中图分类号：**TP391.4**

论 文 编 号：**10006SY2039125**

北京航空航天大學

硕 士 学 位 论 文

基于上下文分析的智能家居事件安全性研究

作 者 姓 名　徐骁赫

学 科 专 业　网络空间安全

指 导 教 师　毛剑　　副教授

培 养 院 系　网络空间安全学院

[– This page is a preset empty page –]

Context-based smart home event security analysis

A Dissertation Submitted for the Degree of Master

Candidate : Xu Xiaohe

Supervisor: Prof. Mao Jian

School of Cyber Science and Technology

Beihang University, Beijing, China

[– This page is a preset empty page –]

中图分类号：**TP391.4**

论 文 编 号：**10006SY2039125**

硕 士 学 位 论 文

基于上下文分析的智能家居事件安全性研究

作者姓名　徐骁赫　　申请学位级别　工学硕士

指导教师姓名　毛剑　　职　　称　　副教授

学科专业　网络空间安全　　研究方向　　物联网安全

学习时间自　2020 年　09 月　01 日　　起至　　2023 年　01 月　15 日止

论文提交日期　2018 年　01 月　10 日　论文答辩日期　2018 年　03 月　01 日

学位授予单位　北京航空航天大学　　学位授予日期　　年　　月　　日

[– This page is a preset empty page –]

关于学位论文的独创性声明

本人郑重声明：所呈交的论文是本人在指导教师指导下独立进行研究工作所取得的成果，论文中有关资料和数据是实事求是的。尽我所知，除文中已经加以标注和致谢外，本论文不包含其他人已经发表或撰写的研究成果，也不包含本人或他人为获得北京航空航天大学或其它教育机构的学位或学历证书而使用过的材料。与我一同工作的同志对研究所做的任何贡献均已在论文中作出了明确的说明。

若有不实之处，本人愿意承担相关法律责任。

学位论文作者签名：_____ 日期： 年 月 日

学位论文使用授权

本人完全同意北京航空航天大学有权使用本学位论文（包括但不限于其印刷版和电子版），使用方式包括但不限于：保留学位论文，按规定向国家有关部门（机构）送交学位论文，以学术交流为目的赠送和交换学位论文，允许学位论文被查阅、借阅和复印，将学位论文的全部或部分内容编入有关数据库进行检索，采用影印、缩印或其他复制手段保存学位论文。

保密学位论文在解密后的使用授权同上。

学位论文作者签名：_____ 日期： 年 月 日

指导教师签名：_____ 日期： 年 月 日

[– This page is a preset empty page –]

摘　要

摘要是学位论文内容的简短陈述，应体现论文工作的核心思想。论文摘要应力求语言精炼准确。博士学位论文的中文摘要一般约800～1200字；硕士学位论文的中文摘要一般约500字。摘要内容应涉及本项科研工作的目的和意义、研究思想和方法、研究成果和结论。博士学位论文必须突出论文的创造性成果，硕士学位论文必须突出论文的新见解。

关键字是为用户查找文献，从文中选取出来揭示全文主体内容的一组词语或术语，应尽量采用词表中的规范词（参考相应的技术术语标准）。关键词一般3～5个，按词条的外延层次排列（外延大的排在前面）。关键词之间用逗号分开，最后一个关键词后不打标点符号。

为了国际交流的需要，论文必须有英文摘要。英文摘要的内容及关键词应与中文摘要及关键词一致，要符合英语语法，语句通顺，文字流畅。英文和汉语拼音一律为Times New Roman体，字号与中文摘要相同。

关键字： 北航，学位论文，博士，硕士，中文

Abstract

What were you doing 500 years ago? Oh, that's right nothing, because you didn't exist yet. In fact, several generations of your family had yet to leave their mark on the world, but one very special shark may already have been swimming in the chilly North Atlantic at that time, and the incredible animal is somehow still alive today.

Scientists studying Greenland sharks observed the particularly old specimen just recently, and after studying it they've determined that the creature is approximately 272 to 512 years old. That's an absolutely insane figure, and if its age lands towards the higher end, it makes the animal the oldest observed living vertebrate on the entire planet.

Greenland sharks are an incredible species in a number of ways, but most notable is its longevity. The sharks are well over 100 years old before even reaching sexual maturity, and regularly live for centuries. This particularly old specimen, along with 27 others, were analyzed using radiocarbon dating. The reading came back at around 392 years, but potential margin of error means the animal's true age is somewhere between 272 and 512.

The shark, which is a female, measures an impressive 18 feet long. That's pretty large, but it might not sound particularly large for an ocean-dwelling creature that lives hundreds of years. That is, until you consider that the Greenland shark only grows around one centimeter per year. With that in mind, 18 feet is actually downright massive.

As for how this particular shark species manages to live so incredibly long, scientists attribute a lot of its longevity to its sluggish metabolism, as well as its environment. The frigid waters where the sharks thrive is thought to increase overall lifespan in a variety of ways. Past research has shown that cold environments can help slow aging, and these centuries-old sharks are most certainly benefiting from their chilly surroundings.

— Online news *Scientists find incredible shark that may be over 500 years old and still kicking*, 12.16.2017. (http://bgr.com/2017/12/14/oldest-shark-greenland-512-years-old/).

Key words: News, BGR, Shark

目 录

图 清 单

[– This page is a preset empty page –]

表 清 单

[– This page is a preset empty page –]

主要符号表

E 能量

m 质量

c 光速

P 概率

T 时间

v 速度

[– This page is a preset empty page –]

第一章　绪论

1.1　课题来源

国家自然科学基金面上项目：基于多源事件复合推演的物联网安全溯源与异常检测机理研究。

1.2　研究背景

智能家居（Smart Home）（又称为自动化家居、智慧家居）这一概念在1984年由American Association of House Builders提出[1]，是一种物联网（Internet of Things，IoT）的典型应用场景，由一系列控制家居环境的智能设备组成。起初智能家居被用于照明和供暖、制冷等系统的自动控制，而随着技术发展，目前智能家居基本涉及到了用户房屋内几乎所有种类的部件（包括门窗、电灯、智能音箱、各种开关、各种传感器等等）。此外，智能家居可以实现对房屋环境的实时监控，以及对智能设备的远程控制，用户以直接独立操作设备或设定自动化规则的形式参与其中[2]。通常情况下，智能家居提供的功能包括舒适度、安全性、可靠性、远程控制能力和能源节约能力[3]。

从结构上来讲，智能家居可用以下三层结构来描述[4]：感知层、网络层、应用层。感知层包括智能设备和传感器，用于感知物理环境的状态；网络层包括网关、移动设备和服务器等硬件设施，通过构建家庭网络来为设备间通信提供可能；应用层包括在移动设备或服务器上运行的App等形式的智能逻辑，负责提供用户UI接口，或通过自动化规则进行智能决策。作为智能家居系统运转的基础，感知能力保证了家居环境状态能够被各类传感器和设备实时收集，从而进一步为用户和应用层逻辑（通常以App的形式）提供准确信息以供决策。现有大多数成熟的智能家居平台（如SmartThings[5]与Home Assistant[6]）为事件驱动的（Event-driven），即通过事件实现信息传递，其中事件（Event）是由设备发出的一种网络消息，用于描述相关的设备状态或环境信息[2]。

智能家居系统通过事件总线（Event Bus）[6]来收集汇总事件，通过发布（Publish）-订阅（Subscribe）机制来控制事件的流向，设备通过发布操作向事件总线添加新事件，应用层逻辑通过订阅操作在事件总线中登记，当有相应事件的发布时，事件总线将这

一消息转发给所有订阅过此事件的应用层逻辑[2]。图1为一个简单的智能家居事件示意图，当智能设备“灯”被用户或被系统远程打开时，其会向智能家居云发布一个“开灯”事件，云中运行的App通过订阅此事件来感知智能电灯的状态，从而进一步做出智能决策。总体来说，事件是现实生活中发生事件的抽象表示，并将物理空间中的智能设备实体映射到网络空间中，使设备能够参与智能决策的过程、提供信息或接受指令[7]，是智能家居系统的核心要素。

事件是智能家居实现感知能力的基础，然而事件存在诸多安全隐患，这些安全隐患可能导致系统出现逻辑上的异常，从而出现意外行为；或是被攻击者利用，使敏感设备受控。文献[8]整理总结了智能家居事件可能出现的安全问题，其中包括事件丢失、事件截断、事件错误、虚假事件等。文献[9]以SmartThings为例指出，事件消息是缺乏保护的，一旦攻击者得到设备ID等敏感信息便可轻易地伪造事件消息，此攻击称为事件欺骗攻击（Event Spoofing Attack）。此研究进一步指出，绝大多数设备均会向系统发布事件，这些事件均会面临被伪造的风险。事件欺骗攻击的实施手段多种多样，包括节点妥协[10]、恶意App植入[11]、中间人攻击[12]、流量包伪造[13]以及超声波语音助手攻击[14–16]等攻击方式，故对攻击者而言非常容易实施。

上述伪造的虚假事件会造成系统对设备状态的错误感知，导致系统执行错误的决策，而通过伪造事件来有目的性地触发自动化规则，攻击者可以在未经授权的前提下控制一些敏感设备。例如，攻击者可以利用“用户到家则解锁门锁”的自动化规则，向系统注入“到家”的虚假事件，从而使门锁打开。以上案例说明，诸如事件欺骗攻击一类的攻击可以使系统产生虚假的事件，从而使系统自动化规则被意外触发，使敏感设备受控。

除此之外，由于智能家居的使用环境以及用户或App设置的关联规则的影响，事件之间存在错综复杂的因果性触发规律，构成了极为复杂的事件依赖关系。其中跨应用程序干扰[17, 18]由不同App对同一设备进行同时操作引起，会造成命令冲突，引起意外的系统行为，使命令未按照用户期望执行；而危险事件依赖[19, 20]可能来源于某些隐式事件依赖，进而引起危险的触发动作，造成敏感设备受控，或家居环境的危险物理状态，如扫地机器人触发动作传感器，进而使系统认为有人在家，进而使门锁打开。上述两种安全隐患的共同之处在于，其均违背了智能家居的正常运行模式，故本文将上述两种安全隐患统称为“异常事件依赖”。

综上所述，智能家居事件的安全问题会造成严重的安全后果，故针对智能家居事件安全的研究是至关重要的。本文拟从上述讨论触发，从虚假事件、异常事件依赖两方面安全问题出发，展开相关虚假事件检测、事件依赖提取、异常事件依赖检测方面的研究，旨在提高智能家居事件的安全性，保证智能家居的正确感知能力与逻辑正确性，从而为整体系统的安全运行提供基础保障。

1.3 研究意义

针对上述讨论，本论文的研究意义主要表现在以下三方面：

1. 为了解决虚假事件问题，有效检测事件欺骗攻击，本论文研究一种基于物理上下文的事件真实性验证方案，对现有方案的准确率进行改进，同时考虑到实际部署情况，通过缩减传感器数量达到轻量化的目的；
2. 为了对事件依赖进行全面提取和展示，本论文研究一种基于复合上下文建模的事件依赖图构建方案，实现了事件依赖的全面、准确提取和直观表示，提升用户对系统的感知能力；
3. 为了检测异常事件依赖，本论文研究一种基于事件依赖特征的异常检测方法，保证了系统中的事件依赖关系在语义上是准确的、安全的。

1.4 本文研究内容以及论文构成

本文基于智能家居的上下文信息，研究事件安全性相关问题，旨在提升智能家居系统的安全性。具体来说，考虑可能出现的虚假事件，设计并实现一种基于物理上下文的事件真实性验证方法，保证智能家居事件的真实性；考虑可能存在的异常事件依赖，设计并实现基于复合上下文建模的事件依赖图构建方法，以及基于事件依赖特征的异常检测方法，实现事件依赖的全面提取与静态异常检测，分析系统可能出现的安全隐患，提高智能家居安全性。

本论文的研究内容如下：

1. 智能家居自动化数据收集实物平台搭建：
2. 基于物理上下文的事件真实性验证：研究通过智能家居事件物理上下文建立事件指纹的方法，并利用指纹实现事件验证，优化现有方案效果；
3. 基于复合上下文建模的事件依赖图构建：研究通过逻辑上下文提取显式事件依赖

的方法，研究通过事件物理上下文的事件物理关系建模方法，利用物理关系提取隐式事件依赖，并实现事件依赖图的构建；

4. 基于事件依赖特征的异常检测：研究事件依赖图结构特征的提取方法，研究事件依赖语义的表达方法以及语义特征提取方法，将事件依赖向量化，研究用于事件依赖异常检测的机器学习模型，最终实现基于事件依赖特征的异常检测。

第二章　智能家居事件安全性现状分析

本章对智能家居事件安全相关研究做一整理总结。其中第一节介绍单一事件本身的安全问题相关研究，第二节介绍事件依赖安全问题相关研究。

2.1　事件安全问题相关研究

2.1.1　事件安全问题总结

文献[16]基于SmartThings用户论坛以及相关安全研究整理总结了智能家居系统可能出现的安全问题。其中包括事件丢失、事件截断、事件错误、虚假事件等。其中事件丢失、事件截断指由于网络或攻击者等原因造成了丢包，从而使控制系统不能实时准确地感知设备，事件错误指由于设备自身故障造成错误事件的产生，虚假事件指恶意攻击者通过某种攻击手段进行了事件注入。

文献[14]提到，SmartThings没有对产生事件的过程做访问控制，也没有对SmartApps提供验证事件完整性的方法，故所有SmartApp或SmartDevice都有权限发出任意事件通知。基于这一发现，恶意app可以向Hub发送错误的事件通知，从而导致错误的自动化规则被触发；或者可以阻止发送正确的事件通知，导致正确的规则没有触发。这一攻击方式被称作事件欺骗。文中还指出，此种攻击方式可以让恶意app越权控制其它设备。如恶意app可以对位置事件进行篡改，在用户在外时发送回家的物理事件，从而触发所有订阅此事件的app做出相应改变，导致门锁打开等规则的触发，达到越权控制的目的。

尽管本文使用SmartThings中的SmartApp说明事件欺骗攻击是如何产生的，且SmartApp已经不是SmartThings自动化规则的主流实现方式，但现实系统中的事件不仅仅来源于App发送的事件通知。文献[17][18][19]给出了一些通过改变物理环境来影响智能设备的感知，进而向智能家居系统中注入虚假事件的攻击手段。在这些攻击中，攻击者通过注入超声波来潜在地改变智能家居系统的物理环境，使得智能音箱感知到一个错误的事件，由于人耳无法感知超声波，故用户完全无法察觉攻击的发生。

文献[13]总结了通过事件欺骗攻击来达成潜在攻击目标的方式。攻击者的攻击目标、攻击设备多样，后果严重。但文中同样提出这些设备可以被附近的传感器感知，从而用作事件验证。如门的状态可以被附近的加速传感器、光线传感器、气压传感器和

麦克风感知，这些传感器就可以用作有关开门、关门事件的验证。

为了识别错误事件或虚假事件，可以为每个事件制定指纹。每个指纹与现实生活中的事件一一对应。当Hub收到app或设备的事件通知时，可以通过验证指纹信息确定事件在现实生活中是否真实发生，从而验证事件的正确性。以下展开介绍事件指纹的相关研究。

2.1.2　事件指纹相关研究

由于本文着重研究有关错误或虚假事件的防御方案，即事件指纹构建方案，故本节列举并介绍了现有事件指纹相关研究。目前关于构建事件指纹的方法主要分为两种：基于网络流量和基于传感器数据。

在基于网络流量构建事件指纹方面，文献[20]提出了一种用加密的网络流量检测家居设备身份和行为的方法，基于此思路攻击者可以获得用户的隐私信息。论文在识别设备身份方面，使用ZigBee网络流量包的统计特征（包括平均包长度、平均到达间隔和包长度的标准差）作为分类依据，选择算法为K近邻分类。在识别设备行为方面，作者基于活动设备的数据包发送率会显著增加这一想法，将数据包序列划分为长度为W的窗口，提取统计特征，作为监督学习的训练数据集，算法分别选用随机森林分类和K近邻分类，两者效果相似。作者还对设备行为的具体分类和用户行为分类进行了研究，其基本思路相近，均为提取流量包的相关特征，以机器学习算法进行分类。基于上述四个步骤，攻击者可以对加密流量进行分析，获得被攻击者家居环境内设备、用户行为的信息，造成隐私泄露。

文献[3]介绍了Homonit系统，此系统同样是基于网络流量的物理事件指纹构建。与上述文献的工作不同的是，此文献的目的是构建监视系统，识别来自app的错误事件通知。作者认为所有智能家居app的行为都遵循DFA（Deterministic Finite Automation，确定有穷自动机）模型，自动机的每个状态代表app和响应家居设备的状态，自动机的状态转换代表app和设备的交互。作者对app行为构建自动机，从无线通信环境中提取事件特征作为指纹。作者指出，此系统是独立于智能家居系统之外的第三方监视系统，在提高系统安全性的同时未对系统做任何修改。但本系统仍然不能在智能家居系统已经被攻击的情况下正确工作，这是由于若设备被控制，它发出的网络信号则不具有正确性。故此系统不能识别来自设备的错误事件通知。

文献[21]提出了PingPong，可以自动从加密网络流量中提取设备事件的数据包级签名。文中指出，流量包长度的唯一序列通常可以描述某些特定简单事件，并可以利用他们来作为唯一事件指纹。首先，PingPong使用Android Debug Bridge（ADB）和shell脚本在智能手机屏幕上模拟触屏输入，来模拟用户发起的设备事件，同时在路由器上使用tcpdump抓包。之后对抓到的数据包进行过滤，只保留源或目的IP为智能家居设备的，同时选择一个时间窗，只保留事件发生后一小范围内的包，然后过滤掉所有TCP或TLS的控制包（如TCP握手挥手和TLS密钥协商）。之后通过构建数据包对，并使用无监督学习算法提取与事件相关的数据包对，并通过相关数据包对构建事件数据包级签名。

在基于传感器数据的指纹方面，文献IoT-CAD提出了IoT-CAD，通过IoT系统中的传感器来捕捉物理环境状态从而识别异常。通过在IoT系统中安装持续性监测的传感器，并定期在时间窗口内产生数据快照，构建快照向量，来作为指纹描述此时刻IoT系统的状态。

文献[22]指出，尽管智能家居内的传感器具有异构性，但它们受相同的物理事件影响都有一定的表现方式。文献[23]提出了基于异构传感器数据来获得事件指纹的方法。作者收集除了摄像头视频数据之外全部传感器的数据，之后提供了两种机器学习的方法：人工标定训练数据集的标签，然后进行两层的SVM分类；缺少人工标定的标签，则进行非监督式机器学习，首先对数据降维，之后使用最大期望算法进行聚类。

文献[13]设计了一个办公室的日常场景，使用树莓派自行构建异构传感器体系，收集了时间跨度为两周的传感器数据，并记录物理世界中发生的真实事件为其标签，作为监督学习的训练数据集。由于收集的时间数据是连续的，作者使用网格搜索选定了针对不同事件的时间窗，提取各个传感器与事件的相对互信息量作为指纹特征，并选择指纹特征大的数据进行SVM分类。这一指纹选取方案可以保证在智能设备被控制时仍有效地检测事件欺骗攻击。

2.1.3 现有研究不足

表 1总结了上述提到的智能家居事件指纹相关研究。总体来说，现有研究的方法大多局限于从特征工程到机器学习的方案，将指纹数据来源直接看做一个大型数据集，从其中提取并筛选特征，然后选用适当的学习算法。

表 1 智能家居事件指纹相关研究总结

指纹来源	文献	网络流量包特征				传感器特征		学习算法
		TS	L	AD	P	TD	FD	
网络流量数据	Peek-a-boo	✓	✓	×	×	-	-	KNN
	Homonit	✓	✓	✓	×	-	-	DFA
	PingPong	✓	×	×	✓	-	-	DBSCAN
传感器数据	IoT-CAD	-	-	-	-	✓	×	RNN
	Synthetic sensors	-	-	-	-	✓	✓	SVM/EM
	Peeves	-	-	-	-	✓	×	SVM

注: TS–时间戳; L–包长度; AD–地址; P–协议; TD–时域; FD–频域

然而，由于智能家居系统中事件的多样性，使用一套通用方法论很难适用于所有事件。例如，Peeves[13]对开门、关门等具有瞬时影响的事件的准确率几乎达到了100%，然而这一效果无法泛化到开加热器等具有持续性影响的事件。

除此之外，某些特征工程的手段可能并不适用于智能家居场景。具体来说，减小特征数量的手段可以分为两种，特征选择（feature selection）和特征抽取（feature extraction），前者是选取原始特征集合的一个有效子集，后者是将原始高维特征空间映射到一个低维空间上。对智能家居场景来说，使用特征选择的好处在于可以减小模型线上验证时需要的数据来源，如使用特征选择的手段对传感器数据特征进行筛选，可以减小用于事件真实性验证的传感器数量，减小部署成本。而特征抽取手段（如Synthetic sensors[23]的自动编码器）没有这一优势。

2.2　事件依赖安全问题研究

2.2.1　事件依赖提取与表示

由于智能家居系统的事件依赖由物理环境和用户设置的自动化规则决定，故不同场景下的事件依赖是截然不同的。为了研究某智能家居系统的事件依赖安全问题，需要先进行此特定系统事件依赖的提取和表示。本节首先着重介绍现有研究中提取事件依赖的方法。

对大部分智能家居平台而言，由于显式事件依赖直接由系统中的触发-响应自动化规则直接定义，而规则通常是由用户设定，或在app中包含，故这些规则很容易通过代码分析、配置文件解析等手段获取[9][14][26][27]。如SmartThings平台的规则通常包含在SmartApps里或.json配置文件中，Home Assistant平台的规则通常包含在一个.yaml文件中。然而，隐式事件依赖在系统中没有类似代码或配置文件这样的显式定义，故需

要结合智能家居物理环境进行考虑[9][14]，或是直接在真实环境中进行动态测试来提取[13]。

在显式事件依赖提取方面，通常使用代码分析的手段，包括静态分析（AST、调用关系图等）或动态分析（代码插桩、建模、符号执行等）。以下介绍现有显式事件依赖提取相关的研究。

文献[24]提出了IoTSan，通过静态分析的方法从IoT应用程序源代码中提取依赖图（dependency graph），依据依赖关系建立状态模型。具体来说，IoTSan首先通过一个App依赖分析模块从SmartApp的groovy源码中寻找事件订阅函数subscribe()、读设备状态的API以及通过schedule()方法定义的定时任务来提取输入事件，并通过调用的API提取输出事件，从而构建完整的事件依赖。然后本研究使用Bandera Tool Set工具对源码进行代码行为建模，由于此工具只接收java代码输入，故本研究开发了一个groovy转java的翻译器，由上述步骤得到一个由Promela语言描述的代码逻辑模型。使用基于应用程序逻辑的建模方法可以更加准确、细粒度地描述智能家居中的事件依赖关系，但由于方法限制，其只能提取到应用程序内定义的规则，而无法衡量物理环境参与其中后的影响，也就是其不能对隐式事件依赖进行提取或识别。

文献[26]提出了Soteria，通过静态分析的手段验证物联网系统中的App或其他行为的安全性和正确性。Soteria使用一种中间表示（Intermediate representation，IR）来描述物联网应用程序的逻辑。具体来说，与之前介绍的静态分析类似，Soteria检测源代码中调用智能设备的函数以及定义触发-响应的函数。然后本研究通过中间表示IR提取应用程序的状态模型，包括智能设备的状态以及状态转换（即事件）。本研究通过模型检查来判断事件的行为是否符合安全属性。与IoTSan[24]类似，由于只从物联网应用程序中提取事件依赖关系，所以本研究无法对隐式事件依赖关系进行识别。

显式事件依赖可直接基于App源代码或配置文件进行提取，较为简单、直观、准确；而隐式事件依赖需要结合智能家居的物理环境进行考虑，这些方法通常需要挖掘不同事件和物理环境的语义并进行综合，通常较为复杂，不够准确。以下介绍隐式事件依赖提取相关研究。

文献[25]提出了IoTMon，来识别和分析IoT App之间的交互链，从而得到事件依赖关系。本研究首先使用静态分析方法从SmartApp的groovy源码中提取应用内交互规则，具体地，首先对App构建抽象语法树（Abstract Syntax Tree，AST），然后寻找声

明自动化规则的代码段，如SmartApp中的installed()和updated()函数，根据其中定义的事件订阅函数subscribe()以及调用的handle函数总结出触发-响应规则。之后，本研究使用NLP对SmartApp的描述做分词和词性标注工作，提取其中的名词，并对这些名词作聚类，选择每个类别中具有代表性的进行人工审查，最终的结果作为物理通道名（如温度、湿度）。之后根据具体事件和物理通道的关系进行不同App之间的关联，从而识别隐式的事件依赖关系。

文献[16]提出了HAWatcher，使用相关性（Correlation）的概念来表达事件与事件之间、事件与物理环境状态之间的语义信息。与上述研究类似，本研究首先SmartApp的代码进行符号执行来获取中间表示IR，并将每个app的语义都以触发-条件-响应的形式来表示。之后，本研究通过NLP对物理环境进行提取，并以假设的方式提出设备属性和物理属性之间的相关性，最后再通过事件日志对上述所有假设的相关性进行检验。

文献[27]提出了IoTGaze，通过无线流量特征和App的描述和UI构建行为基准模型，并检测运行时安全违规行为。本研究跳出物联网平台和应用程序内部的角度，从无线流量的角度重新思考IoT安全和隐私问题，通过收集无线射频环境内的通信数据包，构建基于程序的无限上下文。具体来说，本研究首先提取无线数据包的特征来将通信流量与事件相关联，然后通过发现事件时间性依赖来为物联网系统构无线上下文，最后通过用户界面（UI）提取实际用户预期的上下文，并与无线上下文进行对比，来发现物联网的行为异常。由于通过无线上下文来研究事件和事件间依赖，IoTGaze同样可以检测和识别到隐式事件依赖。

除常规的基于语义和机器学习的方法外，文献[15]提出了IoTSafe，通过设计一个动态测试方法来对物联网系统中的显式和隐式交互行为进行准确挖掘。与IoTMon类似，本研究首先通过代码分析提取App内交互规则，并通过NLP进行分词、聚类、人工审查，提取物理通道信息。然后设计动态测试样例的生成方法，目的是遍历地测试所有设备的所有运行状态对物理环境的影响，同时对于单房间不同设备的并行和多房间并行测试做出优化，减少动态测试时间。通过动态测试来补充设备与环境、环境与设备之间的交互关系，从而完善交互规则。IoTSafe是目前第一个使用动态测试的方法对事件依赖关系进行提取的研究，其优点是能够直接准确地提取传统静态分析方法难以提取到的隐式事件依赖，缺点是由于直接使用现实环境进行测试，若无敏感设备或状态的限制，可能会对用户的人身或物理财产造成影响。

除此之外，文献[28]提出了Hєlion，通过识别用户驱动的家居自动化规则序列的规律性，来通过语义的方式表达用户活动规律。基于人类使用的语言通常是可预测的这一假设，本研究使用统计语言模型，来通过已发生的事件预测未来发生事件的可能性。本研究首先通过用户调查的方式收集以某些特定结构表达的用户使用规律，之后通过解析上述自然语言得到事件令牌，然后使用一个n-gram方法来学习用户在智能家居自动化规则中的规律性。

在事件依赖的可视化表达方面，IoTMon[14]给出了自己的表达方案，使用有向图（文中称为交互链图）来表达App内部和App之间的依赖关系。其中，图的节点有许多类型，包括触发（形式可以是事件、传感器）、响应（形式为事件）、物理通道等，尽管以人的角度实现了一条链的可读性，但结点的类型过多也复杂化了此有向图。此外，HAWatcher[16]将智能家居系统的相关性分为两种：事件到事件，以及事件到状态。具体来说，事件到事件的相关性表示为$E \rightarrow E$，即某一事件触发另一事件的发生；事件到状态的相关性表示为$E \rightarrow S$，即某一事件的发生导致了某传感器的数值以某规律变化。从具体含义上来看，HAWatcher对相关性的定义与本研究对事件依赖的定义类似。

本论文拟在使用一种更统一、准确的表达方式，通过统一有向图节点的类型，并在图的边中加入信息予以辅助说明，来实现有向图的简化，完善事件依赖表达方案，并与后续的异常事件依赖检测进行对接。

2.2.2 异常事件依赖检测

异常事件依赖指超出代码原本逻辑或超出用户期望的事件依赖，其可能会破坏智能家居所在的物理环境，造成安全隐患；或被攻击者利用，通过事件依赖链发起跨事件的攻击。异常既可以由用户自身造成（如误配置或规则冲突等），也可以由潜在的攻击者造成（如恶意代码等）。为了检测这种超出用户期望的异常事件依赖，现有研究对智能家居系统中的正常行为进行建模，利用机器学习方法进行分类或聚类进行异常范围的划分[14][30]；或进行程序级别行为的精确建模，然后动态地监控系统或代码的行为，对基线之外的事件依赖判定为异常[9][26][28][31]。

文献[29]提出6thSense，一种基于上下文感知的入侵检测系统，用于全面检测智能设备中基于传感器的攻击手段。本研究通过实时观察传感器数据，并根据智能设备确定当前的传感器是否是恶意的，来确定设备的上下文；同时，本研究通过观察用户不

同任务的传感器数据变化，将传感器数据与用户活动相关联；然后该方法使用了三种基于机器学习的检测机制（马尔科夫链、朴素贝叶斯和LMT）进行基于传感器的恶意行为的检测。本研究主要侧重于通过传感器来对正常行为进行建模，来上层地、通用地检测一些攻击手段，而非针对事件依赖异常的检测。

IoTMon[25]通过之前提取的应用程序交互关系来检测可能会对物理环境造成危险影响的交互关系，首先对所有的交互进行向量化，然后进行行为建模和聚类，将离群的交互行为认为是有危险性的。具体来说，IoTMon首先分析了不同物理通道的出现频率；然后从出现频率最高的物理通道开始，循环地为与当前物理通道的相关性最高的物理通道赋递增、等间隔的值，其中相关性由共同出现的频率计算；然后对每个事件依赖构建向量，每一维代表其对某物理通道的影响。从结果上来看，此方法有着77%的精准率，其效果不是很理想的原因主要在于构建的事件依赖向量过于简单，每一维的值是固定的。且本研究只对单一事件依赖进行异常的检测，而并没有考虑多层事件依赖可能带来的危险。

除上述静态异常检测外，一些研究也提出了动态运行时的异常检测方法。IoTSan[24]对IoT中可能出现的安全问题进行了检测，包括物理状态异常和敏感信息的泄露。本研究使用一个输出分析器，使用基于启发式的算法，来验证智能应用程序的安装阶段和运行阶段。在安装阶段，输出分析器枚举所有可能的配置并对实际配置进行对比，来判断应用程序是否是恶意的；在之后的阶段，新应用程序会与已安装的应用程序进行分析，观察是否有异常情况，从而界定是否安装此App。此方法能够提取由不同App带来的规则冲突或恶意规则问题，从而对用户误配置或攻击者的恶意代码注入进行一定程度的防御。

HAWatcher[16]通过提取出的智能家居相关性，设计了一个异常检测模块，此模块订阅了家居系统的所有事件，并与家居自动化系统并行运行。利用之前提取的事件、传感器的相关性作为行为模型，收集实时的智能家居设备事件，判断其前序和后序事件进行上下文检查，从而检测偏离行为模型的异常事件关联。文献[30]提出了IoTGuard，一个基于策略的物联网动态安全防护系统。本研究通过模拟应用程序的生命周期，修改IoT应用程序的源代码，收集应用程序运行时信息并存储在动态模型中，然后直接对应用程序运行环境中的不安全和不期望的状态进行阻止，通过监视IoT和“触发-响应”平台应用程序的行为，保护用户免受不安全的设备状态的影响。

文献[31]提出了HomeGuard系统，用于检测跨应用程序干扰（Cross-App Interference，CAI）问题。在安装每个新的App时，此系统都会介入，从已安装的App出发进行规则干扰的检测。具体来说，本研究通过设计实现了一个符号执行器，从SmartApps中提取规则语义信息，利用可满足型模型理论检测冲突。除此之外，本研究还将App的规则语义与用户配置的规则结合以进一步检测威胁。

文献[32]首先全面分析了出发物联网平台规则间漏洞的空间，将现有研究提出的漏洞进行分类和系统化，然后本研究提出了iRuler，使用形式化方法来检测这些漏洞。具体来说，通过可满足性模型理论（Satisfiability Modulo Theories，SMT）来进行模型检查，从而发现规则间漏洞。最后，本研究同样通过NLP来对App描述来进行处理，从而推断规则间信息。

文献[33]检测了IFTTT平台小程序（applet）可能出现的一些问题，其中IFTTT也是一个以触发-响应为自动化规则模板的智能家居平台。本研究通过两种基于URL的攻击展示了IFTTT小程序的隐私泄露问题，并对在野的IFTTT小程序进行了分析，根据其来源的敏感程度进行了分类，发现了30%的applet可能侵犯了隐私。除此之外，本文还提出了一种信息流控制的框架，中断从私密source到敏感sink的访问控制机制，同时监测小程序输出的信息流，来加强小程序的隐私保护。

对上述动态异常检测的研究而言，由于其模型刻画准确的系统行为，故其准确性强，可解释性强，但只能实时做出异常检测，设计一个通用的防御策略比较难以实现。

2.2.3 现有研究不足

表 2 智能家居事件依赖安全相关研究总结

文献	主要方法	事件依赖提取		异常事件依赖检测	
		显式	隐式	静态	动态
IoTSan[26]	代码分析、模型检查	✓	×	-	✓
Soteria[27]	代码分析、模型检查	✓	×	×	×
IoTMon[14]	代码分析、NLP、聚类	✓	✓	✓	-
HAWatcher[9]	代码分析、NLP、假设检验	✓	✓	-	✓
IoTGaze[28]	NLP、事件转移图	✓	✓	×	×
IoTSafe[13]	动态测试	✓	✓	×	×
6thSense[30]	马尔可夫链、朴素贝叶斯、LMT	×	×	✓	-
IoTGuard[31]	代码插桩、代码运行监控	✓	×	-	✓
HomeGuard[12]	代码分析、SMT	✓	×	-	✓
[32]	形式化方法、SMT	×	×	-	✓

表 2总结了现有事件依赖提取和异常依赖检测相关研究。对静态异常事件依赖检测方法而言，其优点是能够全面地分析系统中的所有事件依赖，并且在系统启动或正式

运行之前发现潜在的威胁，但缺点是由于引入通用的机器学习模型来刻画不同种类的事件依赖，其准确性严重依赖于特征工程和模型选择，而现有研究在特征工程方面没有很细粒度的设计，这是由于事件依赖本身的语义比较复杂，故难以设计特征的计算方法；而对动态异常分析而言，由于其模型刻画准确的系统行为，故其准确性强，可解释性强，但只能实时做出异常检测，设计一个通用的防御策略比较难以实现。

此外，现有研究大多数集中于两事件之间依赖的异常检测，如IoTMon[14]，没有考虑多跳的事件依赖链的异常，例如用户配置了$E_{away\ home} \rightsquigarrow E_{light\ off}$以及$E_{light\ off} \rightsquigarrow E_{window\ open}$两条事件依赖，其分别都是符合用户预期的，但若没有多余条件限制，$E_{away\ home} \rightsquigarrow E_{light\ off} \rightsquigarrow E_{window\ open}$的事件依赖链成立，则会导致用户离开家后窗户被打开，造成可能的危险情况。

第三章　小型拟真智能家居实物实验平台

3.1　小型拟真智能家居实物实验平台需求调研

3.1.1　现有公开数据集调研

近年来，智能家居越来越受到研究者的重视，学术界逐渐有一些研究者自行搭建智能家居实验环境，收集用于自己学术研究的数据集并将其公开，供后续研究使用。然而，这些学术研究的侧重点不同，导致其数据集通常不能覆盖所有智能家居的使用场景。本节调研了现有学术界面向智能家居场景的公开数据集，分析其数据特性及适用场景，为本文后续研究中的数据集选择，以及实验平台搭建需求分析做支撑。由于本文研究主要关注事件安全性问题，所用数据主要涵盖各智能设备状态流、控制中枢的事件日志、以及自动化规则等，故本节着重从上述方面进行调研。

英国牛津大学Birnbach等人在其研究[7]中收集了一组数据集并将其公开[21]，他们在一个真实办公室环境内，使用路由器、智能手机、笔记本、树莓派以及各种高精度传感器电路模组构成了覆盖12种感知能力的传感器组合，并以不同的数量、不同的位置布置在办公室的各个位置。这些传感器不仅包括温度、湿度、光照强度、声音强度等对物理通道状态的感知，还包括接触传感器、开关等对设备状态的感知。在实验持续的14天内，办公室内的用户均手动地使用屋内各种设备，各个传感器不间断收集数据，并存储在本地U盘中。实验结束后研究者对数据进行了打包并公开。数据分传感器以csv文件的形式储存，由二元组(t, v)的形式组成，其中t为时间戳，v为传感器读数。此数据集的优势在于其完全利用自制传感器，传感器汇报率、精度较高，便于进行细粒度的数据分析以及复杂机器学习模型的训练，并且其采样持续时间长，传感器类型丰富；然而，其并没有引入智能家居平台作为中枢，即其不能体现智能家居的触发-响应特性，故不适用于事件依赖的相关研究。

类似地，Chimamiwa等人[22]公开了其智能家居传感器序列数据集，该数据集从多个环境传感器收集而来，主要目的为捕获人类的日常生活活动。传感器类型包括温湿度、开关、光传感器等，收集持续时间为6个月，采样率为1Hz。同时，此数据集包括一个公寓内多个房间采集的数据。数据格式方面，传感器以数据库的形式分别存储在五个表中，并以csv文件的形式公开。传感器数据按整型数据和浮点型数据分别存储在两

个表中，剩余三个表则为传感器名称等基本信息。每一数据条目由四元组(id, t, v, s)的形式组成，其中t为时间戳，v为传感器读数，s为传感器名称。从使用场景来看，此数据集相比上述牛津大学数据集更加丰富，且时间跨度更大。但此数据集每个房间内的传感器密度、种类较小，且同样没有考虑智能家居的触发-响应特性。

此外，华盛顿州立大学自适应系统高级研究中心（Center for Advanced Studies in Adaptive Systems，CASAS）公开了其智能家居环境内采集的传感器数据集[23]，通过使用数据分析及人工智能技术来对家居环境进行解释、建模和预测，主要用于用户活动感知、自动健康诊断、节能家居自动化等领域，目标为改善用户的生活。此数据集主要面向用户活动引起的智能家居设备状态改变活动，采集的数据为总日志条目的形式，每一条目以三元组(t, d, s)组成，其中中t为时间戳，d为设备名称，s为设备状态。此数据集相比上述牛津大学数据集，用户活动以及场景更加丰富，但传感器精度、采样率、种类略有不足，故其更适合用于更上层的建模分析工作，如用户活动感知等。同样地，此数据集亦没有引入智能家居平台作为中枢，没有体现触发-响应特性。

除此之外，一些民间技术爱好者将数据集分享至知名数据分析竞赛平台Kaggle[24]上，这些数据集主要关注智能家居场景中某单一方面的数据，如各设备用电功率数据、智能家居控制指令数据等。总体来说，这些数据集涵盖事件类型、物理通道种类较少，并且缺少数据采集时的实验细节，不适合用于学术研究。

3.1.2　实验平台需求分析

上述介绍的现有智能家居公开数据集，绝大多数局限于传感器读数的记录，即在一个设定好的智能家居场景内安装传感器，并按不同的时间间隔连续采集数据。在事件指纹的研究方面，Peeves[7]进行的是类似的工作，故本文在事件指纹方面的研究可用此数据集进行；而由于本文还需要对事件依赖安全进行研究，在显式事件依赖方面，需要考虑智能家居场景中设定的触发-响应形式的自动化规则吗，而现有数据集均未考虑这一设定；而在隐式事件依赖方面，需要细粒度、定量地建模不同事件对物理通道的影响，故需要触发单一事件并连续收集物理通道的影响，而现有数据集中多个事件可能同时或以很小的时间间隔触发，造成其物理通道影响杂糅交错，会造成后续数据分析工作的误差。因此，现有公开数据集均不适合作为本文事件依赖相关研究的支撑。

此外，除上述公开数据集的研究之外，现有智能家居安全相关研究（尤其是事件指纹、事件依赖等相关研究），绝大多数都未采用现成公开的数据集，而是自行搭建智能家居环境并设计应用场景来进行数据采集，导致这些数据集的泛用性较低，使用场景局限。同时，这些研究绝大多数均未公开自己的数据，这使得后续研究仍需重新设计并采集新的数据集，导致工作量加大，同时也无法进行有效、公平的横向对比。

综上所述，为了使智能家居场景灵活化，数据收集的过程方便化，本文设计并实现了小型拟真智能家居实物实验平台。本实验平台的目标为支撑本文后续研究中需要的数据收集工作，并作为后续多样的相关研究工作提供灵活的数据来源。具体来说，本实验平台的设计需求如下：

（1）拟真性：首先，不同于现有数据集并未引入真实的智能家居平台，本实验平台需要尽量地贴合真实的智能家居使用场景。具体来说分为三个方面：在控制中枢方面，本实验平台需要引入一个真实的智能家居中枢系统，并设计合理的自动化规则来模拟用户的使用场景；同时，在设备方面，本实验平台需要设置贴合真实场景的传感器与智能设备组合；此外，本实验平台还需模拟真实智能家居环境的其他特性，如家居物理环境、数据传输方式、协议类型等。

（2）灵活性：其次，本实验平台需要具有灵活多变的场景模拟能力。具体来说，本实验平台需要能够灵活改变各类用户属性及配置，如自动化规则等，用于满足用户的各种使用场景，以满足后续不同研究工作的数据需求。同时，本实验平台需要具有一定的设备扩展能力，即在后续出现新的需求时，可在不影响现有系统的前提下添加新的设备，来扩展平台的使用场景。

（3）易用性：最后，本实验平台需要具有丰富的上层功能设计，以使数据采集的过程更加方便、直观。具体来说，本实验平台需要为数据采集过程设计并实现丰富的前台功能，研究者只需输入与实验设置有关的一些基本信息（如设定哪些规则、触发哪些事件、收集哪些时间段的数据等），实验平台即可自动化完成数据采集与打包工作，无需用户向智能家居平台或数据库等后端应用进行对接。同时，本实验平台需要设计直观、易操作的前端用户UI界面，实时展示状态的同时也能方便用户控制。

3.2 实验平台设计

基于上述实验平台需求，本节从硬件架构、软件架构两方面分别介绍本实验平台

的设计。本平台的整体架构如图1所示。

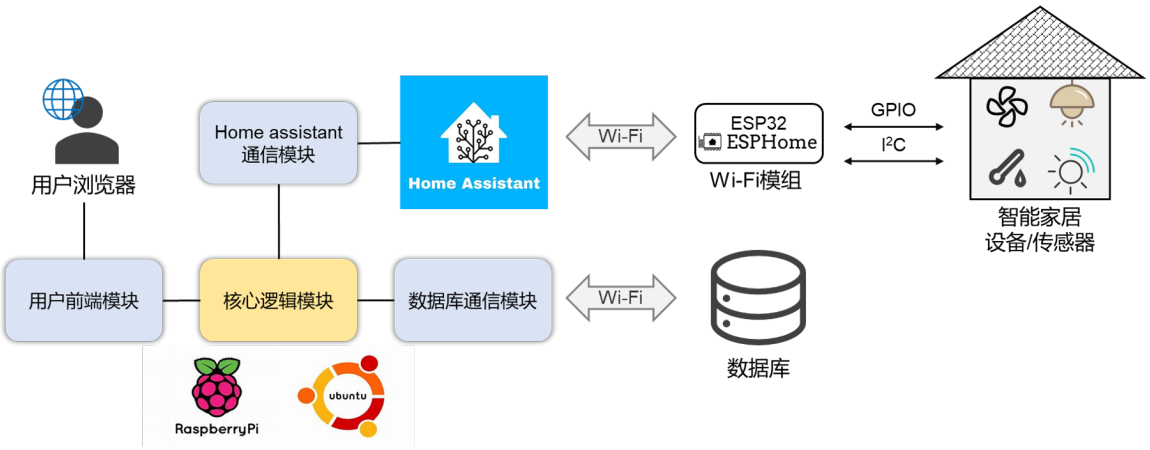

图 1 小型拟真智能家居实物实验平台架构图

3.2.1　硬件架构设计

为满足设计需求中的拟真性，本实验平台引入了真实的智能家居平台作为中枢环境，用于处理与智能家居相关的逻辑。同时，为满足设计需求中的灵活性，即设备的多样性和可扩展性，实验平台选取了DIY平台Home assistant[6]。该平台为开源平台，通过制定一套原生API逻辑与各种不同的智能家居设备通信。目前已有许多商用智能家居品牌的设备支持通过此API连接到Home assistant，如小米、三星等。

Home assistant提供了三种部署形式：Python模组、容器以及操作系统。考虑到本实验平台除Home assistant外还需同步运行一些软件逻辑，平台使用一台树莓派4B作为家居中枢，安装Linux系统并以Python模组的方式部署智能家居平台。由于树莓派具有小型化、轻量化的优点，其较为适合本文的小型拟真智能家居平台，且树莓派4B的计算性能对于Home assistant平台已经足够。为部署Home assistant平台，还需配置sql数据库作为智能家居日志的存储器。考虑到树莓派的存储性能有限，本平台配置了一台远程mysql数据库服务器主机，使用其远程存储智能家居日志，便于后续数据收集的过程。

除智能家居中枢外，本平台还包括一系列小型智能家居设备与传感器。为满足小型化、灵活性的需求，平台采用了DIY设备与传感器模块的形式，这是由于这些模块足够轻量级。设备通信方面，需要将这些设备与传感器模块接入Wi-Fi并与树莓派通信，故平台采用了ESP32 Wi-Fi模组的方式。此模块是一个拥有数个GPIO接口的可编程Wi-Fi开发模块。进一步地，为使设备与传感器能通过Home assistant原生API与其联通，本平台在ESP32上烧录开源ESPHome[25]软件，该软件可以将GPIO接口采集到的数

据转化为API支持的形式，自动通过Wi-Fi与Home assistant联通。

传感器方面，相比于传统商业化智能家居传感器，本平台采用的传感器模组均精度较高，且可达到较高的采样率。通常来说，这些高性能传感器使用I^2总线的形式进行板间通信。I^2C是一种使用一根时钟线和一根数据线即可实现的数字通信总线协议，且可支持多传感器并联。ESPHome开源程序原生在原有GPIO基础上实现I^2C，只需在配置文件中定义^{2}引脚号即可。综上所述，使用ESP32并烧录ESPHome的方式能极为方便地使各种类型的设备和传感器模组与Home assistant联通，可以保证设备的多样性和扩展性，故进一步支持了设计需求中的灵活性。

3.2.2 软件架构设计

在上述硬件的基础上，本实验平台还需软件支撑，用于满足设计需求中的易用性。具体来说，为满足实验平台数据收集的功能，支撑后续研究工作，本实验平台的软件部分计划实现下述基本功能：

1. 自定义事件发起时间和种类：为模拟智能家居的各种使用场景，需要自定义地发起事件或事件序列，并能够预先设定事件发起规则；
2. 自定义数据收集时间段：为满足数据收集需求，需要自定义数据收集的时间段，同步完成预处理，输出能够直接进行数据分析的文件格式（如csv等）。

此外，考虑到软件的可移植与便携性，本平台选择Web应用程序的形式，将后端运行在树莓派上，使树莓派同时运行Home assistant与Web应用程序，如此以来即可将客户端的要求降到最低。

本平台的软件部分主要包含以下四个模块：（1）用户前端模块；（2）Home assistant通信模块；（3）数据库通信模块；（4）核心逻辑模块。

用户前端模块用于为用户提供连接配置接口，其中用户需要输入Home assistant与Mysql数据库的主机、端口等信息，Web应用程序接收到上述连接信息后尝试连接，测试系统的联通性。同时，由于Home assistant具有用户认证功能，正常用户需要输入用户名与口令进入其前端页面，而本Web应用程序通过Home assistant提供的原生RESTful API通信，其认证过程中通过请求头的token字段实现，故此用户前端模块还包括读取用户提供的token文件的功能。

Home assistant通信模块通过原生RESTful API[26]与Home assistant进行交互。通过

此API可以以GET或POST请求的方式完成大部分智能家居核心功能，如读取设备状态、发起事件（服务）来控制设备等。

数据库通信模块用于远程连接Mysql数据库，并使用SQL参数化查询的方式读取数据库中的智能家居日志记录。

核心逻辑模块在上述三个基础模块的基础上搭建，用于实现Web应用程序的上层功能。其中，为实现自定义事件发起时间和种类，Web应用程序首先通过调用API获取Home assistant的所有已知服务和设备，同时接收用户设定，以定时后台任务的方式来控制Home assistant；为实现自定义数据收集时间段的功能，Web应用程序接收用户设定，并将时间等参数传给数据库通信模块。

3.3　实验平台实现

本节介绍本实验平台在实现上的技术细节。具体来说分为两方面，首先介绍实验平台在智能家居设备上的选择、供电方式以及位置安排等，之后介绍实验平台软件方面的代码实现细节和用户接口定义。

3.3.1　设备选择与部署

3.3.2　软件功能实现

3.4　平台使用方法与案例

第四章　基于传感器数据的智能家居轻量级事件认证

4.1　引言

4.1.1　事件指纹介绍

由于智能家居场景中的事件对应现实世界中的物理状态变化，故根据物理规律，事件会导致测量物理通道的传感器按照特定的模式变化。例如，打开或关闭门会导致门上及周边的加速度传感器以一定规律波动，如图2所示。此外，这种对物理通道的影响仅仅体现在事件发生时间戳周围的短时间窗口内发生，即加速度传感器的变化仅局限在“开门”和“关门”事件前后的短短几秒内。

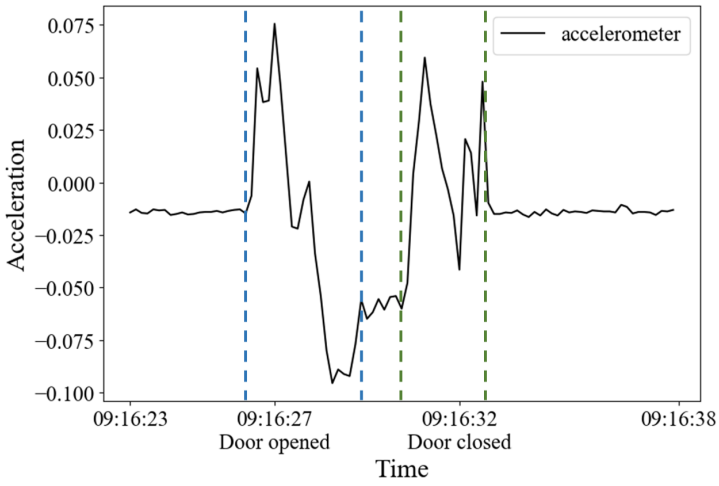

图 2 事件引起的传感器读数波动

针对上述观察到的规律，可以设计一种基于时间窗的传感器特征提取策略，通过在传感器数据序列上加窗，来建立传感器数据与物理事件之间的关联，从而进一步使用特征值构建事件指纹。事件指纹其含义为能够具体刻画某一事件发生的一组特征，用户可以通过识别事件指纹来准确判定事件的发生与否。基于异构传感器的事件指纹信息源自日常生活中的各组传感器数据，通过在传感器数据中提取受事件影响的变化，从而找寻事件与各传感器数据之间的关联。

本文提出的轻量级事件认证方法EGuard以及目前典型的通过传感器构建事件指纹的研究[7, 27]均使用了加窗的思路，基于窗口内数据进行特征构建，并训练机器学习模型进行判决。

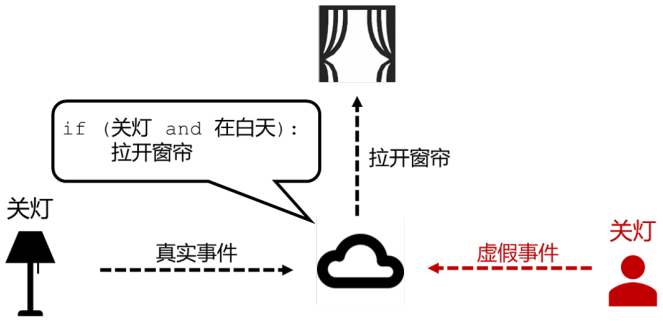

图 3　事件欺骗攻击

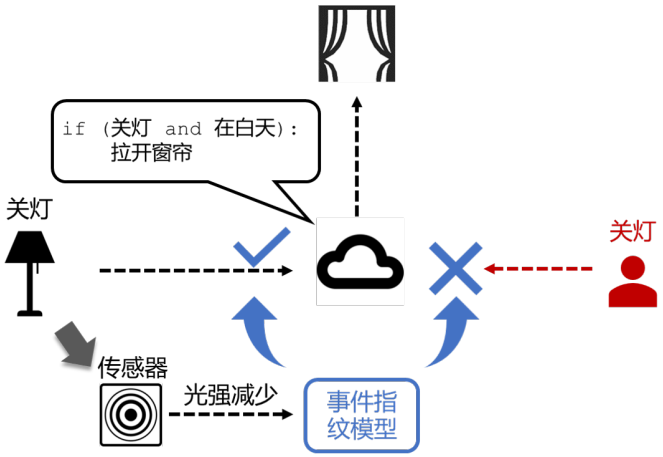

图 4　使用事件指纹检测事件欺骗攻击

4.1.2　目标问题

在智能家居系统中，由用户手动触发或自动化规则触发的物理事件可认为是符合用户意图的，这一类事件是合法的、安全的；而攻击者通过欺骗攻击注入的虚假事件意在破坏智能家居系统的正常运行规律，越权控制家居设备，是非法的、不安全的。需要注意的是，虚假事件无法在物理空间中更改任何设备状态，只能欺骗智能家居系统的控制中枢，或者间接地利用自动化规则使控制中枢下发错误指令。

EGuard构建的事件指纹用于对系统中的虚假事件进行检测，从而达到事件认证的目的。具体来说，EGuard使用智能家居系统中常见的异构传感器系统实时验证事件真实性，在控制中枢接收到事件消息时对其进行认证，有效拦截虚假事件，阻止事件欺骗攻击发生。我们假设EGuard在线运行期间，用于验证的传感器和进行运算的相关服务器是可信的，而智能设备报告的原始事件是不可信的。

下面举一个日常生活中的实例来说明上述问题。如图3所示，攻击者希望针对用户的智能家居系统展开事件欺骗攻击，其实施手段为发送虚构的“关灯”事件，期望触发系统“白天关灯则拉开窗帘”的自动化规则，最终达到间接控制窗帘拉开的目的。正常合法的关灯事件（在拉开窗帘之前）必然会造成室内光强变弱，即光线传感器数

值下降可以作为关灯事件的指纹，但由于事件为虚构，事实上并没有关灯，光线传感器不会检测到光强变弱，故系统若检测到光线传感器数据保持不变，即并没有识别到“关灯”事件的指纹，则可判定此事件为假，从而可进一步阻止此事件通知触发自动化规则，窗帘则不会被拉开，如图4所示。

4.2 EGuard系统设计

表 3 List of the major variables

Variable	Definition
$S_i(t)$	The data sequence of the ith sensor
t_j	The timestamp of the jth event occurrence
l_j	The label of the jth event occurrence
l	The label sequence composed of l_j
$(t^-_{E,S_i}, t^+_{E,S_i})$	Time window for event E and sensor S_i
$\vec{F_{S_i}(t_j)}$	The feature vector of sensor S_i at the timestamp t_j
$f_{S_i,m}(t_j)$	The mth feature value of sensor S_i at the timestamp t_j
$f_{S_i,m}$	The mth feature value sequence of sensor S_i aggregated from $f_{S_i,m}(t_j)$
$\vec{F(t_j)}$	The feature vector including all sensors at the timestamp t_j

本节详细介绍EGuard的设计细节。具体来说，EGuard分为五个模块：1）事件标注模块；2）特征构建模块；3）传感器选择模块；4）事件指纹构建模块；5）机器学习分类模块。

图5为EGuard的整体框架图。其中，事件标注模块根据事件日志记录对事件添加标签，得到包含标签的事件三元组；特征构建模块根据预定指标选取传感器数据序列的最佳时间窗口，并对其进行加窗，得到传感器特征值；传感器选择模块通过训练神经网络对特征的重要性进行衡量，只保留有代表性的、富信息的传感器；事件指纹构建模块根据传感器数据序列和被选传感器集合计算得到传感器特征，获得事件指纹；机器学习分类模块通过上述传感器特征，结合事件三元组中的标签训练得到用于判别事件真伪的事件认证模型，进行事件真伪判别，最终实现对物联网事件真实性的保证以及对虚假事件的检测，提高物联网系统的安全性。

具体来说，EGuard对每种智能家居事件进行分别考虑。对某一待认证事件E，其事件日志中n条记录的且时间戳表示为$t_1, \cdots, t_n$；将智能家居系统中k个传感器表示为$S_1, \cdots S_k$，传感器数据时间序列分别表示为$S_1(t), \cdots, S_k(t)$。此外，表 3中总结了本节中出现的数学符号及其定义。

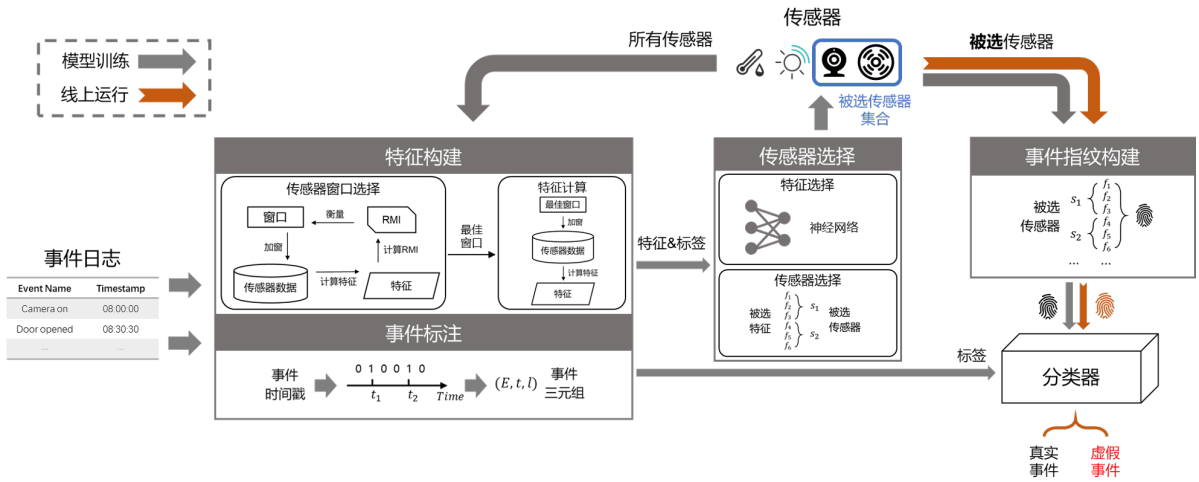

图 5 EGuard整体架构图

4.2.1　事件标注

事件标注模块使用原始事件日志作为输入，输出为三元组形式的带标签事件$(E, t_j, l_j), j \in \{1, 2, \cdots, n\}$，其中$t_j$为事件时间戳，$l \in \{0, 1\}$为时间标签。具体来说，针对某一智能家居事件$E$，在时间轴上标注事件发生的时刻，并将其对应标签打为1（正样本）；同时，在事件发生时刻的间隔内进行采样，作为事件未发生的时刻，并将其对应标签打为0（负样本）。正样本表示此时刻事件真实发生，而负样本表示此时刻事件未发生。

4.2.2　特征构建

特征构建模块使用上述得到的事件三元组以及传感器数据作为输入，输出所有传感器的特征值。为计算特征，首先要对传感器序列按照事件三元组中的时间戳进行加窗，然后在窗口内以某种方式计算特征。以下分别介绍传感器窗口选择以及特征计算的过程。

（1）传感器窗口选择： 正如在 4.1.1中介绍的，事件只会在一个小时间邻域内影响传感器读数，故首先需要对此邻域的范围进行搜索，即最佳传感器窗口的选择。

对于事件E和传感器S_i，定义一个时间窗口为$(t^-_{E,S_i}, t^+_{E,S_i})$，即窗口起点位于事件发生前$t^-_{E,S}$秒，终点位于事件发生后$t^+_{E,S}$秒。首先选取一个窗口的时间范围$-4 \leq t^-_{E,S_i} < t^+_{E,S_i} \leq 3, t^-, t^+ \in N$，N为整数集。之后，对每个候选时间窗口，参照事件三元组中的某个时间戳t_j，在传感器数据序列$S_i(t)$上在时间戳t_j附近加窗，提取窗口范围内的传感器数据序列$S_i(t), t \in (t_j + t^-_{E,S_i}, t_j + t^+_{E,S_i})$。然后，基于此序列进行特征计算（具体方法在下文中详细讲述），得到此序列的m个特征值，记为$f_{S_i,1}(t_j), \cdots, f_{S_i,m}(t_j)$，并拼接得到特征向量$\vec{F_{S_i}}(t_j) = \{f_{S_i,1}(t_j), \cdots, f_{S_i,m}(t_j)\}$，以此类推地对所有事件三元组$(E, t_j, l_j)$重

复如上操作，得到与事件三元组一一对应的特征矩阵：

$$\mathbf{F_{S_i}} = \begin{bmatrix} \vec{F_{s_i}}(t_1) \\ \vec{F_{s_i}}(t_2) \\ \vec{F_{s_i}}(t_3) \\ \vdots \\ \vec{F_{s_i}}(t_n) \end{bmatrix} = \begin{bmatrix} f_{si,1}(t_1) & \cdots & f_{s_i,m}(t_1) \\ f_{si,1}(t_2) & \cdots & f_{s_i,m}(t_2) \\ f_{si,1}(t_3) & \cdots & f_{s_i,m}(t_3) \\ \vdots & \vdots & \vdots \\ f_{si,1}(t_n) & \cdots & f_{s_i,m}(t_n) \end{bmatrix} \tag{4.1}$$

之后，从事件三元组序列$(E,t_1,l_1),\cdots,(E,t_n,l_n)$中提取标签元素，得到标签序列$l=\{l_1,\cdots,l_n\}$；同时从上述特征矩阵中提取每一列，作为维度特征与事件三元组一一对应的特征序列。例如，提取传感器S_i的第m个特征，其序列即为矩阵$\mathbf{F_{S_i}}$的第m列，记为$\mathbf{F_{S_i,m}}=\{f_{S_i,m}(t_1),\cdots,f_{S_i,m}(t_n)\}$。通过上述过程，针对传感器$S_i$的所有m个特征，可以得到其特征序列$f_{S_i,m}(t_1),\cdots,f_{S_i,m}(t_n)$。

然后，对所有特征序列，分别计算其与标签序列的相对互信息量（Relative Mutual Information，RMI），并进一步计算智能家居事件E与传感器S_i之间相对互信息量，并作为传感器窗口选择的评判指标。具体来说，智能家居事件E与传感器S_i的RMI由此传感器下所有m个特征序列的RMI最大值决定，计算过程如下所示：

$$RMI_{E,S_i}=max(RMI_{E,f_{S_i,u}})=max(\frac{I(l;F_{S_i,u})}{H(l)}),u\in\{1,2,\cdots,m\} \tag{4.2}$$

其中，$I(\cdot)$代表两序列间的互信息量，$H(\cdot)$代表信息熵。

最后，在对所有候选时间窗口进行上述计算后，选取RMI最大的候选作为最优时间窗口。

（2）特征计算：

由上一步得到最佳的传感器时间窗口后，再次使用事件三元组和原始传感器数据作为输入，对传感器数据进行加窗，并进一步基于加窗后数据计算特征值。与上一步不同的是，此步骤中只需直接使用最佳时间窗口进行加窗，而无需考虑其他候选窗口。

在计算特征时，考虑到在智能家居环境中传感器本身具有一定的波动性，例如，在没有设备接入的情况下，温度总是会在下午较高，晚上或清晨较低。若这种波动与

事件引起的物理环境变化混杂，则会对特征计算带来误差。故EGuard考虑传感器的变化值，以此消去传感器数值本身波动的影响。具体来说，对某一窗口内的序列而言，其差值由此窗口内每个数据点减去上一窗口的平均值得出，即$S_i(t_j) - \overline{S_i(t_{j-1})}$。

对于具体的特征计算方法，EGuard选用了五种序列的统计特征，包括最小值、最大值、均值、和值和标准差。在传感器数据差值的基础上，这些统计特征基本足以描述传感器在短时间内的变化规律。因此，每个传感器序列可以得到五个特征序列，k个传感器共得到$m * k$个特征序列，其中m为5。这些序列中每个数据点均对应一个事件三元组。之后，将每个事件三元组对应的所有传感器的所有特征进行聚合，得到此事件三元组对应的特征向量，即：

$$F(\vec{t_j}) = \{f_{S_1,1}(t_j), \cdots, f_{S_i,m}(t_j), f_{S_2,1}(t_j), \cdots, f_{S_2,m}(t_j), \cdots, f_{S_k,1}(t_j), \cdots, f_{S_k,m}(t_j)\} \tag{4.3}$$

4.2.3　传感器选择

传感器选择模块接收上一步得到的特征向量作为输入，通过训练一个特定的神经网络进行特征的重要性衡量，并以事件三元组中l_j作为训练标签。此模块的目的在于尽可能地减小用于事件认证的传感器数量，从而使EGuard达到轻量化的目的。此模块的输出为被选的传感器集合。

对于智能家居场景来说，其事件在物理通道状态上的表现是较为复杂和隐性的。由于相比于传统的统计方法或简单机器学习模型，神经网络能够挖掘到数据之间更加隐性的关联，故EGuard选取了神经网络作为衡量特征重要性的模型。进一步地，正如本章引言所说，智能家居中传感器之间也存在者较强的相关性，为了进一步减小传感器数量，需要在多个富信息且高相关的传感器之间仅选择一个或少数。而依据神经网络领域的共识，使用L1正则化可以达到此目的。

具体来说，EGuard采用了包含一层一对一（one-to-one）节点的多层感知机，以此层的权重向量代表输入特征向量每个元素的重要性。一对一层输出为输入和权重向量的哈达玛积，即逐元素相乘。对此层引入L1正则化后，其会对权重向量的值作约束，尽量地使其值绝对值之和最小，以此可以得到一个稀疏的结果（稀疏即向量中大部分

元素为0）。而权重向量的元素为0代表其对应的特征在网络后面的结构中完全被舍去，即可代表此特征完全无用。

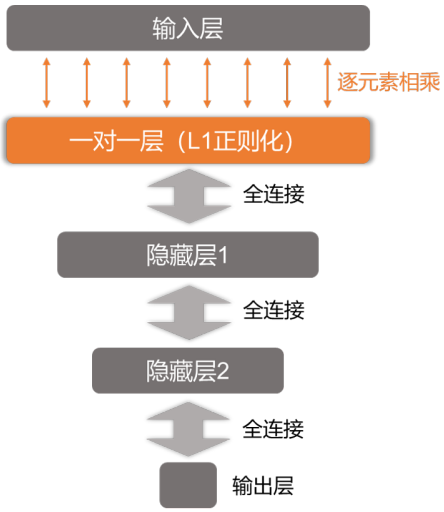

图 6 特征选择神经网络结构

神经网络的具体结构如图6所示。输入层接受输入为$F(\vec{t}_j)$，其节点数为$5*k$即特征数量，5为特征种类，k为传感器数量。通过包含$m*k$个节点的带L1正则化的一对一层，之后通过两层带ReLU激活函数的全连接层，输出层为1个节点，使用Sigmoid激活函数，其目的是将输出压缩至0到1内，从而代表分类分数。之后设置分类阈值为0.5，分类分数高于此阈值则判定为正样本，低于此阈值判定为负样本。

其中，在训练此神经网络时，EGuard使用二值交叉熵（Binary cross entropy，BCE）评判误差，假设输出的预测标签为$\hat{y}_i$，真实标签为y_i，一对一节点中的权重向量表示为$\vec{\omega}$，则神经网络的损失函数可以为：

$$J = \frac{1}{n}\sum_{i=1}^{n} L(\hat{y}_i, y_i) + \lambda\|\vec{\omega}\|_1 \tag{4.4}$$

其中，$\vec{\omega}$为一对一层的权重向量，$L(\hat{y}_i, y_i)$为BCE损失函数，$\|\cdot\|_1$为L1范数，λ为L1正则化系数。

需要注意的是，在参数更新过程中，实际的权重向量很难得到一个稀疏结果，而是得到一系列很接近0的较小值。这是由于特征维度过大，并且程序在计算梯度时存在近似误差导致的。为解决此问题，EGuard使用SGD-L1(clipping)优化算法[28, 29]，该算法是SGD（stochastic gradient descent，随机梯度下降）算法的一种改进。具体来说，此算法在每次迭代时保$\vec{\omega}$中的每个权重不会越过0。整体的参数更新过程如算法 1所示。

通过训练上述神经网络，所有重要程度为0的特征被去除，而其余特征被采用。进一步地，我们选择包含被采用特征的最小传感器集合作为最终选择结果，记为S_v。

算法 1: Neural network parameter update procedure

Input: Model parameters from the last iteration $\vec{\omega}, \vec{\omega_1}, \vec{\omega_2}, \vec{\omega_3}, \vec{b_1}, \vec{b_2}, \vec{b_3}$, loss J, learning rate η_0, η_1

Output: Model parameters after update $\vec{\omega'}, \vec{\omega_1'}, \vec{\omega_2'}, \vec{\omega_3'}, \vec{b_1'}, \vec{b_2'}, \vec{b_3'}$

for *each parameter* $p \in (\vec{\omega}, \vec{\omega_1}, \vec{\omega_2}, \vec{\omega_3}, \vec{b_1}, \vec{b_2}, \vec{b_3})$ **do**
    compute gradient $\frac{\delta J}{\delta p}$ using back-propagation;
end
// parameters excluding one-to-one layer
for *each parameter* $p \in (\vec{\omega_1}, \vec{\omega_2}, \vec{\omega_3}, \vec{b_1}, \vec{b_2}, \vec{b_3})$ **do**
    $p' = p - \eta_1 \frac{\delta J}{\delta p}$;
end
// parameters in one-to-one layer, using SGD-L1(Clipping)
for *each parameter* $p \in \vec{\omega}$ **do**
    $p' = p - \eta_0 \frac{\delta J}{\delta p}$;
    // limit the parameter sign changes
    if *(*$p \cdot p' < 0$*)* **then**
        $p' = 0$;
    end
end

4.2.4　事件指纹构建与机器学习分类

事件指纹构建模块根据传感器数据序列和被选取的传感器集合计算特征，作为事件指纹。特征计算方法与上述方法相同。具体来说，对每一事件三元组(E, t_j, l_j)，根据传感器数据序列，和通过上述步骤获取的用于事件认证的传感器集合，并利用上述步骤中计算特征向量的方法，计算传感器的特征，并将其聚合得到特征向量$\vec{F_{S_v}}(t_j)$作为事件指纹。

机器学习分类模块则使用上述事件指纹作为输入，输出二值的判决结果，其中1代表事件的确发生，即此事件是真实事件；而0代表事件实际没有发生，则此事件是虚假事件。具体来说，在训练时，EGuard使用$\vec{F_{S_v}}(t_j)$作为输入，使用事件三元组中的l_j作为标签。使用SVM分类器的意义在于，SVM在特征维度较大的场景中也具有较高的效率，且能够减小方法在进行线上事件认证时的复杂度。

在训练阶段结束后，为了对事件E进行认证，EGuard只需保留三类参数：最优时间窗口$(t_(E, S_i)^-, t_(E, S_i)^+), i \in \{1, 2, \cdots, k\}$、用于认证的传感器集合$S_v$以及训练好

的SVM分类器模型。

4.3 方案实现

本节介绍本文在EGuard实现上的细节。本文使用了牛津大学Birnbach等人收集和分享的数据集[21]作为数据来源，对EGuard方案进行了全面实现。除此之外，本文基于同一数据集实现了Birnbach本人的研究Peeves[7]，以进行横向对比。本节主要分以下三个方面对EGuard的实现细节进行介绍：预处理、传感器窗口选择、模型训练。环境方面，本文在一个Intel i5处理器，16GB内存的笔记本电脑上完成方案实现，编程语言为Python。

在Peeves的实现方面，其使用的步骤与本文类似，但他们使用了RMI阈值来选择特征。在默认情况下，Peeves使用40%的RMI阈值，即选择RMI高于0.4的特征，以限制特征的噪声并减少时间开销。本文的实现部分与其介绍保持一致。此外，本文调整这个阈值以获得不同的传感器数量，以便做更具体的实验评估，详细内容见 4.4。

4.3.1 预处理

预处理的目的是为了进行数据清洗，并得到三个子数据集用于不同的目的。此数据集由一台笔记本电脑和12台树莓派在办公室里收集，历时13天，其中包括49组传感器数据和22组事件记录。首先在数据清洗方面，本文依据数据集的说明文档删除了由设备故障造成的异常时间段中的数据点；对于说明文档中未提到的异常数据，如长时间内的连续离群点，本文进行了简单的数据观测，并将明显不合理的传感器序列进行排除。

如 4.2.1所述，由于负样本由在正样本时间间隔之间的时间轴上手动添加，故可以通过操纵采样率来控制负样本的样本总量。本文选取合适的间隔，使得到的样本总量约为10000个。之后，本文将所有样本分隔为三个子数据集，其中开发集（development set）占比为1/13，专用于进行时间窗口的选择；剩余的部分中，训练集占60%，用于训练神经网络以及机器学习分类器，测试集占40%，用于进一步评估。

4.3.2 传感器窗口选择

在传感器窗口的选择范围方面，时间窗口简化地表示为区间(t^-, t^+)，则本文选取候选时间窗口范围为$t^- \in [-4, 3], t^+ \in [t^- + 1, 4]$，共36个窗口。

在窗口评价指标方面，本文选取指标为RMI。对于RMI的计算，一般方法为计算事件和传感器特征之间的互信息量$I(X;Y)$，用互信息量除以事件的信息熵$H(E)$。传统的互信息量计算公式如下所示：

$$I(X;Y) = \Sigma_{y\in Y}\Sigma_{x\in X}p(x,y)log(\frac{p(x,y)}{p(x)p(y)}) \tag{4.5}$$

此基于概率的计算方法只有当随机变量X，Y是离散分布时才能取得较好的效果。然而大多数传感器特征的分布是连续的，每个取值均无重复，故无法计算每个取值的概率。传统离散化的方法涉及到分割区间数量的选取等问题，且会造成较大误差，而现有研究Peeves[7]使用了一种基于k-近邻的方法[30]来计算离散-连续的互信息量。然而，根据本文前期测试，此算法在计算离散-离散互信息时会产生很大的误差，故并不适合计算离散性质传感器特征的互信息。因此，对于离散的传感器特征，本文对其进行离散化，并使用上述基于概率的传统方法计算互信息量。此外，一些研究提出了适合于任意分布数据的互信息估计算法[31]，它可以应用于本工作的特征值的离散和连续混合的情况，但这种算法在实验过程中显示出很高的时间开销，故被弃用。

4.3.3　模型训练

传感器选择神经网络使用PyTorch[32]进行实现。输入节点的数量（即特征数量）为1340。网络为多层感知机，除一对一层外均为权力娜姐；包含两个隐藏层结构，分别有400和100个节点。优化器选用随机梯度下降（SGD），二元交叉熵（BCE）被用作误差的评判标准，通过增加一对一层权重向量的的L1范数来进行L1正则化，故整体损失函数表示为式 4.4，其中λ是L1规则化的惩罚。调整λ可以改变稀疏程度并控制所选特征的数量，进而控制验证传感器组的大小。

参数方面，本文尝试了不同的优化器和学习率，进而发现，改变优化器对收敛性能的影响不大。本文对一对一层的学习率以及其他层的学习率进行分别调整，以获得更快的权重向量更新和更稀疏的权重向量。根据实验，与其他层相比，设置更高的一对一层的学习率可以获得更稀疏的结果。对于超参数（包括学习率和 λ），本文遍历了所有的候选参数，并对性能（f1-score）和传感器数量之间进行平衡考虑。详细来说，本文将f1-score基线设定为0.95，传感器数量基线设定为10。并通过计算一个分数$s = |f1 - 0.95|^{1.5} + |n_sensors - 10|$，从而鼓励高f1-score和低传感器数量，并通过挑

选分数最高的超参数来实现参数选取。

在SVM分类器方面，本文使用Python中的scikit-learn进行实现。由于负样本为时间轴上随机采样添加，其数量远远大于正样本，故引入了参数$class_weight$用于两类样本中应用不同的惩罚C，以避免样本数量的不平衡造成分类边界的偏差。

参数方面，根据测试，改变C对分类性能影响不大，而改变$class_weight$对模型测试结果有轻微影响：当$class_weight$从1开始增加时，结果有所改善，然而，在达到一定的阈值后，不仅模型测试结果很难随着$class_weight$增加而改善，反而训练时间成本也会增加。对于SVM中的参数（包括核函数、C和$class_weight$），使用网格搜索来遍历所有候选，并选取f1-score最高的组合。

4.4 实验评估

本节在上述EGuard实现的基础上，设计并进行了一系列实验评估，用于系统地测试并展示EGuard的性能。具体来说，本节的实验评估共分为以下四个部分：

1. 虚假事件检测准确率测试
2. 事件指纹准确率测试
3. 传感器数量调整测试
4. 传感器选择案例研究

4.4.1 虚假事件检测准确率

本测试主要关注EGuard对虚假事件攻击的检测性能。为得到真实确切的测试结果，本实验通过模拟发起事件欺骗攻击，以表明EGuard设法检测到伪造的事件。

本文使用的数据集包括事件记录和传感器数据。通过在数据集的事件记录中随机添加20个条目作为虚假事件，同时保持传感器数据不变，如此以来就可以模拟系统认为事件确实发生了，但传感器数据却没有相应变化的情况，这种情况也与真实的事件欺骗攻击相符。在添加虚假事件时，需要设置一个安全间隔（10秒），以避免与真实事件重叠造成混淆，从而造成传感器数据难以区分。

在上述模拟攻击后，本实验基于预先得到的被选传感器集合计算特征以建立学习样本，然后使用预训练的SVM分类器进行预测，上述过程模拟了EGuard的在线运行过程。对于测试集，本实验保留了原本测试集中的所有正样本，即真实发生的事件；同

表 4 Comparative evaluation of $F1-score$ and *number of sensors* of three event verification approaches

Event name	Real & Forged events (Sec. ??)		Real & Not happened events (Sec. 4.4.2)				Number of sensors	
	F1-score		F1-score		AUC			
	Peeves	EGuard	Peeves	EGuard	Peeves	EGuard	Peeves	EGuard
Window opened	0.9973	**1.0000**	0.9665	**1.0000**	**1.0000**	**1.0000**	32	**3**
Light off	0.9938	**1.0000**	**1.0000**	0.9442	**1.0000**	**1.0000**	27	**2**
Door closed	**0.9996**	0.9977	**1.0000**	0.9979	**1.0000**	**1.0000**	25	**2**
Fridge opened	**0.9893**	**0.9893**	**0.9883**	0.9217	**1.0000**	0.9982	29	**7**
PC on	0.9634	**0.9881**	0.9842	**0.9897**	0.9624	**0.9999**	**3**	5
Fridge closed	**1.0000**	0.9872	0.9883	**0.9893**	**1.0000**	0.9925	29	**6**
Door opened	**0.9992**	0.9858	**0.9982**	0.9974	**1.0000**	**1.0000**	20	**2**
Fan off	0.9805	**0.9842**	0.9500	**0.9616**	0.9908	**0.9968**	9	**2**
Window closed	**0.9866**	0.9839	**0.9649**	0.9088	**0.9999**	0.9995	25	**6**
Fan on	0.9395	**0.9804**	0.9238	**0.9746**	0.8878	**0.9989**	**2**	5
Coffee machine used	**1.0000**	0.9787	**1.0000**	**1.0000**	**1.0000**	**1.0000**	34	**5**
Light on	**0.9969**	0.9779	**1.0000**	0.9641	**1.0000**	**1.0000**	23	**5**
Screen off	0.9535	**0.9626**	0.8657	**0.9735**	0.8696	**0.9875**	**3**	5
Screen on	**0.9719**	0.9618	0.9004	**0.9537**	0.9514	**0.9577**	9	**4**
Camera on	0.9533	**0.9542**	0.7925	**0.8248**	0.9046	**0.9805**	11	**7**
Camera off	0.9180	**0.9268**	0.7648	**0.8223**	0.8997	**0.9340**	**10**	14
Shade up	0.7918	**0.9013**	0.4761	**0.9472**	0.5479	**0.9789**	13	**6**
Shade down	0.7660	**0.8928**	0.5032	**0.9140**	0.5820	**0.9703**	10	**9**
PC off	**0.9233**	0.8621	0.7269	**0.9665**	0.8539	**1.0000**	**4**	5
Radiator on	0.7552	**0.8303**	0.4894	**0.8473**	0.4687	**0.8180**	16	**8**
Doorbell used	**0.8460**	0.7927	0.4472	**0.8095**	0.7316	**0.8923**	19	**5**
Radiator off	0.7172	**0.7774**	0.2123	**0.5636**	0.4287	**0.8664**	10	**7**
Avg.	0.9292	**0.9416**	0.8156	**0.9214**	0.8672	**0.9714**	16.5	**5.4545**

时丢弃负样本，即未发生的事件，并使用先前注入的伪造事件作为代替的负样本进行测试。基于上述测试集的测试结果可以表明从物联网事件记录中检测潜在事件欺骗攻击的能力。除此以外，本实验对于Peeves[7]方案也实现了同样的模拟攻击。并评估其检测精度。

指标方面，本实验选取f1-score作为指标，结果如表4的左边部分所示。表4的右边部分还列出了用于事件认证的传感器数量。结果显示，我们可以看到，对于17/22个事件，EGuard获得了大于0.9的f1-score，并且对于所有事件而言，其f1-score均大于0.75。此结果表明这表明EGuard在检测潜在的事件欺骗攻击方面足够有效。此外，EGuard的平均f1-score比Peeves提高了0.0124。另外，考虑到EGuard使用的传感器要少得多（Peeves平均5.45个，而EGuard为16.5个），这表明了EGuard可以使用较小的验证传感器集合来获得较好的结果，体现了其轻量性。

4.4.2 事件指纹准确率

本测试主要关注EGuard构建的事件指纹是否能够真实准确地描述事件的物理行为。与上一实验不同的是，本测试直接使用原始数据集中分割出来的测试集，没有进行虚

假事件攻击的模拟。在测试集中，正样本代表事件真实发生，而负样本代表事件未发生。此结果可以表明EGuard建立的事件指纹的内在性能，即对事件物理影响的拟合程度。此外，本测试与Peeves进行的测试类似，使用与Peeves相同的数据集以及评估方法使此实验的横向评估更具说服力。

本实验直接使用SVM分类器给出的几个分类指标来显示EGuard的事件指纹性能。具体来说使用f1-score和AUC（曲线下面积），其中f1-score取自正样本和负样本f1-score的算数平均。对于AUC，我们绘制指接受者操作特征（Receiver Operating Characteristic，ROC）曲线，其以假阳性率（FPR）和真阳性率（TPR）为轴。对于ROC曲线来说，靠近左上角的曲线被认为是更好的结果，其代表当FPR小的时候，其TPR高，同时曲线下面积也大。

对于本实验指标选择的理由，在实验正负样本数量不平衡的情况下，f1-score相比传统的准确率等指标更为合理，而AUC侧重于分类器的内在性能，因为在绘制ROC曲线时考虑到了分类器判决阈值的不同取值。表4的右边部分显示了EGuard与Peeves的f1-score、AUC和传感器数量的对比。由结果可看出，EGuard方案中17/22个事件的f1-score大于0.9，这表明EGuard能够通过传感器数据建立有效的事件指纹模型；而我们发现Peeves在f1-score和AUC上分别平均落后0.106和0.104。

更深入的来说，EGuard在 ”暖气片打开”、”暖气片关闭”、”门铃使用”、”窗帘打开”和 ”窗帘关闭 ”五个事件上有很大改进。这是由于EGuard中的传感器选择方法选取了对分类任务贡献更高的传感器集合。从平均情况来讲，EGuard可以使用更小的验证传感器集来拟合事件的物理环境规则，体现了其轻量化的特点。

此外，本实验绘制了其中三个事件的ROC曲线（图7）。由图可得知，EGuard的曲线更接近于左上角，这表明EGuard的事件指纹模型具有更好的性能。

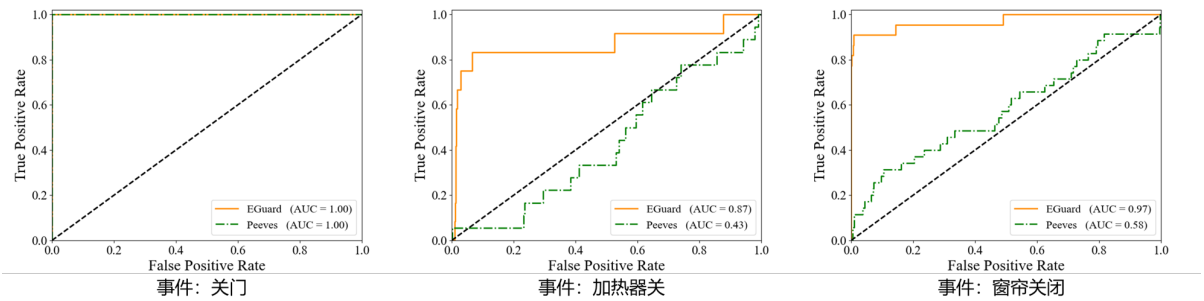

图 7 三个事件指纹的ROC曲线

4.4.3　传感器数量调整

本测试主要调整被选传感器的数量，并测试模型的准确性随传感器数量的变化规律。

在Peeves的实现部分，我们通过改变RMI的阈值来得到不同的传感器数量。对EGuard而言，其传感器数量可通过调整L1正则项系数λ来实现，更高的λ表示惩罚力度更高，得到的权重向量就会更稀疏，进而筛选掉更多的传感器特征。图8显示了传感器的数量对λ的变化规律。图上的每一个点，均为多次训练取平均的结果。

一般来说，传感器的数量会随着λ的增加而减少。需要注意的是，在某些事件（如相机关闭）中，较高λ值会带来离群点（红色矩形标记），这是因为高λ降低了损失函数中预测值和准确值之间误差的重要性，导致损失无法下降，使神经网络难以拟合到真实规律。因此本实验限制λ的大小来避免这些异常值。

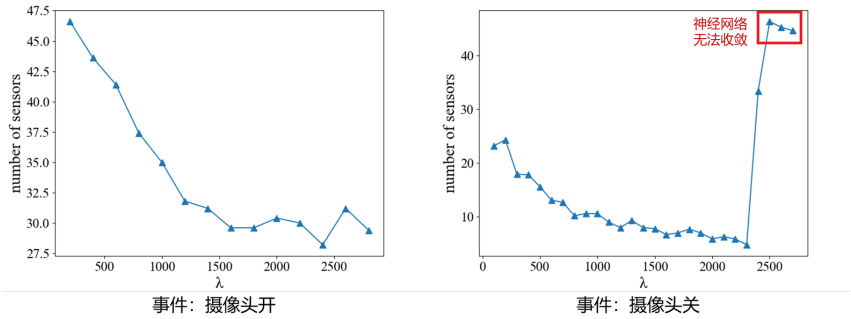

图 8　传感器数量随正则化系数λ的变化规律

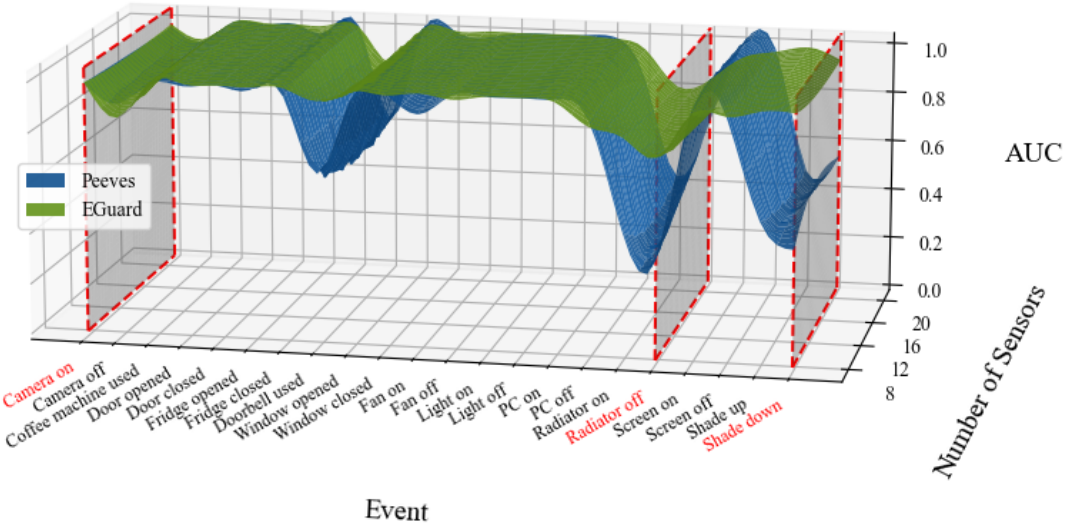

图 9　AUC随传感器数量变化规律的3D曲面图

图9以三维曲面图的方式展示了两种方案的AUC随每个事件的传感器数量之间的关系。从图中可看出，对于大多数事件，绿色平面均略微高于黄色表面，这意味

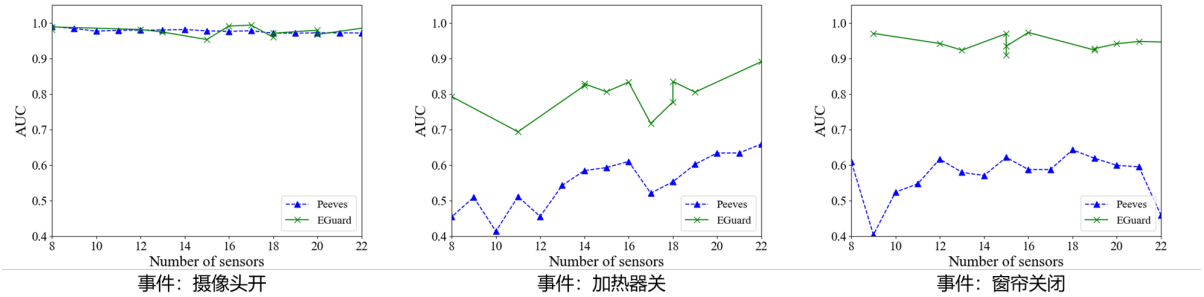

图 10 三个事件的AUC随传感器数量的变化规律

着EGuard在相同数量传感器的前提下获得更好的性能。进一步地，图10选取了图9中的几个事件截面。由此可以看出，在 ”摄像头打开”事件中，两种方法的效果差别不大，而在 ”散热器关闭 ”和 ”窗帘关闭”的事件中，EGuard性能较好。此外，此结果还表明，传感器的数量对分类性能的影响规律不明显。一方面是因为机器学习和神经网络的不确定性和随机性；另一方面可能是因为在本实验中，由于高λ造成收敛失败的限制，传感器数量的下界可能不够低。

总的来说，通过一个可调整的L1正则项系数λ，EGuard可以灵活调整被选传感器数量。且从横向对比来看，在传感器数量相同的前提下EGuard表现出更好的性能。

4.4.4 传感器选择案例研究

本测试展示了两个传感器选择结果的具体案例，用于更加细粒度地说明EGuard传感器选择结果的合理性。

首先，本实验通过计算传感器之间的相关系数，来研究对某事件而言各传感器之间的同质化程度。图11以热力图的形式展示了两种事件的传感器相关系数（取相关系数的绝对值），其中深色意味着高相关。此外，表5展示了传感器的具体选择结果和其对应的分类结果指标（AUC）。为了分析结果中传感器的同质性，本实验根据传感器相关系数对其进行聚类。具体来说，本实验以$1-|corr_{i,j}|$为传感器i和j之间的距离，并根据该距离进行层次聚类，如此以来即可将高相关的传感器划在同一簇中，同一簇中的传感器为事件指纹提供了同质化地信息。

案例1：事件”开门”。从图9a中，我们发现由热力图可看出，对于 ”开门 ”事件，传感器之间的相关性相对较低，但仍有一些深色的点，这意味着高度相关的传感器对数量较少。根据传感器选择结果显示，EGuard避免了在一个簇中选择多个传感器（簇A和簇B），并抛弃了Peeves选择的其他簇，因此大幅减少了传感器数量。

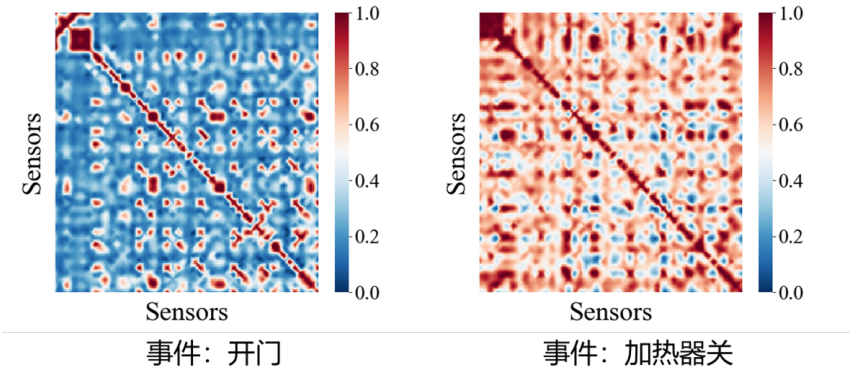

图 11 传感器间相关性热力图

为进一步说明，本实验还考察了被选传感器的物理意义。可看出EGuard只为”开门”事件选择了两个传感器。即pi7_BME680（门旁边的温度、湿度和气压传感器）和pi8_SenseHat（门上的加速度传感器）。基于常识，在门打开的过程中，门的加速度和门上的气压都会发生显著变化，故认为这两个传感器均为高度富信息的。并且实验表明仅仅使用这两个传感器就可以得到1的AUC，这意味着我们的方法在选择代表性传感器地同时避免了信息冗余。

案例2：事件”加热器关闭”。 热力图显示，对于”加热器关闭”事件而言，传感器间的相关性较高。可以看出在此情况下，EGuard选择了完全不同的传感器集，并显著提高了AUC。此结果意味着EGuard选择了更富信息、贡献更高的传感器，这是因为RMI指标并不能完全代表特征对分类器的实际贡献。

总体来说，本实验用上述2个案例来证明。EGuard能够在高相关度的情况下选择信息量更大的传感器集，以此提高事件认证性能；而在大多数情况下，EGuard能够避免信息冗余，选择更小的传感器集，实现轻量化的目的。

4.5　讨论

表 5 Verification sensor sets of different methods

(a) Door opened verification sensor sets. We cluster the sensors according to their correlation, and denote the clusters as capital letters. Sensors in the same cluster provide similar information.

Method	AUC	Verification sensor set	
EGuard	1.0000	*Cluster A:*	pi7_BME680
		Cluster B:	pi8_SenseHat
Peeves	1.0000	*Cluster A:*	pi10_BME680_IN, pi10_BME680_OUT pi1_microphone, pi2_microphone, pi3_microphone pi5_microphone, pi7_microphone, pi9_microphone pi3_BME680, pi7_BME680
		Cluster B:	pi1_BMP280, pi2_BMP280, pi5_BMP280 pi6_BMP280, pi8_SenseHat
		Cluster C:	pi1_TSL2560, pi7_TSL2560
		Cluster D:	pi1_RSS
		Cluster E:	pi7_RSS

(b) Radiator off verification sensor sets

Method	AUC	Verification sensor set
EGuard	0.8664	pi11_MLX90640, pi12_MLX90640, pi3_MPU6050 CamPower, pi4_MPU6050, Pi5_TSL2560, pi9_MPU6050
Peeves	0.4287	pi5_RSS, PCPower, pi10_BME680_IN pi1_BMP280, pi2_RSS, pi6_BMP280 FanPower, ScreenPower, pi7_RSS, WindowShadePower

[– This page is a preset empty page –]

结　论

学位论文的结论单独作为一章，但不加章号。如果不可能导出应有的结论，也可以没有结论而进行必要的讨论。

* 嗯，这就是你的论文了 *

[– This page is a preset empty page –]

参考文献

[1] Suresh S, Sruthi P V. A review on smart home technology[C]//2015 Online International Conference on Green Engineering and Technologies (IC-GET). IEEE, 2015: 1–3[2021-10-18]. DOI: 10.1109/GET.2015.7453832.

[2] Ricquebourg V, Menga D, Durand D, et al. The smart home concept: our immediate future[C]//2006 1st IEEE international conference on e-learning in industrial electronics. [S.l.]: IEEE, 2006: 23–28.

[3] Alam M R, Reaz M B I, Ali M A M. A review of smart homes—past, present, and future: volume 42[M]. 2012: 1190–1203[2021-10-18]. DOI: 10.1109/TSMCC.2012.2189204.

[4] Feng X, Wang Y, Jiang L, et al. A survey on internet of things security based on smart home[C]//2018 5th Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE). IEEE, 2018: 84–91[2021-10-18]. DOI: 10.1109/APWConCSE.2018.00022.

[5] Samsung. One simple home system. a world of possibilities[M]. [S.l.: s.n.].

[6] HomeAssistant. Home assistant[M]. [S.l.: s.n.], .

[7] Birnbach S, Eberz S, Martinovic I. Peeves: Physical event verification in smart homes[C]// Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2019: 1455–1467[2021-10-18]. DOI: 10.1145/3319535.3354254.

[8] Fu C, Zeng Q, Du X. Hawatcher: Semantics-aware anomaly detection for appified smart homes[C]//30th ${$USENIX$}$ Security Symposium (${$USENIX$}$ Security 21). [S.l.: s.n.], 2021.

[9] Fernandes E, Jung J, Prakash A. Security analysis of emerging smart home applications [C]//2016 IEEE Symposium on Security and Privacy (SP). IEEE, 2016: 636–654[2021-10-18]. DOI: 10.1109/SP.2016.44.

[10] Islam K, Shen W, Wang X. Security and privacy considerations for wireless sensor networks in smart home environments[C]//Proceedings of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design (CSCWD). IEEE, 2012:

626–633[2021-10-18]. DOI: 10.1109/CSCWD.2012.6221884.

[11] Sivaraman V, Chan D, Earl D, et al. Smart-phones attacking smart-homes[C]// Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks. [S.l.: s.n.], 2016: 195–200.

[12] Saxena U, Sodhi J, Singh Y. Analysis of security attacks in a smart home networks[C]// 2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence. [S.l.]: IEEE, 2017: 431–436.

[13] Coman F L, Malarski K M, Petersen M N, et al. Security issues in internet of things: Vulnerability analysis of lorawan, sigfox and nb-iot[C]//2019 Global IoT Summit (GIoTS). [S.l.]: IEEE, 2019: 1–6.

[14] Mao J, Zhu S, Liu J. An inaudible voice attack to context-based device authentication in smart IoT systems: volume 104[M]. [S.l.: s.n.], 2020: 101696.

[15] Zhang G, Yan C, Ji X, et al. Dolphinattack: Inaudible voice commands[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. [S.l.: s.n.], 2017: 103–117.

[16] Yan Q, Liu K, Zhou Q, et al. SurfingAttack: Interactive hidden attack on voice assistants using ultrasonic guided waves[C]//Proceedings 2020 Network and Distributed System Security Symposium. Internet Society, 2020[2021-10-18]. DOI: 10.14722/ndss.2020.24068.

[17] Liang C J M, Karlsson B F, Lane N D, et al. Sift: building an internet of safe things[C]// Proceedings of the 14th International Conference on Information Processing in Sensor Networks. [S.l.: s.n.], 2015: 298–309.

[18] Chi H, Zeng Q, Du X, et al. Cross-app interference threats in smart homes: Categorization, detection and handling[C]//2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). IEEE, 2020: 411–423[2021-10-19]. DOI: 10.1109/DSN48063.2020.00056.

[19] Ding W, Hu H, Cheng L. IoTSafe: Enforcing safety and security policy with real IoT physical interaction discovery[C]//Proceedings 2021 Network and Distributed System Security Symposium. Internet Society, 2021[2021-10-18]. DOI: 10.14722/ndss.2021.24368.

[20] Ding W, Hu H. On the safety of IoT device physical interaction control[C]//Proceedings of

the 2018 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2018: 832–846[2021-10-18]. DOI: 10.1145/3243734.3243865.

[21] University O. Peeves: Physical event verification in smart homes - ora - oxford university research archive[M]. [S.l.: s.n.], .

[22] Chimamiwa G, Alirezaie M, Pecora F, et al. Multi-sensor dataset of human activities in a smart home environment[J]. Data in Brief, 2021, 34: 106632.

[23] University W S. Welcome to casas[M]. [S.l.: s.n.], .

[24] Inc. K. Kaggle: Your machine learning and data science community[M]. [S.l.: s.n.].

[25] Casa N. Esphome — esphome[M]. [S.l.: s.n.].

[26] HomeAssistant. Rest api — home assistant developer docs[M]. [S.l.: s.n.], .

[27] Laput G, Zhang Y, Harrison C. Synthetic sensors: Towards general-purpose sensing [C]//Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. ACM, 2017: 3986–3999[2021-10-18]. DOI: 10.1145/3025453.3025773.

[28] Carpenter B. Lazy sparse stochastic gradient descent for regularized multinomial logistic regression[J]. Alias-i, Inc., Tech. Rep, 2008: 1–20.

[29] Tsuruoka Y, Tsujii J, Ananiadou S. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty[C]//Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. [S.l.: s.n.], 2009: 477–485.

[30] Ross B C. Mutual information between discrete and continuous data sets[J]. PloS one, 2014, 9(2): e87357.

[31] Gao W, Kannan S, Oh S, et al. Estimating mutual information for discrete-continuous mixtures[J]. Advances in neural information processing systems, 2017, 30.

[32] Facebook. Pytorch[M]. [S.l.: s.n.].

[– This page is a preset empty page –]

附　录

下列内容可以作为附录：

1) 为了整篇论文材料的完整，但编入正文又有损于编排的条理和逻辑性，这一材料包括比正文更为详尽的信息、研究方法和技术更深入的叙述，建议可以阅读的参考文献题录，对了解正文内容有用的补充信息等；
2) 由于篇幅过大或取材于复制品而不便于编入正文的材料；
3) 不便于编入正文的罕见的珍贵或需要特别保密的技术细节和详细方案（这中情况可单列成册）；
4) 对一般读者并非必要阅读，但对专业同行有参考价值的资料；
5) 某些重要的原始数据、过长的数学推导、计算程序、框图、结构图、注释、统计表、计算机打印输出文件等。

* 嗯，自由发挥吧 *

[– This page is a preset empty page –]

攻读硕士学位期间取得的学术成果

对于博士学位论文，本条目名称用“攻读博士学位期间取得的研究成果”，一般包括：

攻读博士学位期间取得的学术成果：攻读博士学位期间取得的学术成果：列出攻读博士期间发表（含录用）的与学位论文相关的学位论文、发表专利、著作、获奖项目等，书写格式与参考文献格式相同；

攻读博士期间参与的主要科研项目：列出攻读博士学位期间参与的与学位论文相关的主要科研项目，包括项目名称，项目来源，研制时间，本人承担的主要工作。

对于硕士学位论文，本条目名称用“攻读硕士学位期间取得的学术成果”，只列出攻读硕士学位期间发表（含录用）的与学位论文相关的学位论文、发表专利、著作、获奖项目等，书写格式与参考文献格式相同。

* 嗯，研究生不列科研项目 *

[– This page is a preset empty page –]

致　谢

致谢中主要感谢指导教师和在学术方面对论文的完成有直接贡献及重要帮助的团体和人士，以及感谢给予转载和引用权的资料、图片、文献、研究思想和设想的所有者。致谢中还可以感谢提供研究经费及实验装置的基金会或企业等单位和人士。致谢辞应谦虚诚恳，实事求是，切记浮夸与庸俗之词。

* 嗯，感谢完所有人之后，也请记得感谢一下自己 *