一种基于NFV的过载控制框架

一、讲了什么

网络功能虚拟化(NFV)的目标是通过云计算和虚拟化技术提供高性能的网络服务。然而，网络过载是一个主要的挑战。传统的弹性云计算可以通过按需伸缩来解决部分过载问题，但却无法满足专业级电信服务的高可用性需求。因此这篇论文提出了一个新的过载控制框架来过滤进入的流量来保证NFV服务不会因为短期内的流量过多而爆炸并尽可能的利用可用的能力。此外这个框架被设计成可适用于NFV的各种服务模型。论文作者对nfv导向的Clearwater IMS进行了广泛的实验评估，表明该解决方案是健壮的，并且能够以非常小的性能开销维持严重的超载条件。

二、基本介绍

作者先简单介绍了为什么会使用NFV技术（减少开销和上市时间等，简单地说，为了工业化），然后讲云弹性不足以满足工业化高可用性需求，因为种种原因，NFV需要额外的解决方案，通过拒绝或放弃对网络容量的过度流量以减少在短期内的过载。接着讲了一些这种解决方案还要考虑的东西，最终得出结论，过载控制不应该依赖于VNF软件的合作。最后引出本文作者提出一个新颖的过载控制框架NFV(NFV-Throttle)，可以保护NFV服务在很短的时间内过载，通过调优流入VNFs的流量，以最大限度地利用可用容量，并保持网络服务承认的流量服务质量。为了达到这个目的，框架提供了一组模块代理来检测和删除多余的流量，可以在VNF中安装，也可以在NFVI级安装，而不需要对VNF软件进行更改。此外，框架还提供了通用的设计指南和参考实现，让NFV的设计者引入用户定义的启发式来控制流量下降/拒绝率。作者还对一种叫clearwater的NFV导向的IMS做了大量的试验，它的设计目的是支持大规模的横向可伸缩性，并采用流行的云计算设计模式和技术，包括OpenStack/KVM虚拟化堆栈。作者对VNFaaS和NFVIaaS用例进行了实验。实验结果表明:

- 在这两种情况下，解决方案都能够及时地保护VNF网络，使其免受严重超载条件的影响，通过维持一个高的系统吞吐量，甚至达到工程能力的1000%。
- 超载控制解决方案有一个小的性能成本：在非过载条件下，CPU开销最多为1.5%，在最严重的过载条件下(1000%)小于4%；内存开销很少，只有几MBs，并且是常量。
- 在基础设施层面的超载控制可以达到最佳的性能；但是，VNF级的过载控制也可以实现类似的性能，并且对于无法控制底层基础结构的VNF提供者来说是一个合适的解决方案。此外，网络级的超载控制对于通知用户在网络内的过载情况是有用的，以便让他们逐渐减少工作负载。
- 即使Clearwater IMS已经嵌入了过载控制机制，它仍然可以在严重超载条件下体验软件故障和低性能。建议的解决方案能够防止这些故障，并优于这些超载控制机制。

本文的结构如下。第二节讨论了NFV的过载控制问题、提出的解决方案的目标以及云计算和计算机网络过载控制的相关工作。第三节提出了解决方案的设计。第四节给出了该方案的实验评价结果。第五节总结全文。

三、前置问题

3.1NFV的过载控制问题

过载会导致已经建立的连接的中断和高优先级服务的不可用，更糟糕的是，过载会直接导致VNF软件崩溃（会导致连锁反应）。

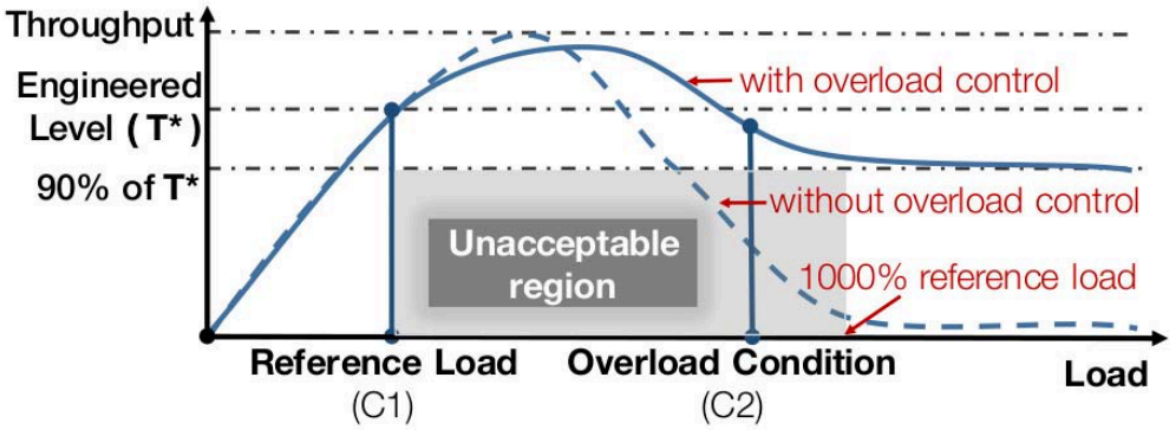


Fig. 1. Network throughput under overload conditions.

有没有过载控制的吞吐量的区别。根据这一观点，NFV过载控制解决方案应考虑以下要求:

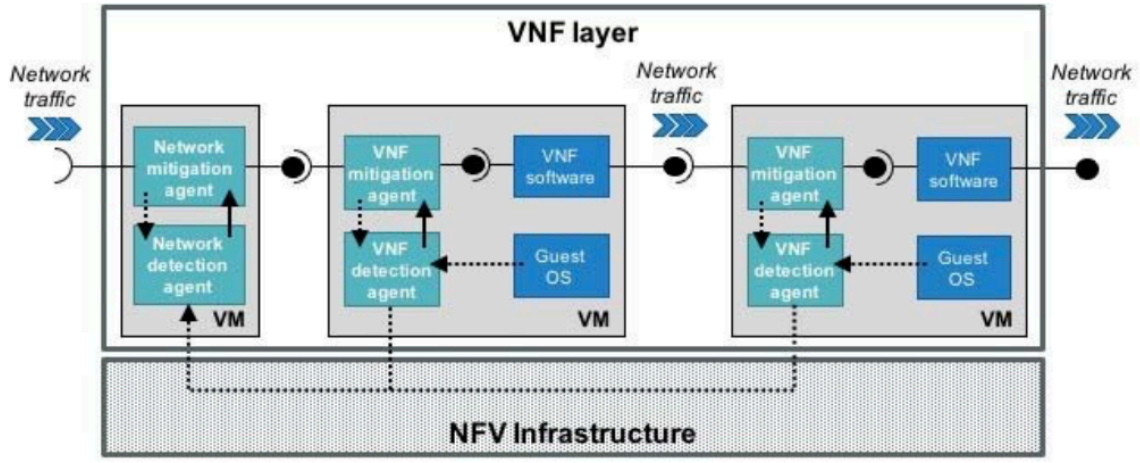
- 在严重超载条件下，NFV网络应该达到可接受的服务水平(例如，不低于其工程吞吐量的90%)(比如参考负载的10倍)。
- 过载控制解决方案应迅速对过载情况作出反应。
- 重载控制解决方案应该与NFV的用例和场景集成。
- 超载控制解决方案应该引入最小的开销，并且不能在正常负载条件下降低服务质量(例如，当处理资源可用时，它不应该过滤流量)。

3.2相关工作

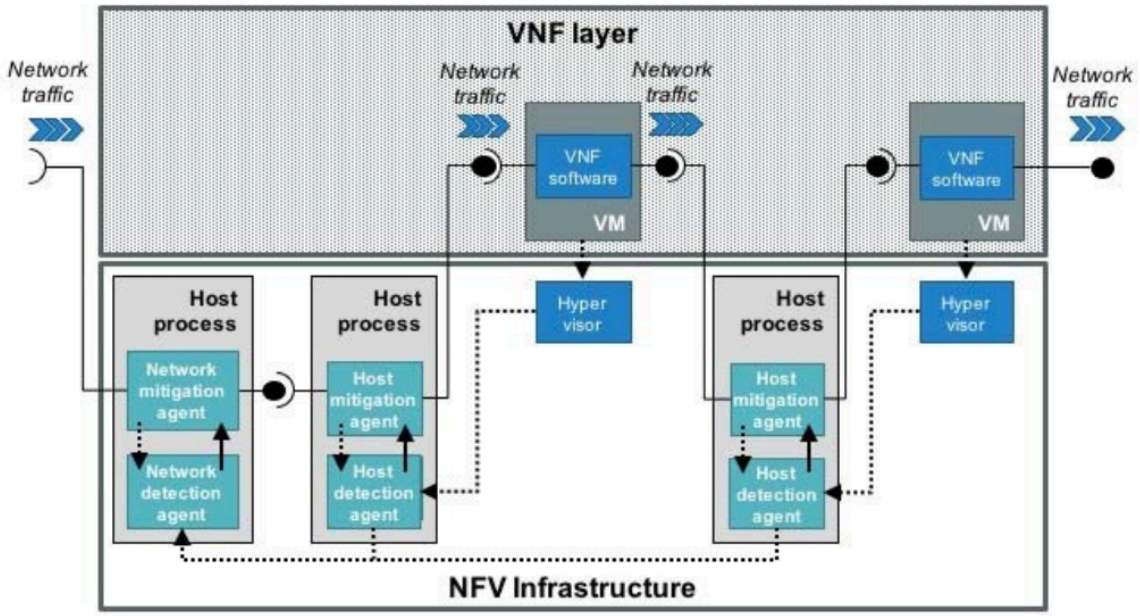
现有的大多数方法都集中于防止过载，即提前扩展系统以避免瓶颈，并为预期的工作负载预留资源。但是，这些解决方案是为了长期的容量规划，而不是为了解决突发的、意外的过载情况。云弹性需要很长时间对过载做出反映，不满足NFV。新的流量控制框架新增了一个防御层满足以上情况，其中组件可以由不同的参与者(例如NFVIaaS和VNFaaS提供者)部署。

四、建议的过载控制解决方案

提出的解决方案是基于一组过载检测代理和过载缓解代理的重载控制框架。这些代理是在NFV网络中部署的软件模块，对VNF软件是透明的(图2)。过载检测和缓解方案进一步分为三种类型。vnf级代理保护单个VNFs不受其vm内部的影响，并通过减少流量来应对过载情况。主机级代理也通过减少流量来保护单个的VNFs，但是它们运行在vm之外。最后，网络级代理保护NFV网络免受影响整个网络的过载条件，并通过拒绝通信和通知客户过载情况来作出反应。



(a) Deployment managed by the VNF provider.



(b) Deployment managed by the NFVI provider.

Fig. 2. Overview of the overload control solution.

该框架不需要改变VNF软件和虚拟化软件，并且可以通过第三方VNF软件和虚拟化技术透明地安装到NFV网络中。很大一部分在说该框架不改变VNF的基本结构，而是对云弹性的一种补充。且用户可以选择之安装一部分。

4.1VNF级设计

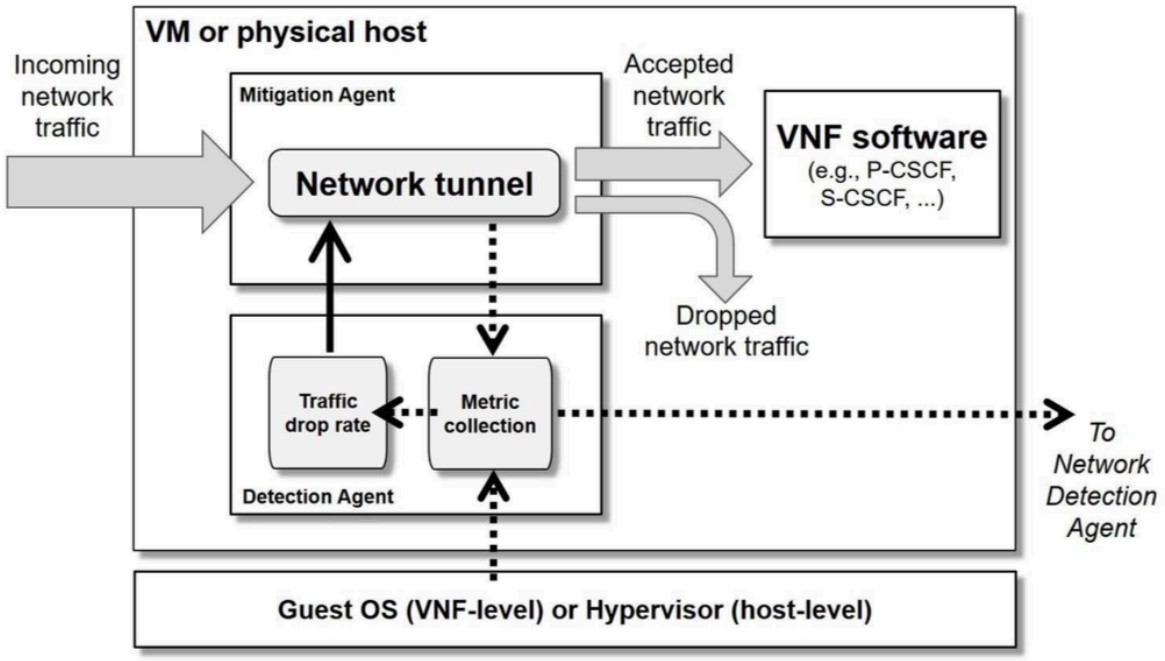


Fig. 3. Architecture of VNF- and host-level overload control.

图3显示了VNF级过载控制方案的体系结构，其中包括一个检测代理和一个缓解代理。VNF级检测代理是在VM中由VNF提供程序部署的组件，以解决单个VNF的过载(图2a)。它通过使用Guest OS公开的接口(如Linux操作系统的procfs虚拟文件系统)，从VM收集资源利用指标。具体来说，它收集关于VM使用虚拟CPU的指标（CPU节拍、空闲CPU时间消耗等）。此外，VNF级检测代理从VNF级缓解代理中收集VNF流量的吞吐量，以便将CPU利用率与传入网络流量的数量关联起来。VNF级检测代理的输出由流量下降率表示，即，输入VNF流量的百分比不被接受。一般来说，如果工作负载在VNF的容量范围内(例如CPU利用率低于限制)，那么流量下降率应该为null。如果CPU利用率接近饱和，则代理必须降低输入流量的一部分，以减少VNF软件的负载。代理应该只允许足够的流量，以保持CPU的充分利用率，但不要超载。VNF级缓解代理在NFV网络中充当VNF软件和其他VNFs之间的网络通道。这个转发是通过使用由Guest OS提供的网络流量转发机制完成的。

$$\text{capacity} = \frac{\text{MEAN}[\text{accepted\_traffic}[1 \ldots N]]}{\frac{\text{MAX}[\text{cpu\_usage}[1 \cdots N]]}{\text{reference\_cpu\_usage}}} \qquad (1)$$

试验通过定义一组启发式来控制下降率，实现了框架的参考版本。代理使用简单而健壮的规则更新drop_rate：如果CPU利用率低于参考值，则允许所有传入的流量；相反，如果CPU利用率超过了引用，则允许的流量按比例伸缩到引用和实际CPU利用率之间的差距。其中，cpu_usage是最新的N个繁忙的虚拟CPU时间百分比的滑动窗口(最高为100%，严格大于0)；accepted_traffic是由代理实际传递给VNF软件的流量。

当输入到vnf级缓解剂的流量时(即，incoming_traffic)超过容量，我们评估流量的下降比例为（在incoming_traffic[N]>能力的情况下，不然drop_rate = 0）：

$$\text{drop\_rate} = 100 \cdot \left(1 - \frac{\text{capacity}}{\text{incoming\_traffic}[N]}\right) \qquad (2)$$

4.2主机级设计

主机级别的检测代理是一个多线程应用程序，该代理将替换VNF级别的检测代理，以便在NFVI提供者的部署时保护VNF不受流量的影响。主机级别的检测代理在NFV网络中监视一个或多个VNFs。在相同的云基础上部署多个主机级别的检测代理是可行的，每个主机级检测代理都在NFV网络中监视VNFs的一个子集。主机级代理的架构与VNF级的架构几乎完全一样，算法也基本相似，

4.3网络级设计

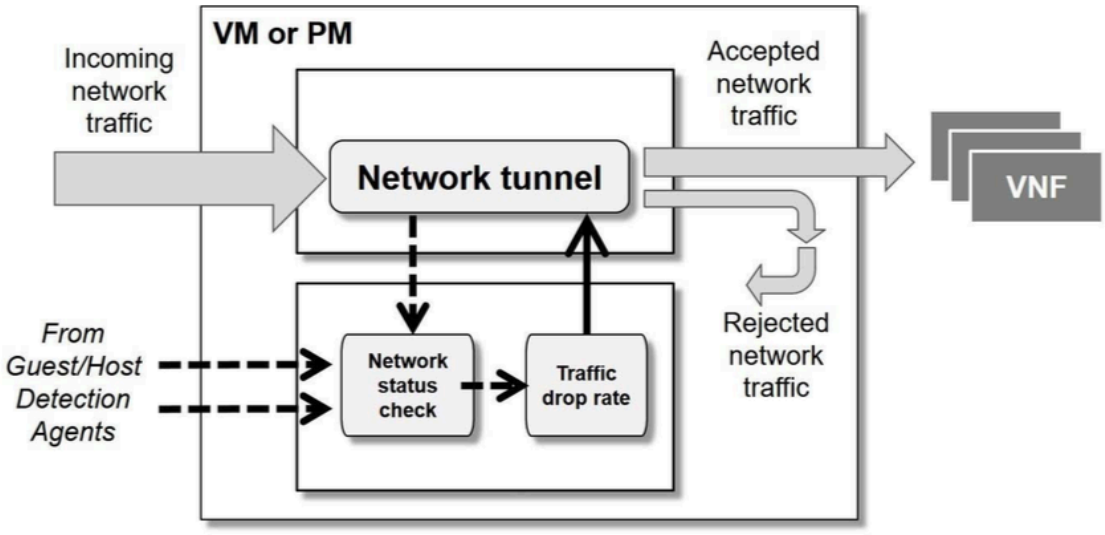


Fig. 4. Architecture of network-level overload control.

检测网络级过载条件的标准可以由NFV网络的管理员配置:一个简单的标准是计算受过载影响的VNFs的数量，并且在重载VNFs时检测过载状态。另一种可能的标准是，考虑到VNFs在NFV网络中的相对重要性，计算超载的VNFs的数量。网络级缓解代理在NFV网络的边界上充当网络隧道。它接收原本用于NFV网络的流量，并将其转发给VNFs。这个转发是通过将网络级缓解代理安装到负载均衡器中完成的，它位于NFV网络的边界，无论是在专用的VM中，还是在物理机器中。因此，网络级缓解代理对VNFs是透明的。此外，网络级缓解代理对网络延迟和吞吐量的影响很小，因为它不执行任何流量分析或操作。过量的流量不会被转发到VNFs。网络级检测代理的迭代周期比VNF-/主机级的代理长，因为它的目的是减轻持续较长时间的重载(例如，我们对网络级检测的时间周期为30秒)。对于这种持续超载，值得花额外的资源(CPU和网络)来发送通知来拒绝用户，而不是简单地减少他们的流量。

五、对NFV IMS的实验评价。

5.1NFV系统概述

为了评估过载控制框架，作者对基于nfv的IP子系统（IMS）进行了实验分析。在实际的NFV软件的环境下评估过载控制框架的能力，在过载条件下的性能，框架的开销，以及NFV软件的故障。
Clearwater IMS是IMS核心标准的开源实现。它是模块化的，它的部署只能通过启用特定的功能来定制。所有的组件都可以使用基于DNS的简单的无状态负载均衡横向扩展。此外，Clearwater遵循了可伸缩和可靠的Web服务的常见设计模式，通过保持大多数组件基本上无状态，并通过在集群数据存储中存储长寿命的状态。Clearwater是一个大型的软件项目，主要是用c++和Java编写的，包括几个子系统。

5.2实验环境

5.3实验计划

作者在Clearwater IMS案例研究的背景下，通过执行有压力的工作负载和资源争用的实验来评估拟议的超载控制框架。特别评估：

- 框架的能力以确保高吞吐量(达到系统的最大容量)，以每秒注册次数(RAPS)和成功处理没有失败的每秒调用次数(CAPS)来保证。
- 框架引入的资源开销，在CPU和内存占用方面由重载控制框架的代理所消耗。

实验使用了SIP注册和调用设置请求的混合。工作负载由SIPp流量生成器生成，以模拟SIP订阅者。每两个订户将尝试平均每5分钟注册或更新注册。在成功注册之后，订阅者可以尝试设置一个调用(概率为16%)，或者保持空闲状态直到下一次注册更新(有84%的概率)。呼叫保持时间配置为60。试验对VNF实例的数量进行了校准，并使用400k订阅者进行了初步容量规划。试验将VNF实例的数量调整为平均80%的虚拟CPU利用率，并且没有失败的请求。IMS可以用10个副本来处理这个工作负载，包括Bono、Sprout和Homestead, 4个Ralf副本，以及1个Homer和DNS的副本。每个副本运行在一个具有1个虚拟CPU的不同VM上。在这些情况下，CPU成为性能瓶颈。此外，当IMS与部署在同一物理基础结构上的其他服务竞争资源时，就会出现过载情况。试验考虑了三个高负载场景，以评估在用户峰值下的过载控制性能。每个场景分别执行4次，分别是:plain Clearwater IMS；VNF-level过载控制；Host-level过载控制；以及网络级的过载控制。总的来说，总共有12个高负荷实验。试验采用以下工作负载表：

# Subscribers	Load Level	RAPS	CAPS
400k (Engineered Level)	**100%**	**1,379**	**111**
480k (Small Overload)	120%	1,655	133
1M (Medium Overload)	250%	3,448	278
4M (High Overload)	1000%	13,793	1,111

- 加载生成(升级):当实验开始时，在初始的15分钟(初始的加速阶段)中创建了400k订阅者。系统可以在没有失败的情况下处理此负载。这个负载是由10个SIPp实例的集合生成的，在整个实验期间。这一阶段在所有的实验中都是常见的。
- 过载生产:这个阶段从20分钟开始，持续30分钟。在这一阶段，在短时间内(过在上升期)内引入了额外的子scribers。然后，所有订阅者都会不断地生成调用设置和注册更新的请求。
- 过载终止(下降):这个阶段从第50分钟开始，持续到运行结束。在此阶段，每个未注册或调用的订阅者将不会尝试重试，并将离开系统。

5.4试验结果

六、总结

这篇文章提出了一种新型的NFV-节流阀结构，用于NFV的过载控制。这个框架被设计用来支持NFV(特别是NVFIaaS和VNFaaS)的服务模型，通过提供一组过载检测和缓解代理，可以部署在vm上或者在物理主机上。这些代理通过分析CPU利用率和网络流量，采用简单和健壮的规则来控制流量下降和拒绝。