端会对客户端提交上来的时间戳与自己的时间进行对比, 只有波动在很小范围内的时间戳所标识

的请求才会被响应。但依然不是严格的抗重放机制, 短时间内的请求仍旧可以实现重放; 而在云服务端常用会话令牌设计中, 当客户端用户登录成功以后, 服务端会返回给客户端一个会话令牌, 标识了此次会话用户的身份及权限, 正常情况下会话令牌应及时更新。我们研究发现厂商 B 的处理方法是每次登录就会产生一个新的会话令牌, 厂商 A、厂商 D 也对会话令牌进行了控制, 有一定的生存时间, 超过这个时间, 就要申请一个新的令牌进行通信, 同时旧的令牌就会失效; 而厂商 C 虽然也会更新会话令牌, 但旧的令牌不会失效, 即只要存在过的会话令牌永久有效。在这种情况下, 只要攻击者拿到用户任何一个使用过的会话令牌, 都可以冒充用户身份对设备进行操作。

(3) 云服务端业务层内容安全威胁

在智能设备云服务端, 除了接入协议的安全威胁, 在业务层内容安全上也存在着隐患, 而且近些年业务层安全漏洞正越来越多的吸引黑客注意, 引发了很多业务层内容相关的异常行为, 而以往相关安全研究工作较少。本文拟先建立业务层异常行为分类模型, 对业务层内容安全威胁进行系统梳理。

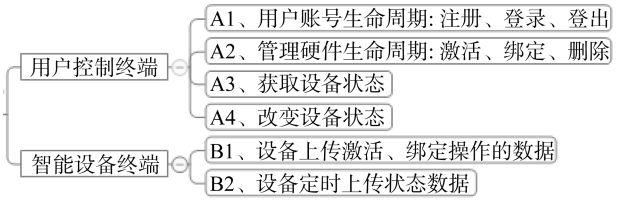

图 12 智能设备业务云数据内容分类

Figure 12 Intelligent device service cloud data content classification

首先, 业务层内容按照数据类别划分, 可分为用户控制终端数据和智能设备终端数据, 如下图 12 所示。其中, 用户控制终端数据包括用户账号生命周期数据, 如用户注册、用户登录和用户登出; 管理设备生命周期数据, 如设备激活、设备绑定和设备删除; 获取设备状态和改变设备状态。智能设备终端数据包括设备上传激活和绑定操作的数据, 以及设备定时上传状态数据。

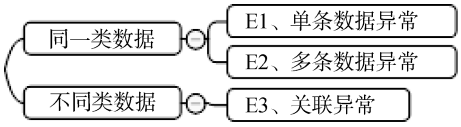

图 13 智能设备业务云数据内容异常行为分类域

Figure 13 Intelligent device business cloud data content exception behavior classification domain

然后, 按照异常行为涉及的数据类型是否相同, 我们将业务层内容安全异常行为分为如下图 13 所示的两个类别, 对应同类数据异常域和不同类数据异常域, 分别如表 3 和表 4 所示。

云服务端业务层出现漏洞之后, 由于智能设备数量大、影响面广, 极易引起黑客的注意而发起批量攻击。根据《Verizon 2016 数据泄露报告》中的数据统计[97], 有超过一半的企业网络安全事故和身份认证凭据盗用有关。授权和身份认证大部分是由云服务端进行控制的, 云服务端会存在用户安全校验简单、设备识别码规律可循、设备间授权不严等安全问题。目前可以在分析出设备身份认证标识规律的情况下, 如 MAC 地址、SN 号等都可以通过猜测、枚举的方式得到, 从而批量控制大量设备。这个漏洞根本上是设备唯一标识命名规则可猜测, 此危害在智能设备系统里是最大的, 因为它能够影响到全部的智能硬件。360 安全研究人员在《黑客告诉你, 315 晚会上的智能产品是怎样被黑的》[98]一文中就提到, 在发现能够控制到智能硬件自身之后, 黑客就会想

表 3 同类数据异常行为域

Table 3 Similar data anomaly domain

E	A1	A2	A3	A4	B1	B2
E1 单条	登录出错	暂无	非授权	1、非授权 2、改变设备状态至非法值	暂无	传感器状态非法值
E2 多条	频繁登录登出	批量绑定	批量扫描	1、单用户频繁控制某单个设备 2、单用户控制多个同型号设备 3、同型号批量设备被设置成同状态	暂无	设备频繁上传状态(如频繁重启等)

表 4 不同类数据异常行为域

Table 4 Dissimilar data anomaly domain

A1	/					
A2	暂无	/				
A3	暂无	暂无	/			
A4	暂无	暂无	暂无	/		
B1	暂无	无效非法绑定	暂无	暂无	/	
B2	暂无	暂无	暂无	暂无	设备未知异常	/
E3	A1	A2	A3	A4	B1	B2

办法看看能不能横向影响到其他设备。横向的控制就是去看设备的身份认证标识是怎么设计的。很多智能硬件厂商是拿 MAC 地址作为身份认证的唯一标识, 然而 MAC 地址的规律是可循的, 所以可以通过遍历 MAC 地址控制这个厂家的所有设备。本文针对厂商 A 和厂商 B 的研究分析也验证发现了同样的问题, 设备标识命名规则可猜测从而造成大批量的攻击行为。

3.3.2 云服务端信息流异常检测技术

云服务端信息流则包含传输过程中网络协议数据组成的网络层数据流, 以及硬件端上传到云服务端的状态数据和用户控制指令数据组成的业务层数据流, 它们都表现为传输态。传输态信息流的攻击行为将导致隐私信息泄露、硬件非法控制、硬件恶意批量攻击等危害。目前研究主要关注传统网络层数据流攻击防御和检测, 如入侵检测、抗 DDoS 攻击等技术。文献[99]提出了针对无线传感网络(Wireless Sensor Networks, WSN)的异常检测, 通过采集 WSN 设备中日志记录和活动行为, 在云服务端采用入侵检测的方式发现异常行为。文献[100]和[101]则针对 IoT 特定的网络协议(如 RPL, 6LoWPAN 和 CoAP 协议等), 结合入侵检测技术对协议交互数据进行异常检测。然而, 目前针对 IoT 智能设备的攻击越来越多的来自业务层数据流[102], 而针对业务层数据流的安全分析才刚刚起步, 且仅针对隐私泄露问题在用户控制终端内进行检测。IoT 智能设备业务部署和数据通路除了涉及用户控制终端, 还包括硬件端和云服务端, 它们为攻击者在业务层数据上提供更大范围的攻击面和漏洞入口, 目前缺乏在全局视图上针对业务层数据攻击的安全研究和分析。

4 思考和讨论

通过本文对 IOT 智能设备安全威胁的整体梳理, 可以发现智能设备安全问题在智能设备终端、云服务端和用户控制终端均存在。物联网和 IT 技术的融合使得智能设备具备了联网和交互的能力, 和传统 PC 主机相比, 智能设备具有数量多、计算资源少、防护能力低等特点。从信息流角度分析, 用户控制终端下达到智能设备终端的控制指令经历了从信息空间到物理系统操作空间的跨界转换, 这是智能设备安全问题的本质, 它导致了访问控制、权限管理和安全操控等多方面的安全威胁。

在目前已有的安全防护技术方面, 基于上述章节论述可以看出: 现有IoT智能硬件安全防护技术大多集中于安全事件处理的某个阶段(例如事前预防、运行时监测阶段), 或仅针对具体问题(例如完整性验证、数据流异常检测), 没有从全局视角对 IoT 智能硬件安全防护体系进行建模, 缺乏涵盖智能硬件安全事件全过程的安全防护体系。

本小节下面的内容将从威胁防范和运行保障视角, 讨论针对IoT智能设备信息安全事件全生命周期过程的防护体系, 如下图 14 所示, 涵盖设备于 1)“事前预防”阶段抗脆弱性加固的群体完整性认证和固件布局多样性技术; 2) “运行监测”阶段信息流动态安全监测的控制流数据流融合度量以及基于网络层业务层进行异常行为检测技术; 3)“事后恢复”阶段受损状态下安全保障的应急处置和设备愈合技术, 提供相关技术的路线与读者进行探讨。

4.1 抗脆弱性技术

(1) 完整性群体快速认证技术

IoT 智能硬件由于其部署分散, 网络环境复杂, 验证者通过网络对批量设备的软件完整性进行远程验证更为适用于这类场景。其中, 设备规模化、资源受限、网络结构不明确且动态变化、问题设备定位都是设计认证方案过程中需要考虑的问题。针对上述问题, 本文将从设备入网、断言生成、断言收集、断言验证等四个方面讨论实现支持星型网络、树型网络、P2P 网络在内的异构网络下智能硬件完整性群体的快速认证技术。

技术路线如下: 设备入网基于信任预置及轻量级密钥协商策略, 实现新增入网设备与现有网络信任关系的初始化与建立, 同时可以构建网络拓扑结构树, 防止恶意设备加入对网络或者协议进行攻击; 断言生成在智能硬件端, 利用校验代码对设备软件进行校验和计算, 并根据安全策略生成及发送校验结果, 安全策略的实施可以保证当前参与设备为验证者需要校验的目标设备, 同时智能硬件可以根据策略内正常状态校验值集合判断自身状态, 实现特定群体设备的完整性感知; 断言收集利用分布式计

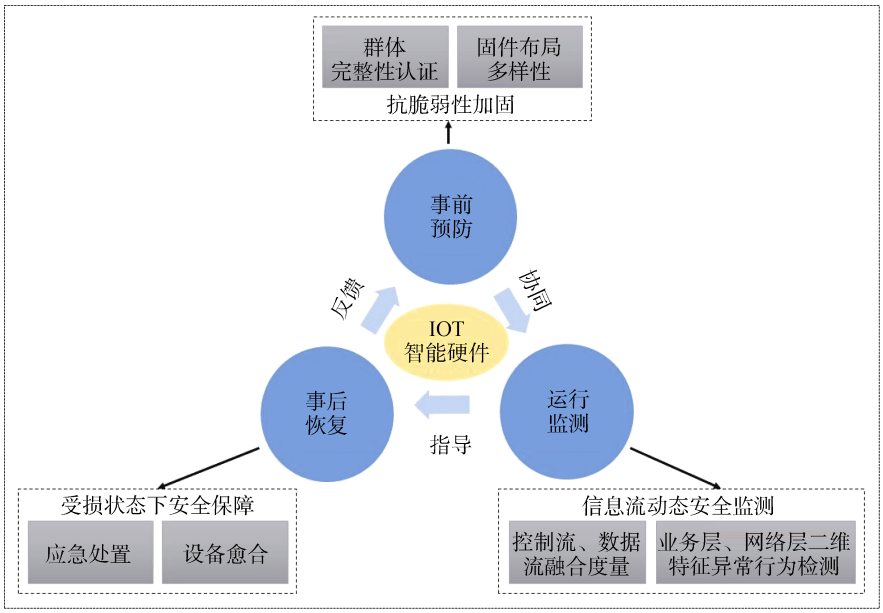

图 14　IoT 智能硬件安全防护模型

Figure 14　IoT intelligent hardware security protection model

算思路，设备能对其拓扑结构树上子节点提交的校验结果进行规约并提交至上级节点，并逐级规约形成最终验证结果，通过规约的方式将校验结果的计算分散至各个节点；断言验证则是验证者利用信任体系对规约后的验证结果进行处理，得出当前网络内设备完整性状态，并得出非正常状态设备集合。

(2) 设备固件多样性技术

传统强计算能力设备系统随机化方案基于内存管理单元(Memory Management Unit, MMU)的支持，在部署后针对程序加载以及执行过程进行随机化。然而智能硬件软硬件资源有限，直接使用物理内存执行代码，没有 MMU 的支持也无法进行复杂计算，无法利用自身计算资源进行随机化。本文将讨论利用云端高计算能力，部署多样性引擎对智能硬件固件内部数据结构和指令进行随机化的方法。

技术路线如下：对待随机化固件镜像进行递归和线性逆处理，分离出镜像文件内所有数据结构与指令，并分别对数据结构和指令进行分析以及布局随机化，在数据层面获取数组、结构体变量等信息，对其按照算子指定大小添加冗余，异构化栈内数据布局。在指令层面根据随机算子获取上下文无关指令，在不影响程序正常执行的情况下进行上下文语句的随机重排，异构化程序语句布局。最后对随机化后的固件代码进行重打包，生成拥有标记的异构化固件并进行部署，达到固件的随机化目的。

4.2　运行时信息流动态安全监测

(1) 控制流与数据流融合度量

智能硬件运行时系统态信息流包含控制流和数据流，传统的检测方案只涉及其中之一，没有充分考虑到这两者关联性以及相互影响。如控制流检测方案仅对控制流完整性进行检测，忽略数据流对控制流的指引，在数据流被篡改的情况下无法发现程序未按预设逻辑执行。这些无交集融合技术，只能检测已知的系统态信息流异常，无法发现上述数据流间接攻击导致的程序运行失控。对系统内部数据的分析可以同时从控制流和数据流两方面入手，对采集的数据利用机器学习分类算法对智能硬件系统态的数据流和控制流进行关联性分析，建立融合度量模型，提升追踪数据流间接攻击的检测能力。

融合度量可以分为数据采集以及分析检测两个阶段。在信息流采集阶段，需要在程序运行时获取数据流和控制流的标记信息，并把这些信息进行持续的记录直到待检测路径上所有的信息流标记数据收集完成，形成检测路径所对应程序段的数据流/控制流完整标记信息集，并分别形成数据流和控制流报告，用于进行融合检测。

在系统态信息流分析检测阶段，对已形成的控制流和数据流报告，进行数据匹配融合，生成针对整体信息流的数据集，根据历史检测记录对信息流数据集进行异常与非异常标注，形成信息流标注数据集并利用机器学习分类算法训练其相应的度量模型。最后，输出系统数据流、控制流融合异常报告，

标识待测系统路径的安全状态。

(2) 基于网络层和业务层的异常行为检测

智能硬件运行时传输态信息流包含网络层数据流和业务层数据流。在网络层, 特征属性来源包括基于主机的数据(如进程标识、系统调用), 以及基于网络协议的数据(如流量、源地址/目的地址 IP、端口号、协议类型); 在业务层, 特征属性来源包括硬件端和云服务端交互的激活绑定数据和硬件状态数据, 以及用户控制端和云服务端交互的用户生命周期管理、设备生命周期管理、设备操作指令等数据。现有研究大多从网络层数据流开展安全检测, 在业务层仅关注用户控制端内部隐私数据保护, 无法从业务层数据流全局视角进行检测。

针对上述问题, 本文认为从网络层数据流和业务层数据流两方面进行安全检测将是未来检测技术的发展趋势。通过对网络层和业务层的二维特征矩阵建模, 在网络层单维度、业务层单维度、以及网络层和业务层双维度发现 IoT 智能硬件异常行为。在技术路线上可以基于数据采集、批量数据预处理, 利用数据挖掘技术对异常行为进行发现, 并结合模式匹配、分类和聚类的方法, 对异常行为进行检测。整个过程包含数据预处理、已知异常模式匹配检测、已知异常行为分类检测和未知异常行为聚类检测。

在数据预处理阶段, 将对数据进行规范化处理, 并进行特征选取, 形成二维特征矩阵; 对简单规则可定义的已知异常行为, 使用模式匹配方法, 在网络层向量维度或业务层向量维度进行快速发现; 针对复杂的异常行为, 可以采取分类或者聚类的方法进行交叉发现, 分类检测能发现已有训练模型的异常行为, 而对于规则未知的异常, 则需要借助聚类方法检测。在整个路线中, 异常挖掘过程可以采取渐进式以减少不同异常发现方法之间的影响。

4.3 受损状态下安全保障技术研究

(1) 智能硬件受损条件下应急处置

当设备计算资源遭到攻击时, 对设备进行应急处置, 保证设备核心功能正常工作是保证设备可用性的基础。可以采用在操作系统任务调度器内部添加钩子并指向位于安全域的安全代理的方式, 对准备加载的任务进行安全审计, 构筑安全隔离抽象层对资源进行隔离审计。待运行任务在被调度器拷贝到主系统任务上下文空间进行执行前, 需要经过安全代理的审核决定是否加载。

例如, 实时操作系统中通常运行一个主系统任务, 可以访问所有的硬件, 其他一些子任务可以控制不同的硬件资源进行工作, 而这些子任务的执行需要主系统任务使用调度器把它们加载到其上下文空间才可以实现。在子任务执行之前, 可以设置安全代理用来阻止调度器拷贝受到攻击的子任务到主系统任务的上下文环境中, 从而阻止攻击者获取响应传感器资源的控制权限。

(2) 智能硬件受损条件下愈合恢复

当系统关键路径受到攻击而失去作用时, 需要系统具备功能性恢复的能力。本文认为一种可行的思路可以基于微重启技术和递归恢复思想进行恢复。系统以功能相关性对各个计算单元进行树型建模, 树中每一个节点表示为系统的计算单元, 以树状层次结构描述资源的逻辑调用关系, 形成系统功能资源树。系统恢复将以资源树中需要恢复的模块为根节点, 递归恢复此子树内所有的模块。例如, 可以对操作系统内的计算资源按照功能逻辑进行划分, 即操作系统内计算资源资源池包含 RTOS、Socket、WLAN 等孩子资源, 为操作系统计算资源与下设的几个资源建立了功能上的“父子”关系。当判定某个子任务的控制流遭到破坏, 则使用安全代理对该子任务进行“阻断”, 剥夺其运行与访问其他计算资源的权限。同时从云端远程更新该子任务以及其在树状结构上所有孩子节点的上下文内容, 更新重启后才可继续使用。

5 总结

近些年来, 伴随着物联网的产生和发展, IoT 智能设备越来越多地出现在市场上, 深入到人们工作和生活各个方面, 然而智能设备大规模普及的同时, 也给用户个人资产与隐私保护带来了极大地冲击和挑战。本文首先基于智能设备终端、云服务端和用户控制终端系统架构, 综述目前智能设备安全威胁的主要来源和技术攻击手段, 并针对性地梳理已有安全研究和防护技术现状。针对现有 IoT 智能设备安全防护体系缺失、安全设计不足的问题, 本文对 IoT 智能设备在安全防护全生命周期下的能力需求进行了讨论, 提出了系统防护模型设计思路。

致谢 衷心感谢各位评审专家对本文提出的宝贵意见。向所有对本文的工作给予建议的老师、参与本文工作的同学表示感谢。本研究由中国科学院青年创新促进会(课题号: 1105CX0105)和信息安全国家标准项目“信息安全技术-智能互联设备信息安全技术要求”提供资助。

参考文献

[1] 中国信息通信研究院. 移动智能终端暨智能硬件白皮书(2016 年) [EB/OL]. 2017 [2017-3-8]. http://main.catr.cn/kxyj/qwfb/bps/201610/P020161027525277225001.pdf.

[2] 中国通信网. 工信部乔跃山: 2020 年我国智能硬件市场规模将达到万亿元 [EB/OL]. 2017 [2017-3-8].

[3] Čeleda P, Krejčí R and Krmíček V. "Flow-based security issue detection in building automation and control networks." *Meeting of the European Network of Universities and Companies in Information and Communication Engineering*. Springer Berlin Heidelberg, 2012, pp. 64-75.

[4] Dev J A. "On the Imminent Advent of Botnet Powered Cracking." *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference on*. IEEE, 2016, pp. 188-195.

[5] Sochor T, Zuzcak M and Bujok P. "Statistical analysis of attacking autonomous systems." *Cyber Security And Protection Of Digital Services (Cyber Security), 2016 International Conference On*. IEEE, 2016, pp. 1-6.

[6] 李柏松. 物联网僵尸网络严重威胁网络基础设施安全 [J]. 信息安全研究, 2016, 2(11), pp. 1042 –1048.

[7] 51CTO. 威胁 IoT 环境的 Linux 恶性代码 TOP5[EB/OL]. 2017 [2017.3]. http://netsecurity.51cto.com/art/201701/529274.htm.

[8] FREEBUF. 西班牙智能电表中惊现漏洞, 可导致大面积停电 [EB/OL]. 2017 [2017.3]. http://www.freebuf.com/news/47634. html.

[9] Fernandes E, Jung J and Prakash A. "Security analysis of emerging smart home applications." *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 2016, pp. 636-654.

[10] Miller C, Valasek C. Blackhat 2015: Advance Can Injection Techniques for Vehicle Networks [EB/OL]. 2017 [2017.3]. https://www.blackhat.com/us-16/briefings/schedule/#advanced-can-injection-techniques-for-vehicle-networks-4149.

[11] 360. hackpwn2016 [EB/OL]. 2017 [2017.3]. http://hackpwn.360.cn/2016/index.html.

[12] Alves T and Felton D. "Trustzone: Integrated hardware and software security." *ARM white paper*, 2004, 3(4), pp. 18-24.

[13] Koeberl P, Schulz S, Sadeghi A R, et al. "TrustLite: A security architecture for tiny embedded devices." *Proceedings of the Ninth European Conference on Computer Systems*. ACM, 2014, pp. 10.

[14] Brasser F, El Mahjoub B, Sadeghi A R, et al. "TyTAN: tiny trust anchor for tiny devices." *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*. IEEE, 2015, pp. 1 –6.

[15] Eldefrawy K, Tsudik G, Francillon A, et al. "SMART: Secure and Minimal Architecture for (Establishing Dynamic) Root of Trust." NDSS, 2012, 12, pp. 1-15

[16] IDC. Worldwide Quarterly Wearable Device Tracker[EB/OL]. 2017 [2017.3]. http://www.idc.com/tracker/showtrackerhome.jsp

[17] SAMSUNG. SmartThingsHub[EB/OL]. 2017 [2017.3]. https://shop.smartthings.com/.

[18] GOOGLE. Goole Home [EB/OL]. 2017 [2017.3]. https://madeby.google.com/home/.

[19] RESEARCH, MARKETS. Forcast[EB/OL]. 2017 [2017.3]. http://www.researchandmarkets.com/.

[20] SAMSUNG, SmartThings [EB/OL]. 2017 [2017.3]. https://www.smartthings.com/.

[21] APPLE, HomeKit [EB/OL]. 2017 [2017.3]. http://www.apple.com/ios/home/.

[22] Google Weave Project. https://developers.google.com/weave/.

[23] ALIBABA, 阿里智能云平台 [EB/OL]. 2017 [2017.3]. http://open.aliplus.com/.

[24] TENCENT. 腾讯物联云平台 [EB/OL]. 2017 [2017.3]. http://iot.open.qq.com/.

[25] 环球网. 史上最大规模网攻？大半个美国网瘫美国土安全部及 FBI 介入 [EB/OL]. 2017 [2017.3]. http://world.huanqiu.com/exclusive/2016-10/9587047.html.

[26] DYN. Dyn Statement on 10/21/2016 DDoS Attack[EB/OL]. 2017 [2017.3]. http://dyn.com/blog/dyn-statement-on-10212016-ddos-attack/.

[27] ZELJKA ZORZ H. Researcher ropes poorly protected devices into botnet to map the Internet [EB/OL]. 2017 [2017.3]. https://www.helpnetsecurity.com/2013/03/20/researcher-ropes-poorly-protected-devices-into-botnet-to-map-the-internet/.

[28] 搜狐科技. IoT 僵尸的前世今生 [EB/OL]. 2017 [2017.3]. http://mt.sohu.com/20161102/n472136915.shtml.

[29] 海康威视官方平台. 海康威视就部分监控设备遭网络攻击的后续说明 [EB/OL]. 2017 [2017.3]. http://www.asmag.com.cn/news/201503/78215.html.

[30] 绿盟科技. Mirai 代码及原理分析 [EB/OL]. 2017 [2017.3]. http://www.secjia.com/report/Mirai-Code-and-DDoS-analysis.pdf.

[31] VULNERABILITIES C, EXPOSURES. CVE-2013-2678 [EB/OL]. 2017 [2017.3]. http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-2678.

[32] 安全客. 新型木马: 能够感染 Linux 路由器 [EB/OL]. 2017 [2017.3]. http://bobao.360.cn/news/detail/1882.html.

[33] ANDROID 安全中文站. 小谈智能硬件安全 [EB/OL]. 2017 [2017.3]. http://www.droidsec.cn/.

[34] VULNERABILITIES C, EXPOSURES. CVE-2011-4859[EB/OL]. 2017 [2017.3]. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-4859.

[35] DEV/TTYS0. Hacking the D-Link DSP-W215 Smart Plug [EB/OL]. 2017 [2017.3]. http://www.devttys0.com/2014/05/hacking-the-d-link-dsp-w215-smart-plug/.

[36] 腾讯安全应急响应中心. 潜伏在身边的危机: 智能设备安全 [EB/OL]. 2017 [2017.3]. https://security.tencent.com/index.php/blog/msg/94.

[37] Abdul Afiq Arifin K. "Developing Mesh Network To Extend Wifi Coverage Area." 2014.

[38] Ahamed S S R. "The role of zigbee technology in future data communication system." *Journal of theoretical and applied information technology*, 2009, 5(2), pp. 129.

[39] Noorman J, Agten P, Daniels W, et al. "Sancus: Low-cost Trustworthy Extensible Networked Devices with a Zero-software Trusted Computing Base." *USENIX Security*. 2013, pp. 479– 494.

[40] Asokan N, Brasser F, Ibrahim A, et al. "Seda: Scalable embedded device attestation." *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 964 – 975.

[41] Ngabonziza B, Martin D, Bailey A, et al. "TrustZone Explained: Architectural Features and Use Cases." *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference on*. IEEE, 2016, pp.445 – 451.

[42] Vasudevan A, Mccune J, Newsome J, et al. “CARMA: A hardware tamper-resistant isolated execution environment on commodity x86 platforms.” *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*. ACM, 2012, pp. 48 – 49.

[43] Sumrall N and Novoa M. “Trusted computing group (TCG) and the TPM 1.2 specification.” *Intel Developer Forum*. Vol 32, 2003.

[44] Schulz S, Sadeghi A R and Wachsmann C. “Short paper: Lightweight remote attestation using physical functions.” *Proceedings of the fourth ACM conference on Wireless network security*. 2011, pp. 109 –114.

[45] LI Y, McCune J M and Perrig A. “VIPER: verifying the integrity of PERipherals’ firmware.” *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 2011, pp. 3 –16.

[46] Kennell R and Jamieson L H. “Establishing the Genuinity of Remote Computer Systems.” USENIX security. 2003, pp. 21.

[47] Arbaugh W A, Farber D J and Smith J M. “A secure and reliable bootstrap architecture.” *Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on*. IEEE, 1997, pp. 65 –71.

[48] Parno B, McCune J M and Perrig A. “Bootstrapping trust in commodity computers.” *Security and privacy (SP), 2010 IEEE symposium on*. IEEE, 2010, pp. 414 – 429.

[49] Kong J, Koushanfar F, Pendyala P K, et al. “PUFatt: Embedded platform attestation based on novel processor-based PUFs.” *Proceedings of the 51st Annual Design Automation Conference*. ACM, 2014, pp. 1 – 6.

[50] Kovah X, Kallenberg C, Weathers C, et al. “New results for timing-based attestation.” *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 239 –253.

[51] Forrest S, Somayaji A and Ackley D H. “Building diverse computer systems.” *Operating Systems, 1997., The Sixth Workshop on Hot Topics in*. IEEE, 1997, pp. 67-72.

[52] Sliesarieva E G B. “Automated Methods for Creating Diversity in Computer Systems.” The University of New Mexico, 2005.

[53] Chew M and Song D. “Mitigating buffer overflows by operating system randomization.”. 2002.

[54] Bhatkar S, DuVarney D C and Sekar R. “Efficient Techniques for Comprehensive Protection from Memory Error Exploits.” *Usenix Security*. 2005.

[55] Giuffrida C, Kuijsten A and Tanenbaum A S. “Enhanced Operating System Security Through Efficient and Fine-grained Address Space Randomization.” *USENIX Security Symposium*. 2012, pp. 475-490.

[56] Kil C, Jun J, Bookholt C, et al. “Address space layout permutation (ASLP): Towards fine-grained randomization of commodity software.” *Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual*. IEEE, 2006, pp. 339-348.

[57] Rodes B. “Stack layout transformation: Towards diversity for securing binary programs.” *Proceedings of the 34th International Conference on Software Engineering*. IEEE Press, 2012, pp. 1543-1546.

[58] Trivedi K S, Vaidyanathan K and Goseva-Popstojanova K. “Modeling and analysis of software aging and rejuvenation.” *Simulation Symposium, 2000.(SS 2000) Proceedings. 33rd Annual*. IEEE, 2000, pp. 270-279.

[59] Patterson D, Brown A, Broadwell P, et al. Recovery-oriented computing (ROC): Motivation, definition, techniques, and case studies. Technical Report UCB//CSD-02-1175, UC Berkeley Computer Science, 2002.

[60] Candea G, Cutler J, Fox A, et al. “Reducing recovery time in a small recursively restartable system.” *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*. IEEE, 2002, pp. 605-614.

[61] Avizienis A and Chen L. “On the implementation of N-version programming for software fault tolerance during execution.” *Proc. IEEE COMPSAC*. 1977, 77, pp. 149-155.

[62] Horning J J, Lauer H C, Melliar-Smith P M, et al. “A program structure for error detection and recovery.” *Operating Systems*. 1974, pp. 171-187.

[63] Koo R and Toueg S. “Checkpointing and rollback-recovery for distributed systems.” *IEEE Transactions on software Engineering*, 1987 (1), pp. 23-31.

[64] Golding R and Borowsky E. “Fault-tolerant replication management in large-scale distributed storage systems.” *Reliable Distributed Systems, 1999. Proceedings of the 18th IEEE Symposium on*. IEEE, 1999, pp. 144-155.

[65] Jurkovic G and Sruk V. “Remote firmware update for constrained embedded systems.” *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on*. IEEE, 2014, pp. 1019-1023.

[66] Dong W, Liu Y, Chen C, et al. “R2: Incremental reprogramming using relocatable code in networked embedded systems.” *IEEE Transactions on Computers*, 2013, 62(9), pp. 1837-1849.

[67] Shafi N B, Ali K and Hassanein H S. “No-reboot and zero-flash over-the-air programming for wireless sensor networks.” *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012 9th Annual IEEE Communications Society Conference on*. IEEE, 2012, pp. 371-379.

[68] Cam-Winget N, Sadeghi A R and Jin Y. “Invited: Can IoT be secured: Emerging challenges in connecting the unconnected." *Design Automation Conference (DAC), 2016 53nd ACM/EDAC/IEEE*. IEEE, 2016, pp. 1-6.

[69] Enck W, Gilbert P, Han S, et al. “TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones.” *ACM Transactions on Computer Systems (TOCS)*, 2014, 32(2), pp.5.

[70] Bosman E, Slowinska A and Bos H. “Minemu: The world’s fastest taint tracker.” *International Workshop on Recent Advances in Intrusion Detection*. Springer Berlin Heidelberg, 2011, pp. 1-20.

[71] Newsome J and Song D. “Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software.” 2005.

[72] Schwartz E J, Avgerinos T and Brumley D. “All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask).” *Security and privacy (SP), 2010 IEEE symposium on*. IEEE, 2010, pp. 317-331.

[73] Abera T, Asokan N, Davi L, et al. “C-FLAT: control-flow attestation for embedded systems software.” *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 743-754.

[74] Abadi M, Budiu M, Erlingsson U, et al. “Control-flow integrity.”

Proceedings of the 12th ACM conference on Computer and communications security. ACM, 2005, pp. 340-353.

[75] Hu H, Shinde S, Adrian S, et al. “Data-oriented programming: On the expressiveness of non-control data attacks.” *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 2016, pp. 969-986.

[76] Fora P O. “Beginners guide to reverse engineering android apps.” *RSA Conference*. 2014, pp. 21 –22.

[77] Khalid H, Nagappan M and Hassan A E. “Examining the Relationship between FindBugs Warnings and App Ratings.” *IEEE Software*, 2016, 33(4), pp. 34 – 39.

[78] Fernandes E, Paupore J, Rahmati A, et al. “Flowfence: Practical data protection for emerging iot application frameworks.” *USENIX Security Symposium*. 2016.

[79] Jia Y J, Chen Q A, Wang S, et al. “ContexIoT: Towards Providing Contextual Integrity to Appified IoT Platforms.” 2017.

[80] INC. H. HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack[EB/OL]. 2017 [2017.3]. http://www8.hp.com/us/en/hp-news/press-release.html?id=1744676#.WMEH1YGGO01.

[81] Mario B and Candid W. “Insecurity in the Internet of Things.” *Security Response*, 2015.

[82] 张耀辉等. 网络安全与管理 [EB/OL]. 2017 [2017.3]. http://ptr.chaoxing.com/nodedetailcontroller/visitnodedetail?knowledgeId=3307688.

[83] Duan Z, Yuan X and Chandrashekar J. “Constructing Inter-Domain Packet Filters to Control IP Spoofing Based on BGP Updates.” *INFOCOM*. 2006.

[84] Abad C L and Bonilla R I. “An analysis on the schemes for detecting and preventing ARP cache poisoning attacks.” *Distributed Computing Systems Workshops, 2007. ICDCSW’07. 27th International Conference on*. 2007, pp. 60 – 69.

[85] Yamamoto K, Yamaguchi M, Miyamaru F, et al. “Noninvasive inspection of C-4 explosive in mails by terahertz time-domain spectroscopy.” *Japanese journal of applied physics*, 2004, 43(3B): L414.

[86] Irani D, Balduzzi M, Balzarotti D, et al. “Reverse social engineering attacks in online social networks.” *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. 2011, pp. 55 –74.

[87] OWASP. Internet of Things Top Ten[EB/OL].2017[2017.4] https://www.owasp.org/images/7/71/Internet_of_Things_Top_Ten_2014-OWASP.pdf

[88] Hamad H, Saad M, and Abed R. “Performance Evaluation of RESTful Web Services for Mobile Devices.” *Int. Arab J. e-Technol*. 2010, 1(3), pp. 72 – 78.

[89] Hunkeler U, Truong H L and Stanford-Clark A. “MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks.” *Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on*. IEEE, 2008, pp. 791 – 798.

[90] Iivari A, Väisänen T, Ben Alaya M, et al. “Harnessing XMPP for Machine-to-Machine Communications & Pervasive Applications.” *Journal of Communications Software & Systems*, 2014, 10(3).

[91] Zanella A, Bui N, Castellani A, et al. “Internet of things for smart cities”. *IEEE Internet of Things journal*, 2014, 1(1), pp. 22 – 32.

[92] Ukil A, Sen J and Koilakonda S. “Embedded security for Internet of Things.” *Emerging Trends and Applications in Computer Science (NCETACS), 2011 2nd National Conference on*. IEEE, 2011, pp. 1 –6.

[93] ANKER. MQTT 协议工作笔记 [EB/OL]. 2017 [2017.3]. http://www.cnblogs.com/Anker/p/5353221.html.

[94] 跬步者. 四大物联网协议介绍 [EB/OL]. 2017 [2017.3]. http://www.cnblogs.com/jhj117/p/5521083.html.

[95] FREEBUF. HackPwn: TCL 智能洗衣机破解细节分析[EB/OL]. 2017 [2017.3]. http://www.freebuf.com/news/76071.html.

[96] 马万明. IoT MQTT CoAP XMPP[EB/OL]. 2017 [2017.4]. http://blog.csdn.net/mawming/article/details/51938890.

[97] 深圳欧德蒙科技有限公司. 智能硬件设备安全问题详解 [EB/OL]. 2017 [2017.4].http://www.oudmon.com/article/56.shtml

[98] 雷锋网. 黑客告诉你, 315 晚会上的智能产品是怎样被黑的 [EB/OL].2017 [2017.4]. http://www.leiphone.com/news/201603/ZgctCsNzSxMW5G0R.html.

[99] Abduvaliyev A, Pathan A S K, Zhou J, et al. “On the vital areas of intrusion detection systems in wireless sensor networks.” *IEEE Communications Surveys & Tutorials*, 2013, 15(3), pp. 1223 – 1237.

[100] Pongle P and Chavan G. “Real time intrusion and wormhole attack detection in internet of things.” *International Journal of Computer Applications*. 2015, 121(9).

[101] Wallgren L, Raza S and Voigt T. “Routing Attacks and Countermeasures in the RPL-based Internet of Things.” *International Journal of Distributed Sensor Networks*, 2013.

[102] Alliance C S. Security Guidance for Early Adopters of the Internet of Things [EB/OL]. 2017 [2017.3]. https://downloads. Cloudsecurityalli-ance.org/whitepapers/Security_Guidance_for_Early_Adopters_of_the_Internet_of_Things.pdf.

王雅哲 于 2010 年在中国科学院软件研究所信息安全专业获得博士学位, 现任中国科学院信息工程研究所副研究员, 研究领域为网络与系统安全, 研究兴趣为智能设备安全。Email: wangyazhe@iie.ac.cn

霍冬冬 于 2014 年在中国科学技术大学软件工程专业获得硕士学位。现任中国科学院信息工程研究所第五研究室研究实习员。研究领域为嵌入式系统, 研究兴趣为智能设备安全。Email: huodongdong@iie.ac.cn

张城毅 于 2014 年在华中科技大学物联网工程专业获得学士学位。现在中国科学院信息工程研究所第五研究室攻读硕士学位。研究领域为网络与系统安全, 研究兴趣为智能设备安全。Email: zhangchengyi@iie.ac.cn

李佳琳 于 2017 年在青岛大学信息安全专业获得学士学位。现在中国科学院信息工程研究所第五研究室攻读硕士学位。研究领域为网络与系统安全, 研究兴趣为智能设备安全。Email: lijialin@iie.ac.cn