毕业设计（论文）开题报告

边缘计算中服务功能链的数据平面校验问题

一、课题任务及意义

（一）课题任务

定期和指导教师联系，及时汇报毕业设计的进行情况，以便加强毕业设计管理、提高毕业设计质量，具体达到：

1.通过多种文献检索工具（如CNKI、万方、维普、EI、SCI等学校提供的数字图书馆检索工具，以及Baidu、Google等互联网搜索引擎），充分查阅相关文献资料，包括导师老师在任务书中提供的参考文献，其中英文文献数量不低于学校要求；

2.对所学课程高等数学，概率论，程序设计等课程知识的运用；

3.对所收集文献进行国内外现状的研究与分析，以作为后期完成开题报告的基础；

4.参加每周的组内线上学术研讨活动，撰写读书笔记，并每周交互毕设进展；

5.在设计过程中遵守工程职业道德和行业规范，并考虑社会、健康、安全、法律、文化和环境等因素；

6.严格按照重庆大学毕业设计规范文件和规定进度按时完成开题报告、译文和毕业设计论文；

7.搭建的原型系统能够稳定运行；

8.提出的方法、算法和设计方案切实可行，要求明确列出要解决的关键问题，对提出的技术方案（包括算法）有对比分析和优化过程；

9.有结果的充分展示；

该选题综合性较强，工作量较大，同时会较大程度的锻炼学生的综合能力。

（二）课题研究意义

随着5G时代的到来和边缘计算（Edge Computing）的兴起，越来越多的服务供应商开始在边缘网络环境中部署它们的服务，以使得用户能够获得时延保证的计算服务。移动边缘计算（MEC）的出现为传统云用户访问计算密集型或延迟敏感型服务提供了新的解决方案[1]。通过将存储和计算资源部署到更靠近用户的位置，MEC通常可以保证更低的延迟和更高的带宽[2]。网络服务提供商（NSP）开始将其网络功能从云迁移到边缘网络，为其用户提供必要的网络支持。在迁移过程中，他们必须解决以下两个问题：

1.考虑到边缘节点的存储、计算和其他资源容量，将网络功能放置在何处， 2.考虑到接入点（AP）的通信容量，如何调度请求。NSPs传统上使用专用设备（如PE路由器、CDN服务器）在边缘网络上提供此类网络功能，这带来了很多成本，并且网络服务不够灵活。

为了克服这一挑战，越来越多的网络功能（Network Function）以软件的形式被提供（网络功能虚拟化技术，Network Function Virtualization），NFV是指在一个共享的商业硬件平台上用软件实现网络功能，它将传统网络功能与专用设备分离，降低操作和维护成本，并使网络服务更加灵活。在支持NFV的MEC网络中，每个用户请求都必须通过服务功能链（SFC（Service Function Chain）），该服务功能链按特定顺序包含一系列虚拟网络功能（Virtual Network Function）。给定一个SFC，有必要选择适当的MEC节点来映射SFC中的每个VNF，并放置冗余VNF实例以提高可用性预期[6]。对于NSP而言，确保每个SFC的可用性对于提高声誉和扩大市场份额起着至关重要的作用。

在通用服务器上放置相应的网络功能能够有效改善网络服务：传统网络功能高度依赖于专用的网络设备，一方面成本昂贵，另一方面部署和维护的成本很高。以软件形式实现的网络功能通常被组合为服务功能链（Service Function Chain）的形式为用户提供服务，这带来了显而易见可扩展性和灵活性，网络功能也不再依赖于专用网络的设备的部署位置，虚拟网络功能（Virtual Network Function）能够被灵活的组合以提供复杂的网络服务。随着越来越来的虚拟网络功能在边缘网络环境中部署，也随之带来了一些问题。传统的网络服务功能链依赖于网络设备的放置位置，然而虚拟网络功能能够被灵活的放置。显然，需要一种合理的策略来校验用户的流量是否正确的流经预定义的服务功能链；另一方面这些软件形式的实现也不可避免地带来一些漏洞；更有甚者，篡改者会修改我们的数据包。我们需要一种合适的方法去校验服务功能链在数据平面上的正确性，并且能够正常的服务与用户。

本课题考虑在边缘计算中实现服务功能链的数据平面上的正确性校验问题，在上述用户的流量无法有效的被校验的情况下提出一种解决方案，以确保用户流量能够按照其希望的方式流动与处理。

（三）研究现状分析

边缘计算，是指在靠近物或数据源头的一侧，采用网络、计算、存储、应用核心能力为一体的开放平台，就近提供最近端服务。其应用程序在边缘侧发起，产生更快的网络服务响应，满足行业在实时业务、应用智能、安全与隐私保护等方面的基本需求。边缘计算处于物理实体和工业连接之间，或处于物理实体的顶端。而云端计算，仍然可以访问边缘计算的历史数据。

网络功能（NFs），例如中间体，已经被广泛部署在企业网络和电信服务提供商(TSP)网络上。为了简化NF的部署和管理，提出了网络功能虚拟化(Network Function Virtualization, NFV)，将NFs从私有物理盒中分离出来，在商用硬件上以虚拟化软件的形式运行。随着这一趋势的发展，企业可以将内部NFs外包给第三方基础设施，如公共云或TSPs的中心办公室，也有可能部署在边缘计算应用中。这为企业带来便利的同时也暴露了一些问题，企业完全屈服在部署和管理NF的第三方服务提供者的控制下,并且经常性的不能保证他们在云端的流量是否按照他们本身的想法被正确处理了。这件事情已经存在着一些研究。vNFO通过重播在本地环境经过检验的日志检查包转发的正确性，因此不能实时验证外包的SFCs。同时，它需要修改每个NF的内部函数来记录数据包的轨迹，这让NF的开发变得复杂。另外，传统IP网络中已有的路径验证方法无法用于SFC的验证，原因有二。第一，它们需要预先了解包转发路径，并假定路径是静态的。然而，在有状态NFs存在时，这个假设就不适用了，在有状态NFs中，数据包是根据不可预知的NF状态动态调度的。其次，它们利用在不变包字段上计算的嵌套消息认证码(MACs)来实现路径验证，这不适用于具有不可预测的包变化的SFC验证。为了让用户能够验证数据是否按照自己希望的形式被处理，Wu团队提出了SFC[1],一种允许一个企业精确的验证他们SFC执行的正确性的实时SFC验证机制。特别的是，他可以检测大范围内的SFC的违规，这包括：转发路径不符合、丢包、流攻击。同时，它的敏捷性和通用性让它可以在不需要修改NFs的情况下应用于任意云架构。vSFC是一种通用的、敏捷的、实时的SFC验证方案，该方案显示了良好的基础设施兼容性，并且不修改NF软件。vSFC设计为一种轻量级的逐跳验证方案，使用包标签和流确认消息来检测相关的SFC违规，并采用了一种自适应采样方法，在不牺牲检测精度的情况下，有效地降低了检测开销。vSFC在开销可以忽略不计的情况下实现了较高的检测精度。

在SFC验证方面，我们需要考虑的是如何验证已经部署的SFC按照我们的预期正常的工作，也即是否用户的流量将会流经正确的NFs序列，并能够检查出流在SFC上处理的错误，如流经NFs时乱序、丢包、重复和恶意非法者的篡改等。在这一方面，Liu等人提出了AuditBox[2]，其主要目的是使NFV与审计兼容，并提供了比传统审计程序更强大的保证。在传统的审计中，管理员定期测试系统的正确性，例如，每月一次。相反，AuditBox持续自我监控正确的行为，证明运行时确保系统保持与策略目标的一致性。此外，AuditBox通过提供采样日志，仍然允许审计员手动检查系统行为，从而与传统的审计实践保持兼容。AuditBox会持续自我监控正确的行为，同时提供采样日志保持与传统审计实践的兼容性。AuditBox通过将可信执行环境与轻量级验证路由协议（VRP）相结合来实现其目标。在另一篇工作中，Zhang等人提出 vSFC[18]，以允许企业实时准确地验证SFC实施的正确性。具体而言，它可以检测范围广泛的SFC违规行为，包括转发路径不一致、数据包丢弃和流丢弃攻击。同时，它具有通用性和敏捷性，可以应用于任意云架构，而无需对NFs进行任何修改。他们在基于Linux内核的虚拟机（KVM）上实现了一个vSFC原型，并对实际流量进行了大量实验。

然而，从流量传输角度来看，传输粒度多为包（packet）或者流（flow）, 而以上两种验证方案并未扩展近些年数据中心出于控制负载均衡而提出的各种新颖的流片（flowlet或flowcell）方案。流片级别的负载均衡路由方案通常在端主机上对流进行分割，将长流分割为多个流片。根据流片的大小是否固定又可分为flowlet和flowcell。一个flowlet通常是一组突发的数据包，即两个flowlet之间的时间间隔通常足够大能在一定程度上避免包乱序。一个flowcell则通常是固定大小的流片。基于流片级别的负载均衡能缓解流级别方案的大流碰撞问题，且对传输层协议较友好，不会产生大量的包乱序。因此，本课题设想基于以上发现，设计出能够适配粒度更小的流片的服务功能链的数据平面校验策略。

二、重点研究内容及技术实现途径

（一）重点研究内容

本课题的重点研究内容是设计一种对边缘计算服务功能链的验证策略。为了对服务功能链内的数据或流量进行验证其正确性，我们需要提供一种工具去验证SFC能够正确地按照预期的意图工作。此时，对可能出现的NFs乱序，丢包、重复和恶意的非法者的攻击需要能够有判断的依据，以及能够正确的判断NFs乱序，丢包、重复和恶意的非法者的攻击之间的区别。在这方面Wu团队开发出的vSFC[1]中已经有了一些相关的工作，我可以对其进行参考并加以改进。

从流量传输角度来看，以上参考文献内研究的的SFC验证粒度多为包（packet）或者流（flow）, 现有工作未能覆盖近些年数据中心出于控制负载均衡而提出的各种新颖的流片（flowlet或flowcell）方案。流片级别的负载均衡路由方案通常在端主机上对流进行分割，将长流分割为多个流片。根据流片的大小是否固定又可分为flowlet和flowcell。一个flowlet通常是一组突发的数据包，即两个flowlet之间的时间间隔通常足够大能在一定程度上避免包乱序。一个flowcell则通常是固定大小的流片。基于流片级别的负载均衡能缓解流级别方案的大流碰撞问题，且对传输层协议较友好，不会产生大量的包乱序。但是其不能充分利用所有可行路径，较难实现最优的性能，且当网络流量突发严重时，也会导致严重的报文乱序。例如，flare[19]是基于flowlet的分布式负载均衡路由方案。其指出基于包的路由方案会导致大量的包乱序，TCP会将其判为拥塞，进而导致性能下降。基于流的拆分策略将每条流固定到一个特定的路径，从而避免了包乱序。但是流的大小和速率差异很大，一旦指定了路径，就不可更换。因此，基于流的分割可能会将流量分配给并非最优的路径，或无法在流对网络的需求变化时快速重新平衡负载。Flare 基于此观察首次提出了将传输层TCP流分割成多个流簇（flowlet，一组突发报文集合），再将不同的流簇路由到不同路径。flare设定两个flowlet之间的间隔是50ms，即对于时间间隔小于50ms的包，将被划分至同一个flowlet中。flare要求交换机维护流的状态并记录到达目的主机的时延，从而为每个flowlet选择最优路径。

从SFC的校验算法方面来看，在SFC中使用传统的IP网络中的路径验证方法是不可行的。在以前的方法中，一个资源节点和一个目的节点提前协商转发的路径。在收到包时，每个路由器通过在不变数据包字段和前面的数据包标记之上计算MACs来产生包的标签。之后，目的节点通过之前协商好的转发路径和特定的包字段来重新计算嵌套的MACs。这些值与包的标签对比来验证包是否按照期望的路径进行转发。然而，由于以下原因，这些方法不能应用于验证SFC。

（1）不可预测的包转发路径：不同于无状态的数据平面，L2/L3网络设备(交换机，路由器)基于静态的规则进行包的转发，NFs可能有保持状态和上下文相关的转发行为，到这包转发的路径是不可预测的。比如，一个有状态的防火墙基于流的状态转发包，并且IPS接受负载中不包含恶意签名的报文。这些信息持续的在变化，很难被捕获。因此，之前用于验证静态的包转发路径的方法不适合用来检查动态的SFCs的执行情况。

（2）包变化的不可预测：路由器和交换机的不同点仅仅在于在转发包的时候不修改他们，NFs可能会检查并且修改不同的包字段。例如，NAT改写报文的源IP地址，NF处理IPsec报文修改报文的载荷。因此，当包通过SFCs转发时，由于SFC中的包变化不可预测，现有的解决方案无法轻松识别其不变的包字段来计算用于路径验证的嵌套MACs。

一个正确的SFC需要满足包括包/报文的正确性：没有被拦截，没有被修改；流的正确性，没有被丢弃，没有被重新排序，没有重复，没有包的注入（Packet injection）；转发路径方面，没有NF跳过（NF skipping），没有无序的报文/数据包遍历（Out-of-order packet traversal），没有包转发困惑（Forwarding confusion），没有预期外的NF附加在路径中（Unexpected NF appended），即SFC被正确实现，且报文按照NFs协议的顺序正确遍历。在Wu设计的vSFc[1]中，他们总结了可能会使得SFC存在以上错误的原因：

1.违反转发路径: 恶意的或配置错误的路由器可能会沿着意想不到的SFC路径而不是企业预先指定的路径转发数据包。相关转发违规情况包括:

2.NF跳过：NF实例可能会因为负载或是物理机器的错误而崩溃。为了隐藏NF的错误，一个懒惰的服务商可能会将包重定向并跳过这些NFs。

3.无序的报文/数据包遍历：由于交换机配置错误或恶意攻击，可能导致NF实例连接顺序错误。

4.包转发困惑：配置错误的交换机可能会维持意外的转发决策。

5.预期外的NF附加在路径中：恶意服务提供者可能会在路径上附加意外的NF实例。这些额外的NF实例可以修改流量，例如，丢包(即丢包攻击)，基于包过滤规则丢流(即丢包攻击)，以及操纵包头或有效负载。这里不需要考虑流量隐私侵犯，因为它们可以通过包加密来处理。

6.包注入：一个恶意的路由器可能会通过发送虚假的包或者在SFC的路径上重新发包来破坏SFCs。

7.懦夫攻击（Coward Attack）: 攻击者可以通过挂装Coward攻击来逃避异常检测和SFC验证。例如，它可能故意丢弃不用于SFC验证的流。

我会根据以上特点提出能够判断出SFC异常的校验方法，研究在边缘侧实现SFC验证方案，并经过Python的OpenDaylight 库仿真验证，最终采用开源的NF实现（NetBricks, OSDI’16）+容器（替代虚拟机更加轻量级的实现）+现有的负载平衡路由方案(flare等)+修改过的验证方案（以流片为单位的验证），并尝试是否能够将其扩展为适应流片的验证方案或是提升其性能。

（二）技术实现途径

实验室内软硬件设施完善，拥有数套自研及合作研发的大规模系统平台，如网络空间安全（物理安全、系统安全、公共安全）大数据分析管理体系与平台、基于RFID/NB-IoT的大规模物联网研发实验测试环境，以及开放物联网平台，可为实验的开展和后续测试提供良好的实验环境。本人具有较好的代码能力和文献阅读能力，并已经开始阅读相关文献。本实验需要多台物理设备，但也可以使用虚拟机的方式实现，在实验上具有可行性。

本课题预期采用Python的OpenDaylight 库来代替SFC，对算法进行仿真，并采用开源的NF实现（NetBricks, OSDI’16）+容器（替代虚拟机更加轻量级的实现）+现有的负载平衡路由方案(flare等)+修改过的验证方案（以流片为单位的验证）。

三、课题预期成果

（一）课题预期成果

本课题在完成之后预期实现完成一个边缘计算中服务功能链的数据平面校验的策略，能够验证SFC是否正确地按照预期的意图工作。对可能出现的NFs乱序，丢包、重复和恶意的非法者的攻击能够正确的判断当前发生的状况属于哪一类，且该策略能够适应近些年数据中心出于控制负载均衡而提出的各种新颖的流片（flowlet或flowcell）方案。

（二）课题预期特色

1. 能够判断出SFC内发生的异常情况属于NFs乱序，丢包、重复还是恶意的非法者的攻击

2. 能够适应近些年数据中心出于控制负载均衡而提出的各种新颖的流片（flowlet或flowcell）方案

|  |  |  |
| --- | --- | --- |
| 四、进度计划 | | |
| 序号 | 起止周次 | 工 作 内 容 |
| 1 | 1周至 3 周 | 完成开题和算法熟悉 |
| 2 | 4周至 8 周 | 完成算法实现和系统设计 |
|  |  | 完成边缘计算中服务功能链的验证实验 |
| 3 | 9周至 11周 | 对实验结果进行分析，对实验进行改进 |
| 4 | 12周至 13 周 | 完成毕业设计，撰写并完善论文 |

参考文献：

[1] vSFC: Generic and Agile Verification of Service Function Chains in the Cloud.

[2] Don't Yank My Chain: Auditable NF Service Chaining.

[3] A comprehensive survey of Network Function Virtualization.

[4] Differential Network Analysis.

[5] Programming Network Stack for Middleboxes with Rubik.

[6] Jingyuan Fan, Meiling Jiang, and Chunming Qiao. Carrier-grade availability-aware mapping of service function chains with on-site backups.

[7] A Modular Compiler for Network Programming Languages.

[8] Bento: safely bringing network function virtualization to Tor.

[9] 尹星,朱轶,王良民.虚拟网络新技术 /编著

[10] 李素游,寿国础.网络功能虚拟化：NFV 架构、开发、测试及应用 /

[11] 5G 时代边缘计算：LF Edge 生态与 EdgeGallery 技术详解

[12] 海德希尔·乔杜里.精通 OpenStack

[13] 祖家琛,胡谷雨,严佳洁,李实吉.网络功能虚拟化下服务功能链的资源管理研究综述[J].计算机研究与发展,2021,58(01):137-152.

[14] 阳勇,孟相如,康巧燕,赵文文.基于流量优化的可靠服务功能链部署方法[J].系统工程与电子技术,2021,43(10):3017-3025.

[15] 裴家宁.(2020).SDN/NFV网络中服务功能链的路由和资源优化研究.

[16]X. Zhang, Q. Li, J. Wu, and J. Yang, “Generic and agile service function chain verification on cloud,” in Proc. IEEE/ACM 25th Int. Symp. Qual. Service (IWQoS), Jun. 2017, pp. 1–10.

[17] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, “Making middleboxes someone else’s problem: Network processing as a cloud service,” ACM SIGCOMM Comput. Commun. Rev., vol. 42, no. 4, pp. 13–24, 2012.

[18] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, “Network function virtualization: State-of-the-art and research challenges,” IEEE Commun. Surveys Tuts., vol. 18, no. 1, pp. 236–262, 1st Quart., 2016

[19] Fang Liu, Guoming Tang, Youhuizi Li, Zhiping Cai, Xingzhou Zhang, and Tongqing Zhou. A survey on edge computing systems and tools.

[20] APKeep: Realtime Verification for Real Networks.

学生签名：

2022年 1 月 13 日

五、指导教师意见

指导教师签名：

校外指导教师签名：

年 月 日

说明：

1．开题报告应根据教师下发的毕业设计（论文）任务书，在教师的指导下由学生独立撰写。

2．本页不够，请加页。