

针对 5G 核心网协议的自动化漏洞挖掘方法

吴佩翔, 张志龙, 陈力波, 王轶骏, 薛质

(上海交通大学网络空间安全学院, 上海 200240)

摘要: 随着 5G 在全球范围内的广泛商用, 5G 网络安全问题广泛受到关注。针对 5G 核心网协议的自动化漏洞挖掘手段主要以黑盒模糊测试为主, 但由于 5G 核心网协议设计复杂, 黑盒模糊测试存在效率低、通用性差、拓展性不足等问题, 不能有效检测到 5G 核心网协议安全漏洞。针对上述问题, 深入了解 5G 核心网架构和重点接口协议的工作原理, 总结在 5G 核心网场景下基于静态分析的自动化漏洞挖掘方法相较传统黑盒模糊测试方法的优势。对 5G 核心网的开源解决方案进行了源代码审计, 发现在 5G 核心网协议实现中一类变量生命周期管理不当而导致的内存泄露安全问题, 可导致 5G 核心网网元被拒绝服务攻击。基于此, 提出了针对此类安全问题的通用漏洞模型, 设计并实现了基于控制流和数据流混合分析的自动化漏洞挖掘方法。通过实验对该方法进行了有效性测试和效率评估, 成功挖掘到了通用 5G 开源解决方案——Open5GS 中的 5 个未公开安全漏洞, 涵盖多个接口协议应用场景, 获得了 4 个 CVE 编号。通过模拟环境, 验证了该系列漏洞的影响范围广且利用条件低, 充分证明了所提自动化漏洞挖掘方法的有效性, 并向相关厂商进行了披露。

关键词: 5G 核心网; 开源解决方案; 协议安全; 静态分析; 漏洞挖掘

中图分类号: TN915.08

文献标志码: A

DOI: 10.11959/j.issn.2096-109x.2024006

Automated vulnerability discovery method for 5G core network protocol

WU Peixiang, ZHANG Zhilong, CHEN Libo, WANG Yijun, XUE Zhi

School of Cyber Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China

Abstract: With the widespread development of fifth-generation (5G) mobile communication technology, concerns regarding 5G network security have also increased. Blackbox fuzzing is a commonly used method for automated vulnerability discovery in software security. However, applying dynamic approaches like fuzzing to discover vulnerabilities in the complex design of 5G core network protocols poses challenges such as low efficiency, poor versatility, and lack of scalability. Therefore, a novel static method to examine the open-source solution of the 5G

收稿日期: 2023-01-06; 修回日期: 2023-04-01

通信作者: 陈力波, bob777@sjtu.edu.cn

基金项目: 国家重点研发计划 (2022QY1702)

Foundation Item: The National Key R&D Program of China (2022QY1702)

引用格式: 吴佩翔, 张志龙, 陈力波, 等. 针对 5G 核心网协议的自动化漏洞挖掘方法[J]. 网络与信息安全学报, 2023, 10(1): 156-168.

Citation Format: WU P X, ZHANG Z L, CHEN L B, et al. Automated vulnerability discovery method for 5G core network protocol[J]. Chinese Journal of Network and Information Security, 2023, 10(1): 156-168.

core network was proposed. Through this method, a series of memory leak security issues caused by improper variable life cycle management were identified, which can lead to denial-of-service attacks on the 5G core network. To summarize these weaknesses, a general vulnerability model and an automated vulnerability discovery method called HoI were presented, which utilized hybrid analysis based on control and data flow. By successfully discovering five zero-day bugs in Open5GS, an open-source solution for the 5G core network, vulnerabilities that cover practical application scenarios of multiple interface protocols in the 5G core network were identified. These vulnerabilities have wide-ranging impact, are highly detrimental, and can be easily exploited. They have been reported to the vendor and assigned four Common Vulnerabilities and Exposures (CVE) numbers, demonstrating the effectiveness of this automated vulnerability discovery method.

Keywords: 5G core network, open-source solution, protocol security, static analysis, vulnerability discovery

0 引言

近年来,以 5G 为代表的新一轮通信技术变革正在快速兴起,其中 5G 核心网全新的架构设计^[1]引入了网络功能虚拟化、服务化网络架构、边缘计算等关键技术,可借助云原生、虚拟化等技术实现按需提供的差异化能力和服务,大幅提升移动网络业务能力。值得注意的是,在全球范围内 5G 广泛商用、规模快速扩大的背景下,5G 网络的安全问题研究成为研究热点^[2]。

5G 网络的核心网完整开源解决方案主要有 free5GC^[3]和 Open5GS^[4]两种,free5GC 主要开发语言为 Go,根据第三代合作伙伴计划(3GPP, 3rd Generation Partnership Project)制定的 R15 标准实现;Open5GS 主要由 C 语言开发实现,根据 3GPP R16 标准实现,该标准于 2020 年冻结,是近几年运营商 5G 核心网商用建设的主流版本^[5]。上述开源解决方案遵循 3GPP 设计规范,能够完整模拟 5G 核心网中的各类业务流程。其中,Open5GS 由于其支持的核心网功能更加丰富,开发迭代更加稳定,是近年 5G 核心网的安全研究工作中的主要研究目标,此外 Open5GS 还支持基于开源版本的商业化二次开发,商业用户众多,包括英国小型移动运营商 Telet Research、无线解决方案提供商 NextEPC 以及诸多大学的无线安全研究实验室等。因此,安全研究人员在研究 5G 网络安全的过程中,可以利用其进行核心网搭建、测试及安全研究,相关研究成果在帮助开源项目快速提升安全性的同时,对 5G 核心网商用解决方案的安全实践具备借鉴和指导意义。

5G 核心网安全的研究内容主要包括服务化

架构(SBA, service based architecture)安全、无线安全、协议安全、身份认证安全、网络切片以及软件定义网络(SDN, software defined network)安全等^[6]。同时,由于 5G 核心网中引入了上述多种新的接口协议,因此,其协议设计和实现的安全性成为 5G 核心网安全的重要研究内容。

目前,产业界和学术界针对 5G 核心网协议安全的研究主要分为协议漏洞模型研究和自动化漏洞挖掘方法研究两类。协议漏洞模型研究主要挖掘 5G 核心网中各网元接口协议的设计、实现中存在的安全问题及相关攻击利用场景;自动化漏洞挖掘方法以黑盒模糊测试研究为主,主要通过黑盒模糊测试挖掘不同接口协议实现过程中存在的内存型安全问题。但是,利用黑盒模糊测试方法针对 5G 核心网的协议实现开展自动化漏洞挖掘存在以下局限:一是该方法的漏洞挖掘效率低,由于 5G 核心网的接口协议设计及实现非常复杂,传统黑盒模糊测试很难实现较高的代码覆盖率,难以发现代码深层漏洞^[7];二是该方法的通用性不足,5G 核心网中诸多接口的协议栈与传统 TCP/IP 栈不同,针对不同的接口协议开展测试需要实现不同的传输层协议与应用层协议适配,由于不同接口协议的业务流程差异很大,很难实现通用的接口协议黑盒模糊测试工具;三是该方法的拓展性差,针对核心网接口协议漏洞模型研究中所发现的共性安全问题,难以快速应用,实现对各网元接口协议的安全测试。

当前针对 5G 核心网接口协议的安全研究中,基于静态代码分析方法开展自动化漏洞挖掘的研究工作较少。与黑盒模糊测试相比,使用静态分析方法对 5G 核心网接口协议进行安全审计有如

下优点：一是静态分析方法通过控制流及数据流分析，能够遍历更多协议执行路径，漏洞挖掘效率更高^[8]；二是静态分析方法能够直接针对各接口协议实现的关键代码进行分析，无须运行整个软件，可以显著降低针对不同接口协议安全测试的工作量，相比黑盒模糊测试方法更加灵活通用；三是针对协议实现中的特定漏洞模型^[9]，静态分析方法能够通过模式匹配快速定位其他接口协议实现代码中可能存在的相似问题，在扩展性上明显优于模糊测试。

针对当前 5G 核心网安全研究现状，本文主要做了以下贡献。

1) 针对 5G 核心网开源解决方案，发现了一类网元接口协议实现中变量生命周期管理不当导致的内存泄露安全问题，攻击者可以利用该类漏洞对 5G 核心网网元发起拒绝服务(DoS, denial of service)攻击，从而造成 5G 核心网关键功能的可用性丧失。

2) 提出了针对此类安全问题的通用漏洞模型，基于控制流分析和数据流混合分析，设计了核

心网协议变量生命周期管理不当的自动化漏洞挖掘方法 HoI (hybrid analysis of improper variable lifecycle management)，并基于开源静态应用程序安全测试框架 CodeQL^[10]进行了实现和优化。

3) 利用自动化漏洞挖掘方法 HoI 成功挖掘 5G 核心网开源解决方案 Open5GS 中 N1、N2、N4 接口协议实现上的 5 个未公开内存泄露漏洞，研究并实现了相关攻击利用方法，已获得 4 个 CVE 编号 (CVE-2022-40890、CVE-2022-43221、CVE-2022-43222、CVE-2022-43223)。

1 相关工作

2016 年年底，3GPP 确定了 5G 核心网架构；如图 1 虚线框所示，5G 核心网总体上分成两个部分：控制面 (core-CP) 和用户面 (core-UP)^[1]。控制面采用服务化架构，用于承载控制信令或消息。用户面也叫数据面或转发面，主要负责承载转发用户数据流量。5G 核心网协议漏洞模型研究主要分为 5G 核心网用户面接口协议、控制面接口协议以及身份认证协议安全研究 3 类。

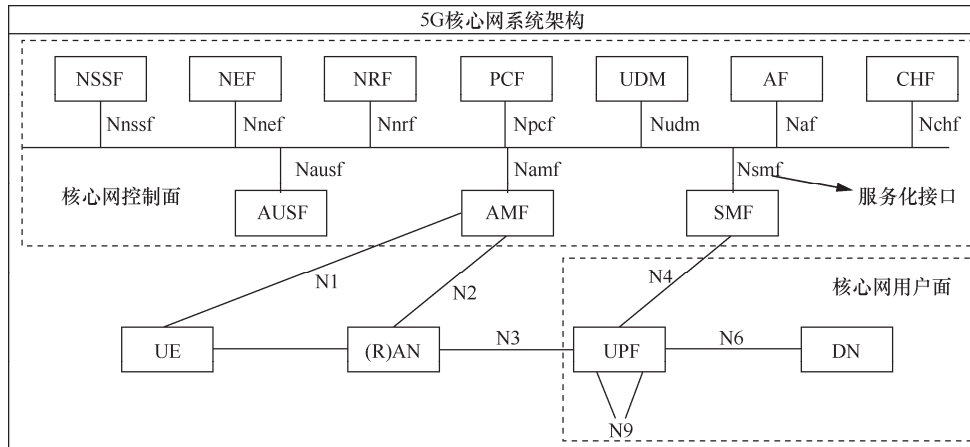


图 1 5G 核心网系统架构及重点网元接口

Figure 1 5G core network system architecture and key network function interface

在 5G 核心网用户面接口协议安全方面，研究报告^[11]中公布了核心网 N4 接口报文转发控制协议 (PFCP, packet forwarding control protocol) 设计中的数个安全漏洞；研究报告^[12]中公布了对 N3 接口通用分组无线服务隧道协议 (GTP, general packet radio service tunneling protocol) 的安全研究成果，研究员认为 GTP 的设计缺陷会导致严重的安全问题，因为无法验证用户的实际位

置，归属网络无法区分来自访客网络中用户的“位置集”信号是否合法，攻击者可以利用此漏洞部署针对运营商设备的拒绝服务攻击、金融欺诈、假冒攻击等。

在 5G 核心网控制面接口协议安全方面，如图 1 所示，5G 核心网中控制面网元以 HTTP/2.0 作为基准通信协议的服务化接口 (SBI, service based interface) 对外提供服务，Hu 等^[13]总结了

表 1 重点网元及接口协议介绍
Table 1 Introduction to Key Network Functions and Interface Protocols

重要网元及接口名称	功能介绍
UE (User Equipment)	5G 用户终端
(R)AN ((Radio) Access Network)	5G 无线接入网
AMF (Access and Mobility Management Function)	接入和移动性管理功能，主要负责终端接入和移动性管理
UPF (User Plane Function)	用户面功能，主要负责数据包的路由转发、与外部数据网络的数据交互等
SMF (Session Management Function)	会话管理功能，主要负责会话管理、策略执行和 QoS 控制部分
N1 接口	UE 和 AMF 间的协议接口，N1 是逻辑概念的接口，应用层协议为 NAS，由 N2 接口协议进行封装承载
N2 接口	(R)AN 和 AMF 间的协议接口，传输层协议为流控制传输协议 (SCTP, stream control transmission protocol)，应用层协议为 NGAP
N3 接口	(R)AN 与 UPF 间的协议接口，主要用于传递 (R)AN 与 UPF 间的上下行用户面数据，(R)AN 与 UPF 之间通过 GTP-U 构建 IP 隧道，承载用户数据
N4 接口	SMF 和 UPF 间的协议接口，主要利用 PFCP 传输 SMF 和 UPF 间的控制面信令

控制面网元 SBI 中可能存在的数种安全漏洞模型；Kqien^[14]提出了漫游场景下 SBI 存在的协议安全问题。

在 5G 核心网身份认证协议安全方面，Borgaonkar 等^[15]发现了在 5G 网络认证及密钥协商协议 (5GAKA, 5G authentication and key agreement protocol) 的一个逻辑漏洞，并给出了一个针对用户位置隐私泄露的攻击利用；文献[16]针对 4G 和 5G 寻呼协议的攻击，展示了位置追踪以及拒绝服务攻击方法；研究发现用户终端在入网认证之前的信息交互没有受到任何保护，且没有经过任何验证，可以被用来进行 5G 设备信息获取、降速等攻击。

在 5G 核心网接口协议自动化漏洞挖掘研究方面，Silveira 等^[17]主要对 N1 接口的非接入层 (NAS, non-access-stratum) 协议以及 N2 接口的下一代应用协议 (NGAP, next-generation application protocol) 的流程进行详细分析，设计并实现开源测试系统 my5G-RANTester，并用上述工具对 Open5GS、free5GC 以及 OpenAirInterface^[18]这个开源项目中 NAS 协议及 NGAP 的实现进行一致性测试及稳定性测试；NCC Group 在研究^[19]中利用自研模糊测试开源工具 Fuzzowski^[20]、开源模糊测试工具 Frizzer^[21]以及 AFL NET^[22]对开源 5G 解决方案 Open5GS 中的 GTP、PFCP、DIAMETER 以及 NGAP 分别进行

了模糊测试，作者自研的 Fuzzowski 工具仅发现了 Open5GS 中 PFCP 实现的一个栈溢出漏洞，该工具在面对更为复杂的 NGAP 和 NAS 协议时效果有限，还需要后续改进；Salazar 等^[23]设计实现了 5Greplay 开源工具，该工具作为无线接入网 ((R)AN, (radio) access network) 与核心网的中间人可以以离线或在线的方式对数据包篡改和重放从而帮助安全人员进行协议安全测试；Hu 等^[24]利用黑盒模糊测试对 NGAP 进行了安全测试，提出了一种基于分区权重表的选择变异算法以提升模糊测试效率；Cao 等^[25]对 PFCP 进行了黑盒模糊测试研究，并提出了一种新的数据变异算法。上述工作均存在通用性差、拓展性不足的问题，不能高效针对 5G 核心网的复杂协议设计进行安全测试，实际漏洞挖掘效果有限。

2 研究动机

5G 核心网架构采用控制面与用户面分离 (CUPS, control and user plane separation) 设计。如图 1 所示，如果将 5G 核心网控制面和用户面看作一个整体，核心网外部攻击面为 N1、N2、N3 接口。与此同时，考虑到实际应用中多接入移动边缘计算 (MEC, multi-access/ mobile edge computing) 场景中，5G 核心网用户面可以下沉至用户侧，核心网的外部攻击面应包含

N4 接口。

本文重点针对上述外部攻击面接口协议开展安全研究,首先,发现了 Open5GS 项目在 N2 接口的 NGAP 实现上存在一个逻辑漏洞(已经申请编号 CVE-2022-40890)。攻击者可以利用该逻辑漏洞导致的内存泄露问题,向 AMF 网元发起 DoS 攻击,使其崩溃。该漏洞原理介绍如下。

在 5G 核心网中,N2 接口是(R)AN 和 AMF 网元的协议接口。如图 2 所示,其传输层协议为 SCTP,应用层协议为 NGAP。当(R)AN 初始接入 5G 核心网时,首先在传输层与 AMF 网元利用 SCTP 建立长连接,之后通过发起 NGAP 中的 NG SETUP 请求尝试与 AMF 网元建立连接。NG SETUP 业务流程如图 3 所示,如果 NG SETUP 请求成功,AMF 网元会向(R)AN 节点发送“NG SETUP RESPONSE”消息;如果失败,则发送“NG SETUP FAILURE”消息^[17]。

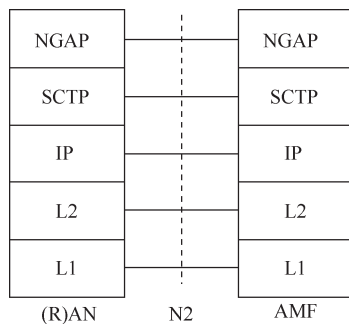


图 2 5G 核心网 N2 接口协议栈
Figure 2 5G core network N2 interface protocol stack

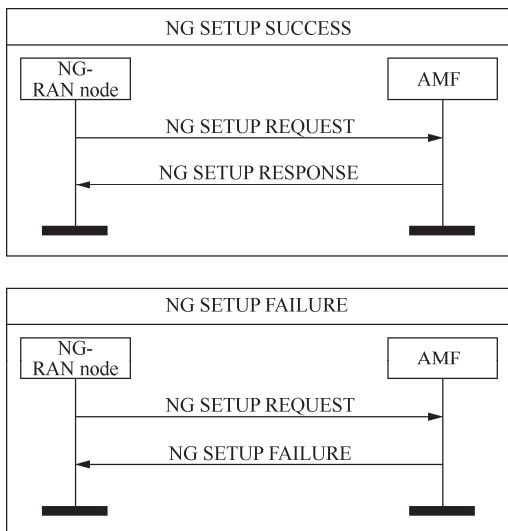


图 3 NG SETUP 业务流程
Figure 3 NG SETUP operation

3GPP 官方文档规定:① (R)AN 与 AMF 之间可以具有多个传输层的 SCTP 长连接^[26];② 当基站侧通过 NGAP 向 AMF 网元注册失败时,如果 NG SETUP FAILURE 的消息报文包括等待时间消息的信息元素(IE, information element)时,则 NG-RAN 节点应至少等待相应的时间,再向同一 AMF 重新发起注册流程^[17]。根据上述标准文档描述可知,当(R)AN 注册失败时,依然可以保留 SCTP 长连接,用于重新发起注册流程。

在 Open5GS 的实现中,AMF 网元中自定义了结构体 amf_gnb,用于管理(R)AN 的上下文信息,在网元初始化的过程中,会创建关于该结构体的内存池 amf_gnb_pool,内存池的默认大小为 128,之后通过该内存池来分配及回收该结构体。

NG SETUP 代码实现示意如图 4 所示,Open5GS 中 AMF 网元对于 NG SETUP 业务流程的处理实现如下。

1) 当(R)AN 与 AMF 网元在传输层的 SCTP 建立长连接成功后,AMF 网元会预先从 amf_gnb_pool 内存池中分配一个 amf_gnb 结构体,并利用该结构体处理(R)AN 之后交互过程中的各项参数及上下文信息。

2) 初始化 NGAP 状态机。

3) 利用 NGAP 状态机实现处理外部实体的 NGAP 请求。

4) 如果 AMF 网元处理 NG SETUP 请求成功,返回 NG SETUP RESPONSE。

5) 如果处理请求失败,将返回 NG SETUP FAILURE 消息,但依然保留了之前分配的 amf_gnb 结构体,并等待(R)AN 重新发送建连请求。该结构体仅在(R)AN 节点主动与其断开 SCTP 长连接时被释放回收。

进一步地,传输层的 SCTP 建立连接不存在身份认证,因此考虑如下场景:未经认证的恶意(R)AN 向 AMF 发起注册接入请求,当 AMF 网元返回注册请求失败时,继续保持 SCTP 长连接,但不通过该连接重新发送建连请求,而是与 AMF 网元建立新的 SCTP 长连接并提交建连请求。如此重复上述流程,AMF 网元就会不断重新分配新的 amf_gnb 结构体,由于其 amf_gnb_pool 内存池有限,在接收超过 128 个 SCTP 的长连接后,该

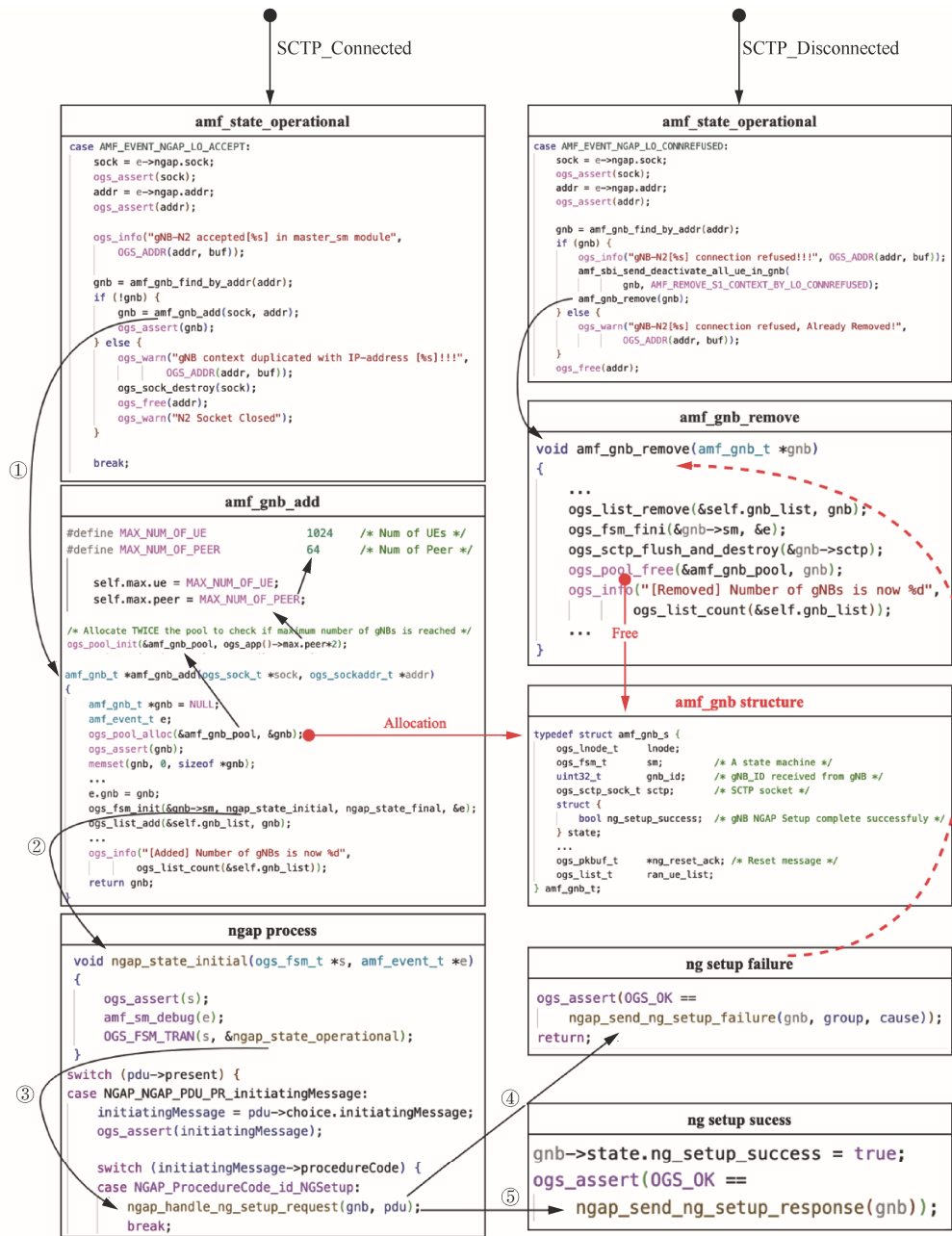


图 4 NG SETUP 代码实现示意
Figure 4 Schematic of NG SETUP code implementation

内存池被占满，无法继续分配 **amf_gnb** 结构体，从而造成 AMF 网元崩溃。该漏洞利用简单且可由未授权的单个 NG-RAN 直接发起攻击，因此发起 DoS 的门槛极低，危害严重。本文基于开源 UE 与(R)AN 解决方案 UERANSIM 实现了漏洞利用攻击脚本，并进行了漏洞验证，如图 5 所示。

分析总结上述漏洞案例，可知由于 5G 核心网中各 Nx 接口的协议设计复杂，核心网网元在

实现过程中自定义了大量的结构体变量来管理外部实体的参数及上下文信息。以 N1 和 N2 接口为例，外部实体变量示意如图 6 所示，**amf_gnb** 结构体包含(R)AN 的各类参数信息；**ran_ue** 结构体包含(R)AN 相关联 UE 的上下文信息；**amf_ue** 结构体包含 UE 终端的各类信息。本文将上述结构体统称为外部实体变量，这类结构体变量彼此之间依赖复杂，定义及其生命周期管理缺乏标准文

```

amf | 09/15 09:45:24.041: [amf] INFO: [Added] Number of gNBs is now 127 (.../src/amf/context.c:880)
amf | 09/15 09:45:24.041: [amf] INFO: gNB-N2[10.10.94.173] max_num_of_ostreams: 10 (.../src/amf/amf-sm.c:698)
amf | 09/15 09:45:24.042: [amf] WARNING: NG-Setup failure: (.../src/amf/ngap-handler.c:305)
amf | 09/15 09:45:24.042: [amf] WARNING: Cannot find Served TAI. Check 'amf.tai' configuration (.../src/amf/ngap-handler.c:306)
amf | 09/15 09:45:25.047: [amf] INFO: gNB-N2 accepted[10.10.94.173]:40959 in ng-path module (.../src/amf/ngap-sctp.c:113)
amf | 09/15 09:45:25.047: [amf] INFO: gNB-N2 accepted[10.10.94.173] in master_sm module (.../src/amf/amf-sm.c:659)
amf | 09/15 09:45:25.047: [amf] INFO: [Added] Number of gNBs is now 128 (.../src/amf/context.c:880)
amf | 09/15 09:45:25.047: [amf] INFO: gNB-N2[10.10.94.173] max_num_of_ostreams: 10 (.../src/amf/amf-sm.c:698)
amf | 09/15 09:45:25.048: [amf] WARNING: NG-Setup failure: (.../src/amf/ngap-handler.c:305)
amf | 09/15 09:45:25.048: [amf] WARNING: Cannot find Served TAI. Check 'amf.tai' configuration (.../src/amf/ngap-handler.c:306)
amf | 09/15 09:45:26.048: [amf] INFO: gNB-N2 accepted[10.10.94.173]:49348 in ng-path module (.../src/amf/ngap-sctp.c:113)
amf | 09/15 09:45:26.049: [amf] INFO: gNB-N2 accepted[10.10.94.173] in master_sm module (.../src/amf/amf-sm.c:659)
amf | 09/15 09:45:26.049: [amf] FATAL: amf_gnb_add: Assertion 'gnb' failed. (.../src/amf/context.c:849)
amf | 09/15 09:45:26.050: [core] FATAL: backtrace() returned 8 addresses (.../lib/core/ogs-abort.c:37)
amf | ./open5gs-amfd(+0xe33f) [0x56149149433f]
amf | ./open5gs-amfd(+0x2f5a5) [0x5614914b55a5]
amf | /open5gs/install/lib/x86_64-linux-gnu/libogscore.so.2(ogs_fsm_dispatch+0x113) [0x7fc97ac56417]
amf | ./open5gs-amfd(+0x8f64) [0x56149148ef64]
amf | /open5gs/install/lib/x86_64-linux-gnu/libogscore.so.2(+0x117e5) [0x7fc97ac477e5]
amf | /lib/x86_64-linux-gnu/libpthread.so.0(+0x8609) [0x7fc97a39e609]
amf | /lib/x86_64-linux-gnu/libc.so.6(clone+0x43) [0x7fc97a2c3133]
amf | /open5gs_init.sh: line 96: 14 Aborted (core dumped) ./open5gs-amfd
amf exited with code 134

```

(R)AN接入失败

amf_gnb结构体达到内存池上限

处理新的接入请求

结构体分配失败
网元奔溃

图 5 CVE-2022-40890 漏洞验证
Figure 5 CVE-2022-40890 vulnerability verification

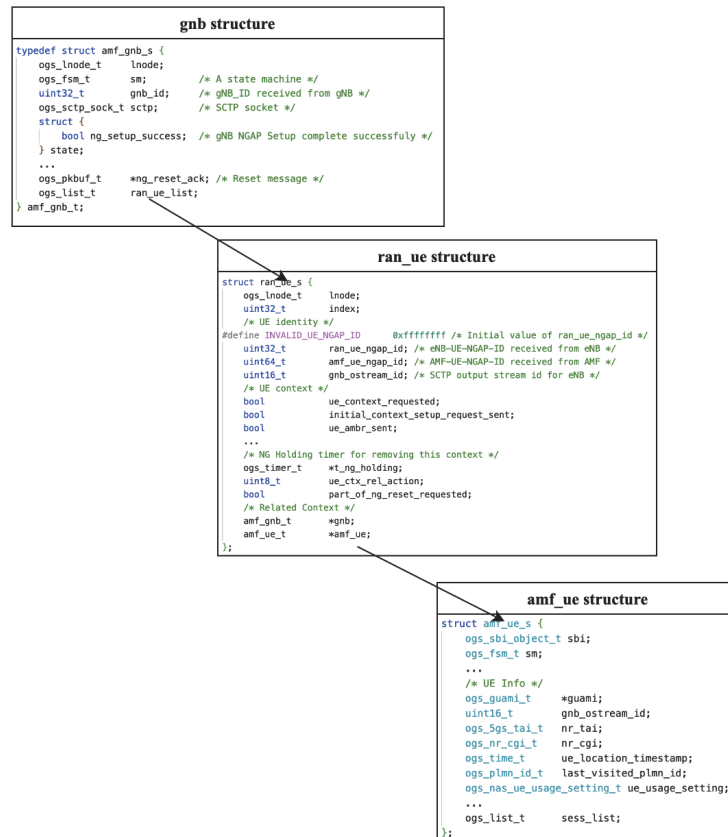


图 6 外部实体变量示意
Figure 6 Schematic diagram of external entity variables

档规范，分配释放时机完全交由开发者根据经验处理，因此一旦处理不当，就很容易产生可被攻击者利用的内存泄露等安全问题。

由前述漏洞案例可知，核心网接口协议的实现中可能会存在外部实体变量生命周期管理不当而导致内存泄露的安全问题。同时，由于核心网中接口协议众多，人工审计耗时较长，为高效对各接口协议实现进行安全测试，本文首先定义了此类安全问题的通用漏洞模型；其次，设计了基于控制流和数据流混合分析的自动化漏洞挖掘方法 HoI，并基于 CodeQL 框架进行了实现优化；最后，根据自动化检测结果，人工验证了核心网及其接口协议实现中是否存在上述安全问题。

3 本文方法设计

自动化漏洞挖掘方法 HoI 设计的主要思路概括如下。

1) 定义漏洞模型，明确哪些变量可以被归类为外部实体变量，以及此类变量在何种条件下存在内存泄露。

2) 设计外部实体变量的自动化发现方法，利用静态分析方法自动化找到各网元中符合本文所定义的外部实体变量。

3) 设计外部数据入口点的自动化发现方法，考虑到 5G 核心网代码实现量大，为提升检测效率，本文希望 HoI 仅对真正涉及外部实体变量的关键代码进行分析，因此下文将根据外部实体变量特点定义外部数据入口点特征，并设计自动化发现方法。

4) 设计基于控制流和数据流混合的自动化检测方法，对 5G 核心网各网元接口协议的实现进行自动化审计，发现其中外部实体变量生命周期管理不当导致的内存泄露安全问题。

3.1 漏洞模型定义

针对 5G 核心网接口协议中涉及的外部实体变量，如果其满足如下条件：① 外部实体变量具备初始内存池，且分配创建存在上限；② 外部实体变量的分配由外部消息输入触发；③ 假设 NG SETUP 的协议建立请求失败，对应的回调函数 `ng_setup_failure` 调用释放函数 `amf_gnb_remove` 释放了 `amf_gnb` 结构体，如图 4 中红色虚线所示，则

上述漏洞将不存在，那么 `amf_gnb` 结构体的分配函数与释放函数之间会存在显式的数据流通路。

由此，本文设定启发式规则如下：设分配函数新创建的外部实体变量为 X ，释放函数的回收参数中外部实体变量为 Y ，如果 X 与 Y 不存在显式关联，则认为该实现存在潜在的内存泄露风险。

3.2 HoI 外部实体变量及其分配释放函数的自动化发现

首先，通过人工审计确定项目框架中统一使用内存池管理函数，如上述示例中的 `ogs_pool_alloc` 以及 `ogs_pool_free`。根据其调用时的参数位置及类型确定外部实体变量的类型和名称。

其次，利用静态分析方法生成程序的控制流图，分别以上述内存池管理函数为起点，逆向遍历程序控制流图，找到内存池分配函数的上层调用函数，并将此类 `wrapper` 函数定义为不同外部实体变量的分配与释放函数。

根据上述方法确定的标准内存池管理函数的上层调用太多，其中很多函数与本文所关注的外部实体变量结构体不相关，并非各自真正的分配释放函数。为了降低自动化发现方法的误报率，采用如下优化思路：通过研究代码实现可知，对于一个真正的外部实体变量的分配释放函数，除了调用框架内的内存函数之外，还会调用框架的数据结构管理函数对外部实体变量进行管理，如 `ogs_list_add`、`ogs_list_remove`，因此以上层调用函数为起点，进行函数分析，如果上层调用函数中既包含内存池管理函数，又包含数据结构管理函数，则认定其为外部实体变量真正的分配释放函数。

3.3 HoI 外部数据入口点的自动化发现

根据前述定义，外部实体变量的创建必须由网元所接收的外部消息所触发，因此外部数据入口点就是此类外部消息处理入口的集合，本文通过人工代码审计发现，在 Open5GS 中，各网元及其 Nx 接口的应用层协议都是以有限状态机的模型进行实现的。如上述示例，AMF 网元会通过 `ogs_fsm_trans` 跳转到有限状态机的实现函数，因此可通过 `ogs_fsm_trans` 宏的调用参数找到各状态机的实现函数，并将其作为外部数据入口点。

由于核心网接口协议的实现中包含大量的异常处理有限状态机，此方法通过 `ogs_fsm_trans`

宏找到有限状态机入口点过多, 会形成误报。因此, HoI 采用如下优化思路: 考虑到外部实体接入时, 有限状态机模型还处于未启动状态, 由此可先根据 `ogs_fsm_init` 宏的调用参数找到各状态机的初始化函数地址, 之后通过状态机初始化函数内 `ogs_fsm_trans` 宏的调用参数找到各状态机的真正实现函数。

3.4 HoI 自动化漏洞模型检测方法设计

根据前述漏洞模型定义, 本文设计了控制流分析与数据流分析混合的自动化静态分析方法。

条件 1 以有限状态机的实现函数为起点, 利用控制流分析, 通过程序控制流图分析其与外部实体抽象变量的分配函数是否存在通路。

条件 2 将分配函数所创建的外部实体抽象变量标记为 Source, 释放函数定义为 Sink 函数, 利用静态污点传播分析, 判断 Source 与 Sink 之间是否存在通路。

如果上述条件 1 存在, 条件 2 不存在, 则认为该外部实体变量存在潜在内存泄露安全问题。

此外, 设有限状态机的入口点数量为 X , 外部实体变量及其分配释放函数数量为 Y , 按照常规实现本文的自动化静态分析脚本需要跑 $X \times Y$ 轮次, 耗时太长。因此采用如下优化思路: 实际上每个网元的有限状态机仅与其网元内的外部实体抽象变量相关, 由于 CodeQL 框架本身提供了

`getLocation/getFile` 等方法来定位函数代码位置, 因此本文依据代码位置, 以网元作为划分, 对有限状态机的入口点与分配释放函数进行自动化分类, 可以将自动化静态分析轮数降低至 $X \times Y = x_1 \times y_1 + x_2 \times y_2 + \dots + x_n \times y_n$, $X = x_1 + x_2 + \dots + x_n$, $Y = y_1 + y_2 + \dots + y_n$ 。

4 方法实现

CodeQL 是 GitHub 官网发布的基于语义的代码分析引擎。通过该工具, 安全研究人员可以利用已知的安全漏洞范式来挖掘类似的漏洞, 目前支持 C/C++、C#、Go、Java、JavaScript、Python、Ruby 等主流编程语言。CodeQL 定义实现了一套名为 QL 的查询语言, 该语言是一种面向对象的高级语言, 其支持库中封装了程序分析所需的所有对象及常用方法。用户理解需要目标代码, 根据潜在安全漏洞的特征, 编写检测规则对应的 QL 查询语句, 从目标代码构建的数据库中查询相关结果。

本文基于 CodeQL 框架通过 QL 语言及 Python 语言实现了上述 HoI: 首先, 配置了 Open5GS 的编译环境, Open5GS 项目使用 meson 开源编译系统及 Ninja 编译器来提升编译速度; 在确定本地可以成功编译后, 使用 CodeQL 框架编译构建目标项目的 CodeQL 数据库。自动化漏洞挖掘方法 HoI 的实现框架如图 7 所示, 分为以下步骤。

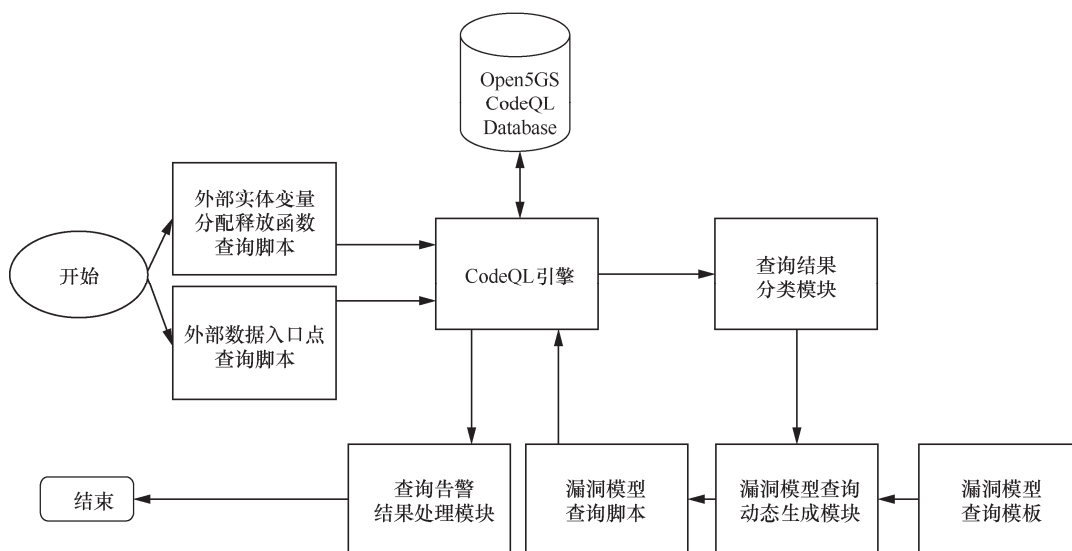


图 7 自动化漏洞挖掘方法 HoI 实现框架

Figure 7 Implementation framework of automatic vulnerability discovery method HoI

1) 使用 QL 语言分别实现了 HoI 中外部实体变量及其分配释放函数发现、外部数据入口点发现的自动化查询脚本。

2) HoI 查询分配模块自动化对步骤 1) 中的输出结果按网元进行分类。

3) 依据漏洞模型编写了 HoI 自动化查询脚本模板。

4) HoI 漏洞模型动态生成模块根据步骤 3) 中的查询模板和步骤 2) 中的输出结果自动化生成最终的自动化漏洞挖掘查询脚本，并调用 CodeQL 框架循环执行查询脚本。

5) HoI 查询告警结果处理模块自动化处理步骤 4) 的查询结果并按网元分类输出存储。

5 实验分析

为了评估本文提出的漏洞模型及自动化分析方法的有效性，需要设计实验回答以下问题。

RQ1: 基于控制流和数据流混合分析的自动化分析方法是否有效？能够挖掘到哪些接口协议实现的未公开漏洞？

RQ2: 自动化漏洞挖掘方法 HoI 的设计实现过程中做了哪些优化，效果如何？

RQ3: 本文研究所挖掘的未公开漏洞产生的原因，以及是否可以利用，具体的漏洞攻击场景是怎样的？

5.1 针对 RQ1 的实验

根据实验方法设计，本文首先使用 QL 查询脚本获得了 Open5GS 各网元的外部入口点以及外部实体变量的内存管理函数的查询结果，并依据网元进行了自动化分类，根据人工复核，Open5GS 各网元外部数据入口点及分配释放函数如表 2 所示；根据漏洞模型查询模板动态生成了各网元的漏洞模型查询脚本，各网元查询轮数及告警数如表 3 所示；由于 5G 核心网内各网元交互流程复杂，本文重点针对核心网外部攻击面接口协议的实现告警进行了人工分析，根据最终实验查询告警结果优先审计了 AMF、SMF、UPF 中 N1、N2、N3、N4 接口的协议实现过程。通过人工复核，成功挖掘了 5 个未公开漏洞，具体数据如表 4 所示，其中，4 个漏洞获得 CVE 编号，另一个已获得厂商确认，在等待 CVE 分配。

表 2 Open5GS 各网元外部数据入口点及分配释放函数
Table 2 External data entry points and memory functions of each network element in Open5GS

网元	外部入口点发现	外部实体变量分配 释放函数对
	发现入口点/ 有效入口点	发现函数对/ 有效函数对
AMF	3/3	4/4
AUSF	1/1	1/1
BSF	1/1	1/1
HSS	1/1	3/3
MME	5/5	10/10
NRF	2/2	0/0
NSSF	1/1	1/1
PCF	3/3	3/3
SCP	1/1	1/1
SGWC	2/2	4/4
SGWU	2/2	1/1
SMF	4/3	7/7
UDM	2/2	1/1
UDR	1/1	0/0
UPF	2/2	1/1
多网元交叉	4/0	22/18

表 3 Open5GS 中各网元查询轮数及告警数
Table 3 Query round numbers and alarm results of each network element in Open5GS

网元	模型运行轮数	告警数/个
AMF	78	15
AUSF	23	3
BSF	23	5
HSS	26	4
MME	160	31
NRF	44	2
NSSF	23	3
PCF	78	7
SCP	23	4
SGWC	52	6
SGWU	46	6
SMF	116	22
UDM	46	5
UDR	22	3
UPF	46	9

表 4 未公开漏洞信息统计
Table 4 Statistics of 0-day vulnerabilities information

5G 核心网网元	外部实体变量	核心网接口协议	危害	漏洞编号
AMF	amf_gnb	NGAP	内存泄露导致 AMF 崩溃	CVE-2022-40890
AMF	ran_ue	NGAP	内存泄露导致 AMF 崩溃	CVE-2022-43223
AMF	amf_ue	NAS	内存泄露导致 AMF 崩溃	已获厂商确认
SMF	pfcf_node	PFCP	内存泄露导致 SMF 崩溃	CVE-2022-43222
UPF	pfcf_node	PFCP	内存泄露导致 UPF 崩溃	CVE-2022-43221

5.2 针对 RQ2 的实验

根据实验方法设计, 本文从 HoI 的外部数据入口点自动化发现、外部实体变量及其分配释放函数自动化发现、漏洞模型查询轮数 3 方面对 HoI 的优化结果进行评估分析。

如图 8 所示, 通过分析 Open5GS 核心网的代码实现方法, 制定启发式优化规则, HoI 在实现过程中采取的优化策略有效减少了上述实验步骤中产生的误报, 根据人工复核, 优化方法所排除的结果中不存在有效的外部数据入口点、外部实体变量及其分配释放函数, 并未造成漏报, 证明了优化方法的有效性和完备性。此外, 根据统计结果可以发现, 对前两项实验步骤进行优化可以有效降低 HoI 查询轮数, 提升自动化静态分析方法的检测效率。

5.3 针对 RQ3 的实验

(1) N2 接口 CVE-2022-43223 漏洞

在 Open5GS 中, 该漏洞涉及的外部实体变量为 ran_ue 结构体, 该结构体初始化内存池大小为 1 024, 当 UE 发起初始化注册请求时, 如果成功接入 5G 核心网, 则核心网中 AMF 网元使用该结构体将 UE 与对应的(R)AN 关联起来, 保存相关上下文信息。

该漏洞发生在 UE 发起注册请求时 NGAP 的处理流程中, 当 UE 发起注册请求时, (R)AN 将 UE 发送的请求信息作为 NGAP 中的 NAS_PDU IE 发送给 AMF 网元, 除此之外, (R)AN 还需在 NGAP 包中加入 RAN_UE_NGAP_ID、User LocationInformation 等 IE。当 AMF 网元接收到该数据包时, 首先利用 NGAP 状态机处理该请求包, 如果通过 RAN_UE_NGAP IE 没有找到对应的 ran_ue 结构体, 则会创建一个新的结构体用于后续流程处理。其次, 对其他 NGAP 的 IE 进行校

验, 如果全部校验处理完成, 则将 NAS_PDU IE 发送给 NAS 协议状态机处理后续流程; 如果处理失败, 则中断处理流程, 并向(R)AN 发送失败原因信息, 但此时 AMF 网元并没有清理该结构体, 该结构体仅在 UE 注册信息验证失败或 UE 主动断开与 5G 核心网的连接时才会被清理。

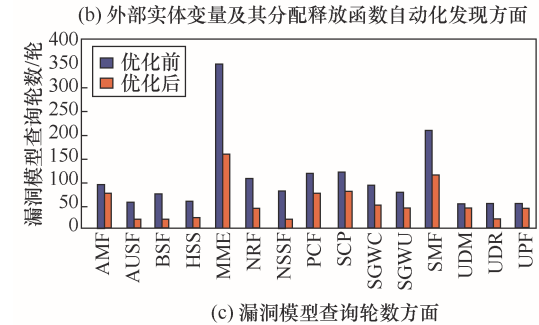
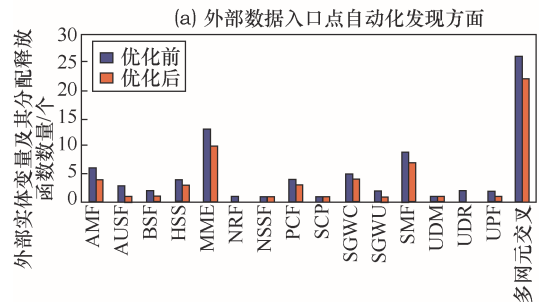
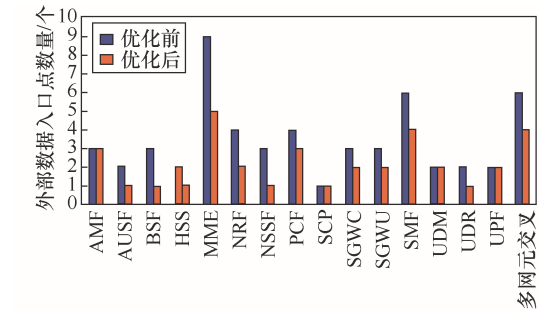


图 8 HoI 的优化结果
Figure 8 HoI optimization result

设想以下攻击场景: 由于 RAN_UE_NGAP_ID 是由(R)AN 自定义生成的, 当恶意(R)AN 成功

接入核心网后，通过 NGAP 向 AMF 网元发送错误的 UE 注册请求数据包，如仅包含不断变更但合法的 RAN_UE_NGAP_ID IE，但不包含 UserLocationInformation、NAS_PDU 等其他 IE，AMF 网元在处理对应请求时会不断创建新的 ran_ue 结构体，但由于后续 IE 校验失败，直接返回错误信息。最终导致 AMF 网元 ran_ue 结构体内存池被占满，无法继续分配新的 ran_ue 结构体而导致崩溃，从而实现对 AMF 网元的拒绝服务 (DoS) 攻击。

(2) N1 接口 NAS 协议漏洞

在 Open5GS 中，该漏洞涉及的外部实体变量为 amf_ue 结构体，该结构体初始化内存池大小为 1 024，当 UE 发起初始化注册请求时，如果成功接入 5G 核心网，则核心网中 AMF 网元使用该结构体保存对应 UE 的上下文信息。

该漏洞发生在 UE 发起注册请求时 NAS 协议的处理流程中，当 UE 发起注册请求时，在 NAS 协议状态机的处理流程中，如果 AMF 网元没有找到对应的 amf_ue 结构体，则会创建该结构体并继续处理后续请求，如果 NAS 协议状态机在解析 NAS_PDU 时发现该请求不合法，会中断后续验证流程，直接丢弃该数据包并返回，但并未清理该结构体，该结构体仅在 UE 注册信息验证失败或 UE 主动断开与 5G 核心网的连接时才会被清理。

设想以下攻击场景：未授权的 UE 不断通过合法(R)AN 向 5G 核心网的 AMF 网元发送错误的注册请求信息，如包含一个非法格式的 NAS 协议数据包，则 AMF 网元会不断创建新的 amf_ue 结构体，最终导致 AMF 网元 amf_ue 结构体内存池被占满，无法继续分配新的 amf_ue 结构体而导致崩溃，从而实现对 AMF 网元的 DoS 攻击。

(3) N4 接口 CVE-2022-43222 漏洞

在 Open5GS 中，该漏洞涉及的外部实体变量为 ogs_pfcpc_node 结构体，该结构体初始化内存池大小为 64，当 UPF 网元通过 N4 接口的 PFCP 向 SMF 网元发送请求时，SMF 网元利用该结构体保存该连接的上下文信息。

该漏洞发生在 PFCP 处理流程中，当 SMF 网元接收到 PFCP 请求时，会通过端 IP 及源端口号查找该结构体是否存在，如果不存在，则认为

该请求是一个新的 PFCP 连接，之后创建该结构体保存对应的上下文信息，并进行后续处理，该结构体仅在 SMF 网元退出时被清理。

由于 PFCP 的传输层协议为用户数据报协议 (UDP)，因此设想如下攻击场景：恶意 UPF 网元不断变更源端口号向 SMF 网元发送错误的 PFCP 数据包，则 AMF 网元会不断创建新的 ogs_pfcpc_node 结构体，最终导致 SMF 网元的该结构体内存池被占满，无法继续分配新的 ran_ue 结构体而导致崩溃，从而实现对 SMF 网元的 DoS 攻击。

上述所有漏洞涵盖了未授权(R)AN 通过 N2 接口攻击核心网 AMF 网元、授权(R)AN 通过 N2 接口攻击核心网 AMF 网元、未授权 UE 通过 N1 接口攻击核心网 AMF 网元、未授权 UPF 通过 N4 接口攻击核心网 SMF 网元等 5G 核心网现实攻击场景，漏洞利用条件低，危害大。上述漏洞发现一方面证明了本文所发现的协议实现安全问题在核心网接口协议设计中普遍存在，通用性强；另一方面证明了本文设计的自动化漏洞挖掘方法 HoI 的有效性。

6 结束语

本文通过对 5G 核心网开源解决方案进行研究，发现了一类在 5G 核心网接口协议实现中，外部实体变量的生命周期管理不当而导致内存泄露的安全问题。针对此类安全问题，本文提出了通用的漏洞模型，并且基于 CodeQL 静态分析框架设计实现了自动化漏洞挖掘方法 HoI。在对 5G 核心网开源解决方案 Open5GS 各网元的接口协议进行自动化安全审计中，HoI 成功挖掘了 5 个接口协议未公开漏洞，实现了相关攻击利用方法用于验证漏洞的有效性，已经向厂商披露，并获得了 4 个 CVE 编号。下一步，一方面将继续完善各类静态分析方法实现；另一方面拟对其他 5G 核心网开源解决方案进行 HoI 的适配，以进一步提升 HoI 安全检测的通用性。

参考文献：

- [1] 3GPP. 5G System architecture for the 5G System (5GS)[S]. ETSI TS 123 501-2018.
- [2] 冯登国, 徐静, 兰晓. 5G 移动通信网络安全研究[J]. 软件学报, 2018, 29(6): 1813-1825.
FENG D G, XU J, LAN X. Study on 5G mobile communication network security[J]. Journal of Software, 2018, 29(6): 1813-1825.

- [3] LIU Y. Evaluation of Free5GC Forwarding performance on private and public clouds[C]//2022 IEEE Cloud Summit. 2022: 9-16.
- [4] JUSTUS R. 5G campus networks: a first measurement study[J]. IEEE Access 2021, (99): 121786-121803.
- [5] 穆佳, 王勇, 马瑞涛, 等. 面向 3GPP R16 的 5G 核心网演进策略研究[J]. 邮电设计技术, 2022(2).
MU J, WANG Y, MA R T, et al. Research on 5G core network evolution strategies for 3GPP R16[J]. Designing Techniques of Posts and Telecommunications, 2022(2).
- [6] TANG Q, ERMIS O, NGUYEN C D, et al. A systematic analysis of 5G networks with a focus on 5G core security[J]. IEEE Access, 2022, 10: 298-319.
- [7] ZHAO B, LI Z, QIN S, et al. StateFuzz: system call-based state-aware linux driver fuzzing[C]//Proceedings of the 31st USENIX Security Symposium (USENIX Security 22). 2022.
- [8] ZHOU S, YANG Z, QIAO D, et al. Ferry: state-aware symbolic execution for exploring state-dependent program paths[C]//Proceedings of the 31st USENIX Security Symposium (USENIX Security 22). 2022.
- [9] CHEN L, WANG Y, CAI Q, et al. Sharing more and checking less: leveraging common input keywords to detect bugs in embedded systems[C]//Proceedings of the 30th USENIX Security Symposium (USENIX Security 21). 2021.
- [10] YOUN D, LEE S, RYU S. Declarative static analysis for multilingual programs using CodeQL[J]. Software: Practice and Experience. 2023, 9(3).
- [11] DOLENTE F, GARROPO R G, PAGANO M. A vulnerability assessment of open-source implementations of fifth-generation core network functions[J]. Future Internet, 2023, 16(1).
- [12] TECHNOLOGIES. P. Threat vector: GTP vulnerabilities in LTE and 5G networks 2020[EB].
- [13] HU X, LIU C, LIU S, et al. Signalling security analysis: is HTTP/2 secure in 5G core network[C]//Proceedings of 2018 10th International Conference on Wireless Communications and Signal Processing (WCSP). 2018.
- [14] KQIEN G M. On threats to the 5G service based architecture[J]. Wireless Personal Communications, 2021, 119(1): 97-116.
- [15] BORGAONKAR R, HIRSCHI L, PARK S, et al. New privacy threat on 3G, 4G, and upcoming 5G AKA protocols[J]. Proceedings on Privacy Enhancing Technologies, 2019, 2019(3): 108-127.
- [16] HUSSAIN S R, ECHEVERRIA M, CHOWDHURY O, et al. Privacy attacks to the 4G and 5G cellular paging protocols using side channel information[C]//Proceedings of Network and Distributed Systems Security (NDSS) Symposium 2019. 2019.
- [17] SILVEIRA L B, DE RESENDE H C, BOTH C B, et al. Tutorial on communication between access networks and the 5G core[J]. Computer Networks, 2022: 109301.
- [18] OPENAIRINTERFACE. An open-source 5G core network solution under development[EB].
- [19] GROUP N. The challenges of fuzzing 5G protocols[EB].
- [20] NCCGROUP. An 5G core network protocol fuzzer[EB].
- [21] DEMANTZ. A coverage-guided blackbox fuzzer based on the Frida instrumentation framework[EB].
- [22] PHAM V-T, BÖHME M, ROYCHOUDHURY A. AFLNet: a grey-box fuzzer for network protocols[C]//Proceedings of 2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST). 2020.
- [23] SALAZAR Z, NGUYEN H N, MALLOULI W, et al. 5Greplay: a 5G network traffic fuzzer - application to attack injection[C]//Proceedings of the 16th International Conference on Availability, Reliability and Security. 2021.
- [24] HU Y, YANG W, CUI B, et al. Fuzzing method based on selection mutation of partition weight table for 5G core network NGAP protocol[M]. Springer Nature Switzerland AG 2022.

- [25] CAO Y, CHEN Y, ZHOU W. Detection of PFCP protocol based on fuzz method[C]//Proceedings of 2022 2nd International Conference on Computer, Control and Robotics (ICCCR). 2022: 207-211.
- [26] JAIN A, LOPEZ-AGUILERA E, DEMIRKOL I. Improved handover signaling for 5G networks[C]//Proceedings of 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC). 2018: 164-170.

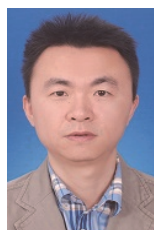
[作者简介]



吴佩翔 (1992-), 男, 新疆伊宁人, 上海交通大学硕士生, 主要研究方向为自动化代码审计、5G 安全、容器安全。



张志龙 (1981-), 男, 江苏南通人, 上海交通大学博士生, 主要研究方向为网络空间安全、计算机外设安全。



陈力波 (1985-), 男, 江西广丰人, 上海交通大学高级工程师, 主要研究方向为软件与系统安全、物联网安全、虚拟化安全。



王轶骏 (1980-), 男, 上海人, 上海交通大学工程师, 主要研究方向为网络与系统安全、网络攻防对抗。



薛质 (1971-), 男, 江苏苏州人, 博士, 上海交通大学教授、博士生导师, 主要研究方向为计算机通信网、网络攻防与评测、安全主动网。