基于Rowhammer 物理不可克隆函数的物理设备测绘框架

刘 镝 徐闻含 王文东 李大伟* 关振宇 刘建伟
(北京航空航天大学网络空间安全学院 北京 100191)

摘 要：网络空间测绘的核心问题是准确识别和动态跟踪设备。然而，随着匿名化技术的发展，设备可以拥有多个IP地址和MAC地址。这使得通过传统的测绘技术将多个虚拟属性映射到同一个物理设备上变得更加困难。该文提出一种基于物理不可克隆函数(PUF)的测绘框架。该框架可以主动检测网络空间中的物理资源，并根据物理指纹构建资源画像来动态跟踪设备。同时，该文提出一种在配备第四代双倍速率(DDR4)内存的个人电脑(PC)上实现基于Rowhammer的动态随机存取存储器物理不可克隆函数(DRAM PUF)的方法。性能评估表明，该方法在PC上提取的Rowhammer PUF的响应是唯一且可靠的，并可以作为设备的唯一物理指纹。实验结果表明，即使目标设备修改了MAC地址、IP地址或重装了操作系统，该文提出的框架仍然可以通过构建一个用于设备匹配的物理指纹数据库，对目标设备进行准确的标识。

关键词：网络空间测绘；物理指纹；物理不可克隆函数；Rowhammer物理不可克隆函数
中图分类号：TP333; TN801 **文献标识码**：A **文章编号**：1009-5896(2023)09-3200-10
DOI: 10.11999/JEIT230388

Detecting and Mapping Framework for Physical Devices Based on Rowhammer Physical Unclonable Function

LIU Di XU Wenhan WANG Wendong LI Dawei
GUAN Zhenyu LIU Jianwei
(*School of Cyber Science and Technology*, *Beihang University*, *Beijing* 100191, *China*)

Abstract: The core problem of cyberspace mapping is to identify accurately and track dynamically devices. However, with the development of anonymization technology, devices can have multiple IP addresses and MAC addresses. This makes it increasingly difficult to map multiple virtual attributes to the same physical device through traditional mapping techniques. In this paper, a mapping framework based on Physical Unclonable Function (PUF) is proposed which can actively detect physical resources in cyberspace and track dynamically devices based on physical fingerprints to construct resource portraits. Furthermore, a new method is proposed to implement the Rowhammer-based Dynamic Random-Access Memory Physical Unclonable Function (DRAM PUF) on a regular Personal Computer (PC) equipped with Double Data Rate Fourth (DDR4) memory. Performance evaluation shows that the response extracted from the Rowhammer PUF on the PC using the proposed method is unique and reliable, and can be used as a unique physical fingerprint of the device. Experimental results show that even if the target device modifies its MAC address, IP address, or reinstalls operating system, the framework proposed in this paper can still accurately identify the target device by constructing a physical fingerprint database for device matching.

Key words: Cyberspace mapping; Physical fingerprint; Physical Unclonable Function (PUF); Rowhammer PUF

收稿日期：2023-05-08；改回日期：2023-08-28；网络出版：2023-08-31
*通信作者： 李大伟 lidawei@buaa.edu.cn
基金项目：国家重点研发计划(2021YFB2700200)，国家自然科学基金(62372022, 62002006, U2241213, U21B2021, 62172025, 61932011, 61932014, 61972018, 61972019, 61772538, 32071775, 91646203)
Foundation Items: The National Key R&D Program of China (2021YFB2700200), The National Natural Science Foundation of China (62372022, 62002006, U2241213, U21B2021, 62172025, 61932011, 61932014, 61972018, 61972019, 61772538, 32071775, 91646203)

1　背景介绍

网络空间测绘是指对网络空间中的各种资源进行探测、获取资源的各种属性、分析资源之间的关系，并通过逻辑拓扑或网络地图等形式将资源属性和分析结果呈现出来的过程。其目的在于获取网络空间资源属性的整体态势，掌握资源之间的交互关系和发展趋势，以及感知网络空间的动态变化。网络空间测绘的对象为网络空间中的各种物理资源和虚拟资源，而探测的范围可能为整个互联网、某个局域网或一个特定的网络范围。

之前的研究[1–3]总结了网络空间资源测绘的相关概念和技术，但未给出具体的测绘实例。传统的资源测绘技术通过发送请求数据包来获取目标资源的IP地址、端口号等信息，使用的工具包括Nmap[4], Zmap[5], Masscan[6], Shodan[7]等。但这些测绘工具只集中于网络层测绘，无法获得物理设备独有的物理特性。一旦目标设备更改其IP地址等虚拟属性，这些工具会视其为新设备，导致资源丢失或重复测绘，并且不能对目标设备进行持续追踪。本文提出的测绘框架专注于物理层，通过真实局域网环境下的物理资源测绘，提取资源的物理指纹、构建资源画像、获取资源间行为关系，并准确标识和追踪目标资源。

该框架为暗网测绘带来新的方法，因为某些配置了多个网卡的设备常同时接入互联网和暗网，但它们在不同网络中表现出来的身份及属性不同，难以通过传统测绘方法标识。然而，这些设备在每次连接不同网络时都使用相同的内存模块，因此基于内存模块的物理指纹不会改变。在互联网和暗网中分别进行测绘，通过比较设备唯一的物理指纹来识别位于互联网和暗网边界的设备，实现暗网的准确测绘。

本框架也可用于组织内部的网络准入管理。组织内部网络实行白名单准入机制，只有经过认证的设备才能够获得网络访问权限。组织管理员为组织内每台白名单设备生成设备指纹，唯一标识该设备。由于每个动态随机存取存储器(Dynamic Random-Access Memory, DRAM)都不可克隆，攻击者将无法生成合法指纹以伪造白名单设备。即使攻击者物理接触白名单设备并将其DRAM更换至恶意设备中，也会由于不同CPU缓存模式与内存地址映射的差异而导致无法生成合法物理不可克隆函数(Physical Unclonable Function, PUF)响应，从而保障网络准入机制的正常运行。

此外，本框架还可用于软件授权场景，收费软件可能会限制单个授权下的可用设备数，利用本框架唯一标识设备，有效防止使用者篡改设备序列号，生成多个相同指纹的设备，以绕过设备数限制。

1.1　挑战

(1)物理指纹获取。主要存在两个难点：一是由于测绘目标是未知的且无法物理接触，需要通过一种可靠的远程代码执行和数据回传的方法来完成测绘；二是需要获取能唯一标识目标设备的物理指纹，且保证获取的指纹是稳定和可重复的。同时，生成物理指纹的过程不应该影响目标设备的正常运行，例如导致目标设备重启或服务异常。

(2)设备动态标识和追踪。完成测绘后，虽然可以得到资源的画像和拓扑关系，但网络空间中的数据流动和资源变化随时都在发生。因此，需要对资源进行定期测绘来更新数据并掌握网络空间的发展态势。IP地址、MAC地址等虚拟属性是易于更改的，但我们希望通过物理指纹持续标识和追踪目标设备，忽略这些变动带来的影响。

1.2　本文贡献

(1)本文提出一种基于物理不可克隆函数(PUF)的测绘框架。通过设计的PUF算法提取目标设备的物理指纹，并结合设备的其他属性构建设备画像，并使用知识图谱的形式实现测绘结果的可视化。

(2)本文实现了一个基于Rowhammer的DRAM PUF，可在配备第四代双倍速率(Data Rate Fourth, DDR4)内存和运行Linux操作系统的个人电脑上运行。访问内存模块运行PUF实现代码即可，无需额外添加硬件。

(3)本文在两种指标下评估了Rowhammer PUF的性能。评估结果表明，所实现的PUF表现出接近理想值的唯一性和良好的可靠性，能够完美地区分不同的设备。据我们所知，本文所做工作是第1个在个人电脑上实现和评估Rowhammer PUF的研究。

(4)本文在局域网环境中进行了3次测绘实验。实验结果表明，即使目标设备更改了MAC地址、IP地址甚至操作系统等虚拟属性，通过物理指纹仍能对目标设备进行识别和跟踪。

2　背景知识

2.1　网络空间测绘

网络空间测绘是一个探测、分析、绘制和应用的完整过程[1]。首先，通过各种探测技术获取网络空间物理资源和虚拟资源的数据，包括资源属性值等相关信息。其目的是为后续的分析过程提供基础数据。例如，使用网络扫描软件获得目标设备的IP地址、MAC地址、端口号、流量包等数据。分析是从探测得到的数据中提取资源属性，对资源进

行标识，构建资源画像的过程。其目的是形成关于网络空间资源的知识库，为系统提供大网测绘数据、互联网暴露资产及其脆弱性的数据等，有助于网络资产云监测、漏洞响应等业务的开展[2]。例如，通过数据挖掘等手段对探测得到的数据进行分析，提取出关键词和主题，构建出有关网络安全的知识库。可视化是基于网络空间资源的属性和映射关系构造网络拓扑并进行可视化的过程。其目的是动态显示网络空间资源的分布和状态，帮助用户理解网络空间的结构和运作机制。例如，使用图论等算法构建网络拓扑，以便直观地展示网络空间中各项关键指标的分布情况，从而为分析和决策提供支持。

测绘的目标对象包括物理资源和虚拟资源。物理资源包括诸如路由器、交换机、基站等交换设备，以及移动电话、计算机、摄像头等接入设备。虚拟资源包括域名系统(Domain Name System, DNS)、邮件等虚拟服务，以及视频、文本、社交媒体账号等虚拟内容[1,3]。网络空间资源具有的特定属性反映了资源的特征和资源之间的关系。资源属性可以根据物理层、逻辑层、认知层等层次进行划分。物理层反映了资源的物理属性，比如设备坐标、尺寸、设备的具体信息等。逻辑层反映了资源的网络属性，比如IP、域名、端口号、操作系统等。认知层反映了资源的社会属性，比如服务类型、服务对象和组织等[1]。

2.2 Rowhammer

Rowhammer是一种DRAM内存利用技术，自2014年起研究人员对其进行了广泛的研究和应用。表现形式为，当重复且快速访问内存中的某一行(内存访问行)时，其邻接行(受害者行)中的某些比特位置可能会翻转。这是由于重复且快速访问某一DRAM行会加速邻接行的电荷泄露过程[8]。为了快速访问DRAM内存行，需要绕过CPU和内存之间的缓存机制。一般使用的方法有使用clflush, clflushopt指令刷新缓存[9]，使用缓存逐出指令回收缓存[10,11]，或使用未缓存的内存[12]等。要成功获得比特翻转，首先需要确定虚拟地址映射到物理地址再到DRAM内存地址之间的关系，然后确定合适的内存访问模式以实现可靠的比特翻转。

DRAM芯片如图1所示，分为w个Bank，每个Bank由d个cell(存储单元)组成。这样的芯片储存了$d{\cdot}w$ bit信息。存储单元组织成阵列。每个存储单元都有自己的地址：(i,j)。i表示行，j表示列。图1中的DRAM芯片有8个Bank(所含字节的位数)，每个Bank由16个cell组成，4行4列。信息可以通过引脚进行传输。每个引脚携带1 bit的信号。图1中有两组引脚，2个地址引脚，用于传输超单元的行与列的地址。8个数据引脚，可传送8 bit的信息。DRAM芯片会与内存控制器相连，内存控制器可以与DRAM芯片进行数据的传送，一次可传送8 bit的信息。每个存储单元包括一个晶体管和电容器。对于true cell，一个充满电的电容表示1 bit数据“1”，一个完全放电的电容表示1 bit数据“0”。Anti cell 则完全相反[13] 。

虚拟地址和物理地址之间的映射由操作系统通过访问pagemap接口或者分配hugepage来完成。但是物理地址到DRAM内存地址的映射关系却不是完全公开的。AMD公司公开了地址映射函数，但英特尔公司没有。对于英特尔CPU，需要通过逆向工程的技术来得到内存地址的映射函数[14]。

DRAMA[15]和DRAMDig[16]是两种纯软件实现的DRAM地址映射逆向工具，这两个工具都通过时间侧信道来确定内存地址中的bank位。具体来说，DRAM中的每个bank都有一个用于访问内存

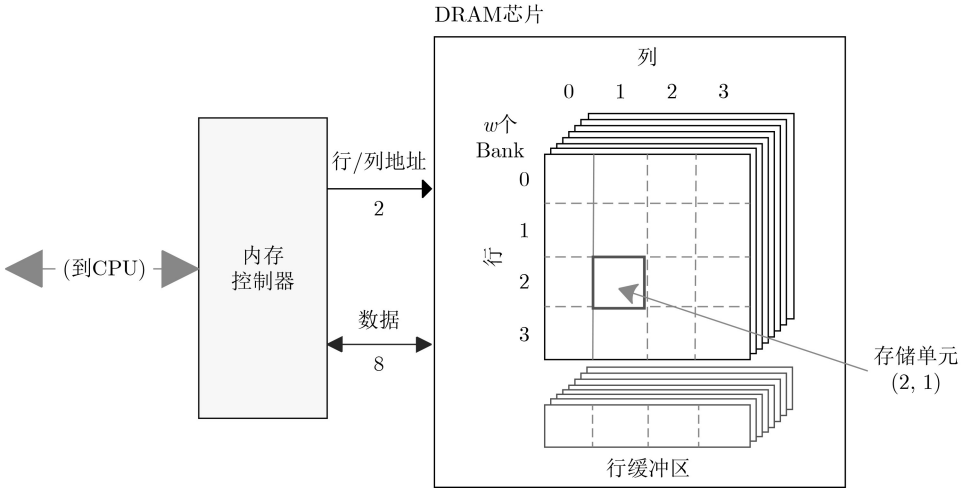

图 1 128 bit的DRAM芯片的高级视图

行的row buffer。通过选择一些地址对，重复交替进行访问，并测量平均访问时间，如果一对地址位于同一bank中，则它们的访问平均时间会显著高于那些不在同一bank中的地址对，这种现象被称为row buffer冲突。因此，可以通过这种方法找到位于同一bank中的地址集合并确定内存地址中的bank位。

内存访问模式是指同时访问的内存行的组合方式。不同的访问模式导致的比特翻转发生的概率差别很大。通过选择合适的内存访问模式，可以获得更多的比特翻转数量。目前主流的访问模式有以下4种：

(1)One-location访问模式。只重复访问bank中的某一行，因此不会导致row buffer冲突的发生[17]。这种模式导致的比特翻转的数量较少，但是在一些情况下可以绕过基于内存访问模式分析的保护机制，并成功导致比特翻转。

(2)Single-sided访问模式。重复访问bank中的某两行，但是这两行之间没有特定的位置关系。

(3)Double-sided访问模式。重复访问bank中满足邻接关系的两行，即访问的两行与同一受害者行相邻接[18]。这种模式可以提高比特翻转的概率。对于大多数缺少目标行刷新(Target Row Refresh, TRR)机制的DDR3内存来说，Single-sided和Double-sided这两种模式都可以导致比特翻转。但是对于具有TRR机制的DDR4内存，很难通过这两种模式导致比特翻转。

(4)Many-sided访问模式。重复访问bank中的多行，且任意两个内存访问行之间满足与Double-sided访问模式一样的邻接关系。Frigo等人[19]首先提出了一种Many-sided Rowhammer工具，可以自动识别访问模式，并通过抑制DRAM中的TRR机制对DDR4 DRAM进行可靠的位翻转。由于TRR采样器只能跟踪有限数量的攻击者行，一旦攻击者行的数量超过TRR采样器的大小，攻击者可能成功诱发位翻转。

此外，之前的一些研究表明[8,20]，在DRAM中发生比特翻转的位置是比较稳定的。并且，DRAM在制造过程中引入的随机差异导致即使相同结构的两个DIMM也会具有不同的比特翻转位置。因此，这种Rowhammer导致的比特翻转效应可以被用来实现物理不可克隆函数(PUF)。

3　基于PUF的测绘框架

要实现动态跟踪网络空间中物理设备的目标，需要一种方法来识别设备，即使设备的IP地址、地理位置甚至操作系统发生变化。一种实现方法是获取设备的物理指纹，并将其与设备的其他属性进行绑定。本文提出了一种利用设备DRAM中实现Rowhammer PUF的算法，以获取设备的唯一物理指纹。具体的测绘实施方案如图2所示。

3.1　基于PUF的指纹生成

在测绘框架中，PUF的功能是为网络空间中的物理资源分配唯一的物理指纹。通过发送构造的PUF挑战到网络空间中的物理设备，获得对应的PUF响应，并将其转化为设备的物理指纹。

传统的PUF很难适用于本文的测绘场景。由于测绘到的网络空间资源是未知的且无法物理接触，因此那些基于FPGA专用电路的PUF无法满足测绘的实际需求，例如仲裁器PUF[21]。一个合适的选择是通过网络空间资源自身的部件来实现PUF，比如静态随机存取存储器(Static Random-Access Memory, SRAM)和DRAM，这些部件被广泛用于互联网设备中，特别是个人计算机，需要用到DRAM作为内存。

设备的正常运行在测绘过程中不应受到影响。具体来说，获取设备的物理指纹的过程不应导致设备重新启动或崩溃。因此，那些需要在设备的启动阶段利用初始化的随机特性来获得响应的固有PUF[22]也不符合测绘的需求。

最近，许多研究人员对DRAM内存的比特翻转特性进行了较为全面的研究。网络空间中的大多数物理资源都具有DRAM内存部件，因此DRAM内存可以被视为网络空间中物理资源的共同特征。因此，本文采用Rowhammer技术在设备的DRAM内存中实现Rowhammer PUF，可以在设备运行时

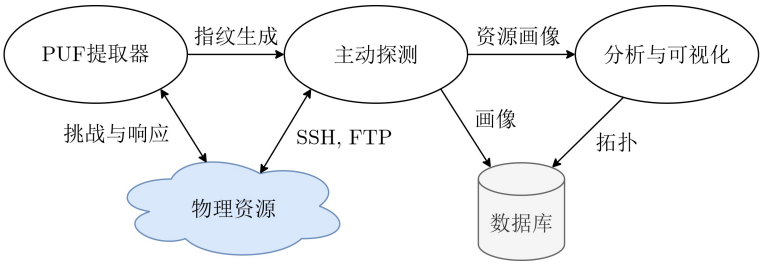

图 2 基于PUF的测绘框架

获得其唯一的物理指纹，并且不会干扰设备的正常运行。

3.2 基于Rowhammer PUF的主动探测

在此过程中，需将特殊设计的PUF挑战以及Rowhammer PUF实现代码发送到目标设备，以获取其物理指纹。同时，可以通过构建的脚本来获取目标设备的CPU型号、内存型号、操作系统版本、IP地址、网络行为和其他设备属性信息。

为获得设备的物理指纹，必须在目标设备上运行构造好的PUF响应代码。针对运行Linux操作系统的设备，可以利用Linux远程代码执行漏洞，在目标设备上运行构造好的代码。通过关键词“linux”和“remote”在CVE(Common Vulnerabilities and Exposures)网站上搜索可得到，此类漏洞版本跨度大，覆盖范围广，从CVE-1999-0002到CVE-2023-29257共有1 295个。通过该方式，可将代码远程植入目标设备，执行代码并获取设备的物理指纹和网络属性，最终将得到的信息返回以进行后续分析。

3.3 资源画像和拓扑分析

物理资源在网络空间的画像可以通过图3展示的信息来描述。在资源画像中，除了物理指纹属性不会改变，其他属性都可以修改，因此其余属性也可以被称为虚拟属性。物理指纹用于唯一标识物理资源，而虚拟属性用于描述物理资源之间的拓扑关系。在这里，本文没有按照级别来划分属性，而是将资源的所有属性放在一个画像中。网络层中资源之间的拓扑关系可以分为IP接口层、路由器层和PoP(Presence of Point)层拓扑发现。资源拓扑可以以知识图谱的形式进行可视化。图3中的椭圆形节点表示资源实体，圆形节点表示资源的画像，资源之间的关系由椭圆形节点之间的边来表示。

4 Rowhammer PUF及其评估

本节介绍了Rowhammer PUF的实现过程，并在两个评价指标下对PUF的整体性能进行了评估。

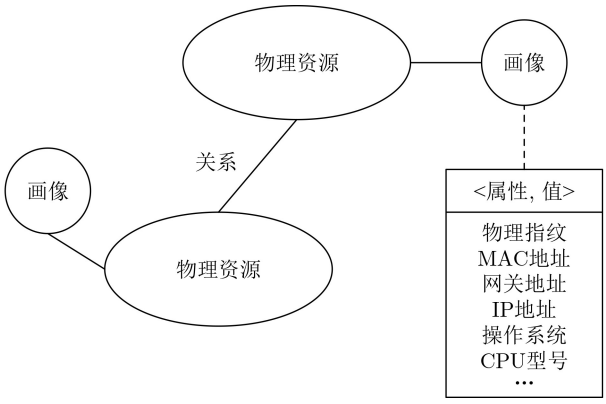

图 3 资源画像及拓扑

4.1 PUF实现

本文基于Frigo等人[19]提出的many-sided rowhammer工具在运行Linux操作系统的个人电脑上实现了Rowhammer PUF。我们首先定义PUF的挑战和响应，然后提供实现PUF的算法。

PUF挑战由多个参数组成，包括PUF地址、内存访问模式、数据模式、测量次数等。在算法实现中，每个参数都会影响PUF的响应值，通过选择合适的挑战参数，可以获得更稳定的响应。需要考虑的参数包括：

(1)PUF地址。PUF地址由行基址和地址偏移量组成。行基址定义为每个bank的第1行，通过更改地址偏移量来更改PUF的起始地址。PUF地址的选取对PUF挑战有直接的影响，因此对于不同的PUF挑战，PUF地址不应相同。

(2)Hammering模式。本文使用Many-sided内存访问模式，其中变量是内存访问行的数量n，其邻接行是受害行(即PUF行)。PUF地址和内存访问模式共同决定了PUF的大小，对于不同的PUF挑战，内存访问行和PUF行不应重叠。

(3)数据模式。数据模式由写入内存访问行和PUF行的初始值决定。由于内存单元存在两种不同的比特逻辑，该参数也对PUF的响应值有很大影响。通过为内存访问行和PUF行选择合适的初始值，可以获得更多的比特翻转。例如，将内存访问行初始化为0x00，将PUF行初始化为0xFF，或分别将二者初始化为0x55和0xAA。

(4)测量次数。测量次数包括访问的bank数量以及访问每个内存访问行的次数，对于每个bank，PUF地址、内存访问模式和数据模式都应该是相同的设置。

PUF挑战中参数的设置决定了PUF响应，本文使用发生位翻转的cell位置集作为PUF响应。更直观地说，PUF响应集合中的一个元素表示发生比特翻转的位置，其格式为(bank,row,column) = (x,y,z)，表示在第x个bank中的y行和z列中发生比特翻转。

PUF查询是指通过输入PUF挑战得到PUF响应的过程。Rowhammer PUF的查询过程在算法1中进行了描述，其中变量m是对测量次数的计数，变量b是对bank数量的计数。首先根据PUF地址、内存访问模式和数据模式初始化内存访问行和PUF行。定义的每个bank都需要相同的初始化操作，然后在定义的各个bank中，根据内存访问模式访问定义的内存访问行。之后扫描PUF行并输出发生比特翻转的位置，最后所有比特翻转位置的集合就构成了PUF响应。

算法1　Rowhammer PUF查询过程

输入：PUF地址、内存访问模式、数据模式、测量次数
输出：PUF响应
分配所需的内存；
while $m <$ 测量次数 do：
　while $b <$ bank数量 do：
　　初始化内存访问行和PUF行；
　end
　while $b <$ bank数量 do：
　　快速、重复访问所有的内存访问行；
　　扫描所有的PUF行并输出比特翻转的位置；
　end
end

4.2　PUF评估

本文对两块相同设计、规格和生产批次的DIMM进行了Rowhammer PUF的可靠性和唯一性测试。测试计算机的CPU型号是Intel(R) Core (TM) i7-10700 CPU @ 2.90 GHz, CPU架构是Comet lake，操作系统是Ubuntu 20.04。其中内存采用了大小为8 GB、频率为2 932 MHz的三星DDR4 SDRAM，没有ECC(Error Checking and Correcting)功能。

DRAM PUF和SRAM PUF之间的一个区别在于，DRAM PUF的响应反映的是内存中比特翻转的位置。因此，经典指标中使用的汉明距离并不适合于评价DRAM PUF的特性。本文使用Jaccard指标来评价Rowhammer PUF的可靠性和唯一性[23]，其中获得的PUF响应集合用S表示。计算两个PUF响应集合之间相似性的Jaccard指标公式为

$$\text{Jaccard}(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} \tag{1}$$

可靠性衡量同一PUF在不同测量中对相同挑战的响应之间的差异。本文使用指标$J_{\text{intra}}(S_1, S_2)$表示Rowhammer PUF的可靠性。由于存在测量噪声，每个测量结果都不完全相同，但理想情况下，两个集合中包含的比特翻转位置应该相同，因此J_{intra}的理想值为1。

唯一性衡量不同PUF在不同测量中对相同挑战的响应之间的差异。本文使用指标$J_{\text{inter}}(S_1, S_2)$表示Rowhammer PUF的唯一性。理想情况下，两个集合中包含的比特翻转位置不应重叠，因此J_{inter}的理想值为0。

在测试中，每一次PUF查询都使用相同的PUF挑战，其参数设置如表1所示。本文将PUF地址固定在每个bank的第1行，一共测量了5个bank。由于某些 DDR4 DRAM具有TRR机制，因此某些具有比特翻转的行可能会在下一次测量中被刷新，从而导致没有比特翻转。为此，本文将内存访问行数设置为22行，并在每次PUF查询中进行10次测量。PUF行和内存访问行分别初始化为0x55和0xAA，以满足邻接行对应的cell的值是相反的，这样可以满足导致比特翻转的条件。

在给定的PUF挑战参数下，本文对两块DIMM进行了20次PUF查询，随机选择2次查询结果构成集合S_1和S_2，并测试了所有可能的组合。两块DIMM之间的唯一性为0，DIMM1的平均可靠性为0.62，DIMM2的平均可靠性为0.63，评估结果如图4所示。

在测绘的主动探测过程中，需要使用Rowhammer PUF为网络空间中的物理资源分配唯一的物理指纹，并对目标设备进行标识和追踪。根据在PUF评估中的观察结果，不同PUF查询中包含的比特翻转位置的数量是不一样的。当各集合的大小相差很大时，通过原始Jaccard指标计算出的值无法准确反映PUF可靠性的实际情况。

原始的Jaccard指标并不适合在测绘场景中用于计算目标设备的相似性。因此，本文修改Jaccard指标为

$$\text{Jaccard}'(f, S_{\text{d}}) = \frac{|f \cap S_{\text{d}}|}{|f|} \tag{2}$$

其中，S_{d}表示从前几个查询中获得的PUF响应集合的组合，例如$S_{\text{d}} = S_1 \cup S_2 \cup S_3$。$S_{\text{n}}$表示新查询的PUF响应集。

使用Jaccard′指标获得的评估结果如图5所示。

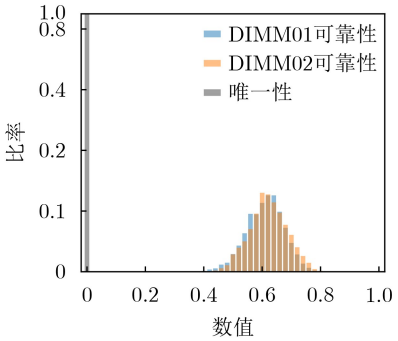

图 4 Jaccard指标下的唯一性和可靠性评价

表 1　PUF 参数设置

参数	值
PUF地址	行基址 = 0，地址偏移 = 0
内存访问模式	22 sided
数据模式	PUF行 = 0x55，内存访问行 = 0xAA
测量次数	5个bank, 10次测量

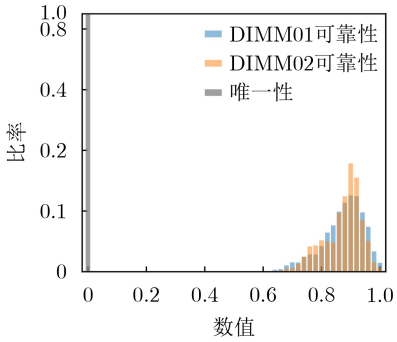

图 5 Jaccard′指标下的唯一性和可靠性评价

本文从20个PUF响应查询中任意选择3个查询结果来形成数据库S_{d}，然后将剩余的其他单个查询结果作为S_{n}。并测试了所有可能的组合，得到了两块测试DIMM的可靠性以及两块DIMM之间的唯一性。在现有的测试数据下，两块DIMM之间的唯一性值为0，DIMM1的可靠性平均值为0.88，DIMM2的可靠性平均值也是0.88。结果表明唯一性和可靠性值的分布差异非常大，这意味着可以完美地区分和标识两个不同的设备。

5 实验分析

本文在实验室环境中进行了3组测绘实验，以模拟真实世界中的测绘场景。实验结果表明，本文提出的测绘框架能够准确识别网络空间中的设备，并获得目标设备的画像。

5.1 实验设置

测绘实验架构如图6所示。实验网络环境为局域网环境，网关地址设置为192.168.171.1。以笔记本电脑作为探测主机，部署Nmap, Zmap等传统网络扫描软件，Hydra, MetaSploit等漏洞利用软件，以及FTP(File Transfer Protocol)文件共享软件等。其他4台计算机作为待探测的目标设备。

本文进行了3组实验，每组实验分为以下3个步骤：

(1)获取系统权限。通过安全外壳协议(Secure Shell Protocol, SSH)服务的漏洞获取当前操作系统的超级管理员权限。

(2)获取目标设备的物理指纹。使用第1步中获得的超级管理员权限，执行构造的PUF实现代码，以获取目标设备的物理指纹和其他属性，并通过FTP服务将结果回传探测主机。此步骤需要收集目标设备尽可能多的PUF响应，以形成目标设备的物理指纹数据库。

(3)分析和可视化。分析在第2步获得的数据，形成目标设备的物理指纹数据库，构建目标设备的画像。分析目标设备之间的网络行为并可视化网络拓扑。

本文设置多组对照实验，通过更改目标设备的虚拟属性，例如IP地址和MAC地址等，对比证明本文所涉及方案可以通过探测到的物理指纹唯一准确识别目标设备。

5.2 3组对照实验

第1组实验使用4台计算机作为目标设备。其中A1和A2模拟用户使用的终端电脑，A3模拟网站服务器，A4模拟数据库服务器，为网站提供服务。通过分析扫描的端口数据，可以得到设备之间的拓扑结构，如图7所示。

根据探测结果中绘制的目标设备画像如表2所示。目标设备的画像由物理指纹、MAC地址、IP地址、端口号、网关、内存序列号、操作系统版本和CPU型号组成。对于每个目标设备，探测过程中收集足够的PUF响应，以形成相应的物理指纹数据库，数据库存储与PUF挑战对应的比特翻转位置的结果。

第2组实验修改4台设备的操作系统MAC地址、IP地址和开放的服务，然后执行新的探测，获得的新目标设备的画像如表3所示。

对于传统的网络探测方法，这4个设备将被标识为4个新设备B1, B2, B3和B4。本文的目标是识别新探测到的目标设备和先前探测到的设备之间的对应关系。换句话说，即使目标设备更换了其所有的虚拟属性，仍然可以通过其唯一的物理指纹来识别目标设备。

将新探测到的目标设备的PUF查询结果与先前探测到的设备的物理指纹数据库进行匹配，根据Jaccard′指标计算的结果如表4所示。结果表明，虽然设备A1改变了其IP地址、MAC地址和操作系统

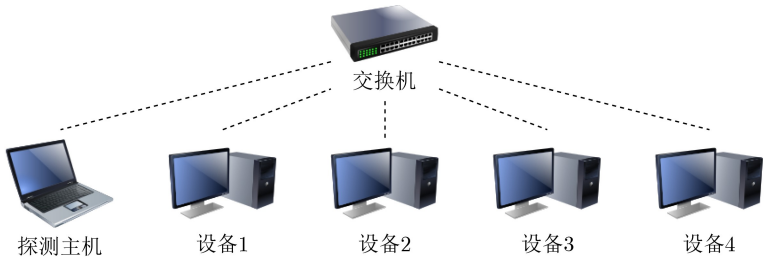

图 6 实验架构

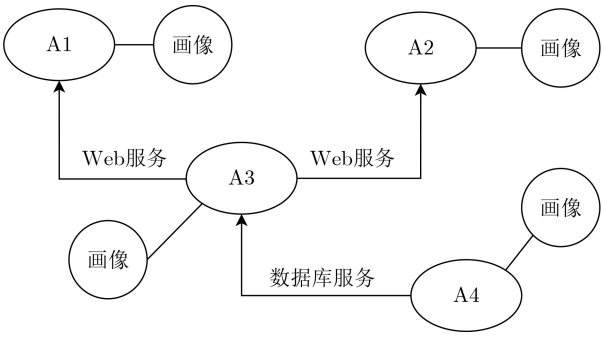

图 7 网络拓扑

版本，但仍然可以通过物理指纹识别出设备A1。设备B1是以前的设备A1，而设备B2是以前的设备A3，设备B3是以前的设备A2，设备B4仍然是以前的设备A4。

在第2组实验的基础上，用新设备替换了第2组实验中4个设备中的两个，另外两个设备保持不变。操作系统版本、MAC地址、IP地址、端口等属性也保持不变，与表3一致。

实验3的目标是标识出新添加的设备，新设备的所有虚拟属性与第2组实验中的设备相同。根据Jaccard′指标计算的结果如表5所示，可以知道C1和C3是新添加的设备，设备C2和C4是以前的设备B2和B4。

6　相关工作

Schaller等人[24]提出了第1个利用Rowhammer效应实现PUF的工作。他们在装有DDR2内存的PandaBoard上实现了Rowhammer PUF，并测试了PUF在不同条件下的性能。这项工作是开创性的，给我们的工作带来了很多启发。但PandaBoard上Rowhammer PUF的实现不需要考虑地址映射问题

表 2　实验1目标设备画像

	A1	A2	A3	A4
物理指纹	数据库1	数据库2	数据库3	数据库4
MAC地址	f4:4d:30:d0:f1:32	f4:4d:30:d0:f2:bf	f4:4d:30:32:2a:fb	f4:4d:30:82:73:91
IP地址	192.168.171.100	192.168.171.122	192.168.171.125	192.168.171.147
端口号	22:sshd	22:sshd	22:sshd; 80:http; 443:https	22:sshd; 3306:mysql
网关	192.168.171.1	192.168.171.1	192.168.171.1	192.168.171.1
内存序列号	M378A1K43 DB2-CVF	M378A1K43 DB2-CVF	M378A1K43 DB2-CVF	M378A1K43 DB2-CVF
操作系统版本	Ubuntu 20.04.3 LTS			
CPU型号	Intel(R) Core(TM) i7-10700 CPU @ 2.90 GHz			

表 3　实验2目标设备画像

	B1	B2	B3	B4
物理指纹	PUF查询1	PUF查询2	PUF查询3	PUF查询4
MAC地址	c0:c6:f7:cf:08:0e	01:e7:19:da:a2:5c	f6:d9:68:a1:88:03	00:86:af:33:84:66
IP地址	192.168.171.100	192.168.171.8	192.168.171.215	192.168.171.216
端口号	22:sshd	22:sshd	22:sshd; 80:http; 443:https	22:sshd; 3306:mysql
网关	192.168.171.1	192.168.171.1	192.168.171.1	192.168.171.1
内存序列号	M378A1K43 DB2-CVF	M378A1K43 DB2-CVF	M378A1K43 DB2-CVF	M378A1K43 DB2-CVF
操作系统版本	Ubuntu 18.04.3 LTS			
CPU型号	Intel(R) Core(TM) i7-10700 CPU @ 2.90 GHz			

表 4　实验2与实验1设备之间的Jaccard′值

Jaccard′	A1	A2	A3	A4
B1	0.96	0	0	0
B2	0	0	0.94	0
B3	0	0.95	0	0
B4	0	0	0	1

表 5　实验3与实验1设备之间的Jaccard′值

Jaccard′	A1	A2	A3	A4
C1	0	0	0	0
C2	0	0	0.96	0
C3	0	0	0	0
C4	0	0	0	1

且不存在任何针对Rowhammer的防护机制，本文在个人电脑上实现的Rowhammer PUF需要进行地址映射并需要绕过相关的软硬件防护机制，因此难度更大。

现有的DRAM PUF都是在嵌入式设备中实现的，无法直接应用于个人电脑，或是可应用于个人电脑但不适用于测绘场景。基于衰减的DRAM PUF[25]需要禁用一段时间的内存刷新，禁用时间过久将导致系统崩溃。为了实现运行时获取PUF响应，程序需要在设备启动时执行，或刷新内存的关键区域以防止崩溃。无法满足不影响被检测设备正常功能的要求，不适用于本文的测绘场景。

基于延时的DRAM PUF[23]通过操纵特定的定时参数使内存处于未定义行为的特殊状态，通过这种方式，可以在系统运行时获得PUF响应，但在运行时修改DRAM的tRCD等定时参数。据我们所知，大多数个人电脑只能在BIOS中修改这些定时参数，这不符合测绘情景的要求。

Sutar等人[26]提出的D-PUF在基于Altera Stratix IV GX FPGA 的 Terasic TR4-230 开发板和DDR3 DRAM内存上实现，Najafi等人[23]提出的Deep PUF在Xilinx XC6SLX45 FPGA和DDR3 DRAM上实现，Talukder等人[27]提出的PreLatPUF在Xilinx Virtex-6 FPGA和DDR3 DRAM上实现。这些工作并未考虑PUF在个人电脑中的使用情景。

7 结束语

本文提出一个完整而详细的测绘实例。首先，本文提出一个基于PUF的测绘框架，其中PUF的作用是生成可以唯一标识目标设备的物理指纹。其次，本文给出了在配备DDR4内存的个人计算机上实现Rowhammer PUF的详细过程，并评估了实现PUF的性能。最后，本文在构建的局域网中进行了测绘实验。结果表明，本文提出的方法能够准确地识别和追踪目标设备，即使目标设备改变了其IP地址、MAC地址，甚至操作系统。

本文基于PUF的测绘框架目前使用Rowhammer PUF来生成物理指纹。当前的Rowhammer技术无法保证在所有的DDR4内存，尤其是具有ECC功能的内存上导致比特翻转。同时也还存在许多保护机制，如TRR机制等被用于检测和防止比特翻转。但与此同时，新的Rowhammer技术也在不断开发，且互联网上部署的大多数设备都不是最新生产的，仍有大量的旧设备受到Rowhammer的影响。因此，本文计划在未来开发适合测绘场景的新的设备指纹生成方法，并在未来进行真实环境下的网络空间测绘。

参 考 文 献

[1] 郭莉, 曹亚男, 苏马婧, 等. 网络空间资源测绘: 概念与技术[J]. 信息安全学报, 2018, 3(4): 1–14. doi: 10.19363/J.cnki.cn10-1380/tn.2018.07.01.
GUO Li, CAO Ya'nan, SU Majing, *et al*. Cyberspace resources surveying and mapping: The concepts and technologies[J]. *Journal of Cyber Security*, 2018, 3(4): 1–14. doi: 10.19363/J.cnki.cn10-1380/tn.2018.07.01.

[2] 陈庆, 李晗, 杜跃进, 等. 网络空间测绘技术的实践与思考[J]. 信息通信技术与政策, 2021, 47(8): 30–38. doi: 10.12267/j.issn.2096-5931.2021.08.005.
CHEN Qing, LI Han, DU Yuejin, *et al*. Practice and thinking of cyberspace surveying and mapping technology[J]. *Information and Communications Technology and Policy*, 2021, 47(8): 30–38. doi: 10.12267/j.issn.2096-5931.2021.08.005.

[3] HOU Yuanwei, CHEN Xiaoxiao, HAO Yongle, *et al*. Survey of cyberspace resources scanning and analyzing[C]. The 14th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS-2020), Lodz, Poland, 2021: 279–291. doi: 10.1007/978-3-030-50399-4_27.

[4] NMAP. Nmap: The network mapper - free security scanner[EB/OL]. https://nmap.org/, 2023.

[5] DURUMERIC Z, WUSTROW E, and HALDERMAN J A. ZMap: Fast internet-wide scanning and its security applications[C]. The 22th USENIX Security Symposium, Washington, USA, 2013: 605–620.

[6] GRAHAM R D. MASSCAN: Mass IP port scanner[EB/OL]. https://github.com/robertdavidgraham/masscan, 2023.

[7] Shodan. Search engine for the internet of everything[EB/OL]. https://www.shodan.io, 2023.

[8] KIM Y, DALY R, KIM J, *et al*. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors[J]. *ACM SIGARCH Computer Architecture News*, 2014, 42(3): 361–372. doi: 10.1145/2678373.2665726.

[9] COJOCAR L, KIM J, PATEL M, *et al*. Are we susceptible to rowhammer? An end-to-end methodology for cloud providers[C]. 2020 IEEE Symposium on Security and Privacy (SP), San Francisco, USA, 2020: 712–728. doi: 10.1109/SP40000.2020.00085.

[10] GRUSS D, MAURICE C, and MANGARD S. Rowhammer.js: A remote software-induced fault attack in javascript[C]. 13th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, San Sebastián, Spain, 2016: 300–321. doi: 10.1007/978-3-319-40667-1_15.

[11] DE RIDDER F, FRIGO P, VANNACCI E, *et al*. SMASH: Synchronized many-sided rowhammer attacks from JavaScript[C/OL]. 30th USENIX Security Symposium, 2021: 1001–1018.

[12] QIAO Rui and SEABORN M. A new approach for rowhammer attacks[C]. 2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), McLean, USA, 2016: 161–166. doi: 10.1109/HST.2016.7495576.

[13] KWONG A, GENKIN D, GRUSS D, *et al*. RAMBleed: Reading bits in memory without accessing them[C]. 2020 IEEE Symposium on Security and Privacy (SP), San Francisco, USA, 2020: 695–711. doi: 10.1109/SP40000.2020.00020.

[14] ZHANG Zhi, CHENG Yueqiang, WANG Minghua, *et al*. SoftTRR: Protect page tables against rowhammer attacks using software-only target row refresh[C]. 2022 USENIX Annual Technical Conference, Carlsbad, USA, 2022: 399–414.

[15] PESSL P, GRUSS D, MAURICE C, *et al*. DRAMA: Exploiting DRAM addressing for cross-CPU attacks[C]. The 25th USENIX Conference on Security Symposium, Austin, USA, 2016: 565–581.

[16] WANG Minghua, ZHANG Zhi, CHENG Yueqiang, *et al*. DRAMDig: A knowledge-assisted tool to uncover DRAM address mapping[C]. 2020 57th ACM/IEEE Design Automation Conference (DAC), San Francisco, USA, 2020: 1–6. doi: 10.1109/DAC18072.2020.9218599.

[17] GRUSS D, LIPP M, SCHWARZ M, *et al*. Another flip in the wall of rowhammer defenses[C]. 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, USA, 2018: 245–261. doi: 10.1109/SP.2018.00031.

[18] SAROIU S, WOLMAN A, and COJOCAR L. The price of secrecy: How hiding internal DRAM topologies hurts rowhammer defenses[C]. 2022 IEEE International Reliability Physics Symposium (IRPS), Dallas, USA, 2022: 2C.3–1–2C.3–6. doi: 10.1109/IRPS48227.2022.9764591.

[19] FRIGO P, VANNACC E, HASSAN H, *et al*. TRRespass: Exploiting the many sides of target row refresh[C]. 2020 IEEE Symposium on Security and Privacy (SP), San Francisco, USA, 2020: 747–762. doi: 10.1109/SP40000.2020.00090.

[20] VAN DER VEEN V, FRATANTONIO Y, LINDORFER M, *et al*. Drammer: Deterministic rowhammer attacks on mobile platforms[C]. 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 2016: 1675–1689. doi: 10.1145/2976749.2978406.

[21] SHARIFFUDDIN S, SIVAMANGAI N M, NAPOLEAN A, *et al*. Review on arbiter physical unclonable function and its implementation in FPGA for IoT security applications[C]. 2022 6th International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, India, 2022: 369–374. doi: 10.1109/ICDCS54290.2022.9780766.

[22] TEHRANIPOOR F, KARIMIAN N, YAN Wei, *et al*. DRAM-based intrinsic physically unclonable functions for system-level security and authentication[J]. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2017, 25(3): 1085–1097. doi: 10.1109/tvlsi.2016.2606658.

[23] NAJAFI F, KAVEH M, MARTÍN D, *et al*. Deep PUF: A highly reliable DRAM PUF-based authentication for IoT networks using deep convolutional neural networks[J]. *Sensors*, 2021, 21(6): 2009. doi: 10.3390/s21062009.

[24] SCHALLER A, XIONG Wenjie, ANAGNOSTOPOULOS N A, *et al*. Intrinsic rowhammer PUFs: Leveraging the rowhammer effect for improved security[C]. 2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Mclean, USA, 2017: 1–7. doi: 10.1109/HST.2017.7951729.

[25] SCHALLER A, XIONG Wemjie, ANAGNOSTOPOULOS N A, *et al*. Decay-based DRAM PUFs in commodity devices[J]. *IEEE Transactions on Dependable and Secure Computing*, 2019, 16(3): 462–475. doi: 10.1109/TDSC.2018.2822298.

[26] SUTAR S, RAHA A, and RAGHUNATHAN V. D-PUF: An intrinsically reconfigurable DRAM PUF for device authentication in embedded systems[C]. The International Conference on Compliers, Architectures, and Sythesis of Embedded Systems, Pittsburgh, USA, 2016: 1–10. doi: 10.1145/2968455.2968519.

[27] TALUKDER B M S B, RAY B, FORTE D, *et al*. PreLatPUF: Exploiting DRAM latency variations for generating robust device signatures[J]. *IEEE Access*, 2019, 7: 81106–81120. doi: 10.1109/ACCESS.2019.2923174.

刘　镝：男，博士生，研究方向为硬件安全、隐私保护技术.
徐闻含：男，硕士生，研究方向为物联网安全、隐私保护技术.
王文东：男，硕士生，研究方向为密码学、网络安全.
李大伟：男，副教授，研究方向为区块链、硬件安全、公钥密码学.
关振宇：男，教授，研究方向为硬件安全、区块链、视频压缩.
刘建伟：男，教授，研究方向为密码学、5G网络安全、移动通信网络安全.

责任编辑：余　蓉