Source Address Validation in Software Defined Networks

张岩

摘要

IETF 源地址验证改进（SAVI）工作组开发了防止连接到同一 IP 链路的节点相互欺骗 IP 源地址的方法。为了支持 SAVI，网络设备供应商需要在其交换机中为各种网络环境实现 RFC 集。然而，新的网络范例软件定义网络（SDN）提供了在不在 SDN 交换机中实现额外功能的情况下部署 SAVI 的机会。本次课程的论文介绍了 SDN-SAVI 的初步设计和实现，这是 SDN 控制器上的一个模块/应用程序，用于在 SDN 网络中实现 SAVI 功能。在本提案中，所有功能均在控制器上实现，而无需修改 SDN 交换机。为了在数据平面中的数据包上实施 SAVI，控制器使用现有的 SDN 技术（如 OpenFlow）在交换机中安装绑定表。有了 SDN-SAVI，网络管理员现在只需在控制器上集成一个模块，就可以在其网络中实施 SAVI，而不必购买支持 SAVI 的交换机并更换旧的交换机。

关键词：SDN 网络；IP 源地址认证

1 引言

互联网作为全球信息化基础设施, 对世界各国政治、经济、文化、社会生活和国家安全等方面产生重大影响, 已成为重要的国家战略资源。网络空间作为继陆、海、空和太空之后的人类第五疆域, 获得了世界各国的高度重视。然而, 由于互联网设计之初没有充分考虑网络成员不可信带来的安全威胁, 网络体系结构本身存在安全设计缺陷, 这使得网络的安全可信面临极大的挑战, 成为需要突破的互联网核心技术之一。在互联网可信性缺失造成的众多潜在危害中, 借助伪造源地址发起的分布式拒绝服务攻击 (distributed denial of service, 简称 DDoS) 是当前互联网重要的安全威胁之一。DDoS 攻击通过发送海量伪造源地址的报文至目标主机, 使其超过处理负载上限或网络负载上限致使服务崩溃, 是一种频繁发生、影响最为严重的攻击手段。由于 DDoS 强大的破坏能力以及带来的经济损失, 国际互联网标准化组织 (Internet Engineering Task Force, 简称 IETF) 于 2015 年 6 月正式成立 DDoS 威胁协作 (DDoS open threat signaling, 简称 DOTS) 工作组, 专门研究减缓此类攻击的技术标准。然而, 作为 DDoS 及众多攻击手段的依赖基础——“IP 源地址伪造”问题并未得到根本性解决。特别地, 自治域间的 IP 源地址验证技术至今仍未形成任何国际标准。为了从互联网体系结构上解决安全可信问题, 自治域间的源地址验证问题是必须应对和解决的关键挑战之一[1]。

2 相关工作

2.1 下一代互联网中源地址认证的挑战

当前 Internet 的报文转发主要是基于目的地的。在转发过程中，大多数情况下不检查源 IP 地址。这使得欺骗 IP 包的源地址非常容易。攻击者通常伪造源 IP 地址，以逃避对恶意报文的责任。数据包源 IP 地址验证是 Internet 面临的最重要的挑战之一。研究和工程界在设计与源 IP 地址验证相关的机制方面做出了许多努力，例如基于密码验证的方法、基于回溯的方法和基于过滤的方法。但是，由于

增量式部署不被很好地支持和 ISP 部署这些机制的动机相对较低，这些机制在 Internet 上没有得到广泛的部署[2]。

2.2 SAVI: The IETF Standard in Address Validation

SAVI，这是一种补充入接口过滤并提供更多保护的机制。SAVI 在互联网工程任务组 (IETF) 中处于标准化的最后阶段。简而言之，SAVI 通过将源地址过滤器放置在离节点更近的地方来保护每个单独的地址免受欺骗，最好是在连接链路节点的第二层交换机中。在此部署场景中，交换机在 IP 地址和特定节点的二层绑定锚点之间配置 SAVI 绑定。节点所连接的交换机的物理端口是绑定锚的典型示例。绑定是通过检查用于配置节点 IP 地址的协议交换自动创建的。这允许交换机过滤掉与现有 SAVI 绑定不对应的数据包[3]。

3 设计

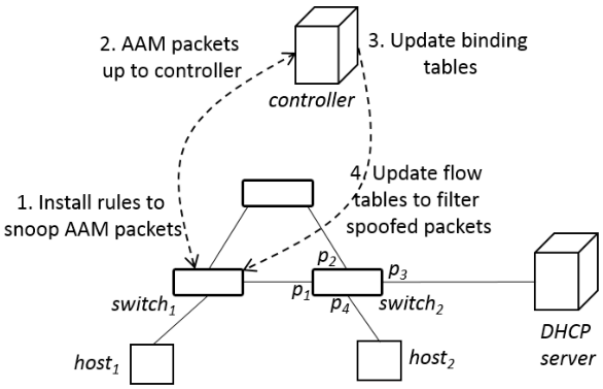

图 1: Overview of SDN SAVI

原文提出了 IPv6 的 SAVI-DHCP 和 SAVI-FCFS 的设计 (IPv4 类似)。该设计使用 OpenFlow 作为 SDN 技术。总体概览如图 1所示。最初，控制器在 OpenFlow 开关中安装几个规则来监听 AAM(address assignment mechanisms) 包。然后，交换机将每个 AAM 数据包作为 OpenFlow 数据包发送到控制器。接下来，控制器根据被窥探的 AAM 数据包指示的地址分配状态更新绑定表。最后，如果绑定表被更新，控制器修改交换机的流表来安装或删除一个与 IP 地址相关的过滤项。

3.1 AAM 监听

为了窥探 AAM 报文，控制器会提前在交换机中安装多条规则。对于 SAVI-FCFS，控制器对重复地址检测 (DAD) 报文进行监视，因此规则匹配 NS (neighbor solicit) 和 NA (neighbor advertise) 字段。对于 DHCPv6，控制器监听 DHCPv6 客户端 (DC) 消息和 DHCPv6 服务器 (DS) 消息，因此规则匹配 UDP 端口 546 和 547。与规则相关联的操作是“向控制器输出信息包”。

我们将交换机端口分为三类，分别是交换机端口、主机端口和信任端口，它们分别连接交换机、普通主机和信任服务器 (如 DHCP 服务器)。例如，图 1中 p1 和 p2 为交换机端口，p3 为信任端口，p4 为主机端口。对主机端口和信任端口进行 Snooping，只对主机端口进行校验。

控制器为每个 AAM 维护一个状态机 (例如，DHCP 的 RFC 7513)。每个 IP 地址都有一个状态，由 AAM 报文触发状态转换。当状态变为 BIND 时，在绑定表中为该 IP 地址设置绑定表项; 当状态变为 NO_BIND 时，该 IP 地址将从表中删除。

3.2 绑定表

绑定表项将 IP 源地址与交换机端口绑定，格式为 < 地址, 交换机, 端口 >，即 IP 源地址报文的入口点应为交换机的端口，否则就是欺骗数据包。

3.3 安全考虑

SDN-SAVI 模块负责对 AAM 报文进行监听，维护控制器上的状态机和定时器。这可能成为 DoS 攻击的一个新漏洞，攻击者可以通过发送 AAM 报文来覆盖控制器。为了解决这个问题，我们使用 OpenFlow 交换机的 meter 元素来限制 AAM 包的速率。这是可行的，因为在正常情况下 AAM 流量是低速率的。

强制验证的最简单方法是在控制器上执行。控制器验证每个新流的第一个包。有两种方法来处理欺骗数据包。第一种方法：忽略。它的主要缺点是攻击者可以反复发送相同的数据包，从而耗尽控制器的计算资源。第二种方法：被动地在交换机中安装与新流量匹配的丢弃规则。但是，当攻击者攻击随机源地址时，会耗尽控制器上的计算资源，也会耗尽交换机上的流表资源。

为了保护这些资源，SDN-SAVI 显式地在交换机中安装规则，以阻止未绑定的 IP 源地址。这样，欺骗报文在数据平面上进行验证和丢弃，而不消耗控制器上的流表资源和计算资源。

3.4 实现

SDN-SAVI 绑定项在交换机中使用流规则强制执行。挑战在于这些规则如何与转发等其他应用程序生成的流规则共存。一种方法是阻断未绑定的 IP 源地址，让不匹配的 (合法的) 报文由其他应用程序的规则处理。然而，在实际操作中，未绑定地址的数量可能会超过流表的大小。另一种方法是将 SDN-SAVI 的规则与其他应用结合起来，例如，对于每个绑定的 IP 源地址，生成 N 条“二维”规则，其中 N 为与 IP 目的地址匹配的转发规则的数量。然而，这种方法会导致流表爆炸。

我们使用 OpenFlow 的“多表”特性解决了流表爆炸问题。SDN-SAVI 只使用第一个流表。匹配绑定规则的数据包被定向到后一个表，由其他应用程序进一步处理，而欺骗的数据包被第一个表丢弃。该方法不仅消除了流表爆炸，而且使 SDN-SAVI 规则分离、独立。

4 复现细节

4.1 与已有开源代码对比

本项目参考了 https://github.com/ldong2014/savi-floodlight.git 的源代码作为基础框架，但该库进度偏为早期、实现的内容也十分有限，本项目对其进行了修改与改进，使其能够正常运行、增加可信端口、放行一些重要协议包的规则的补充等等。

4.2 实验环境搭建

本项目使用 Ubuntu 14.04，Floodlight 1.2 以及 Mininet 2.2.1 作为实验环境。在 Mininet 拓扑搭建上选择了一个交换机与两台主机的简单拓扑，使用 OpenvSwitch 的 add-port 功能将交换机与 Host 相连作为 DHCP 服务器提供服务的路径。

4.3 界面分析与使用说明

4.3.1 SAVI-Floodlight 控制器

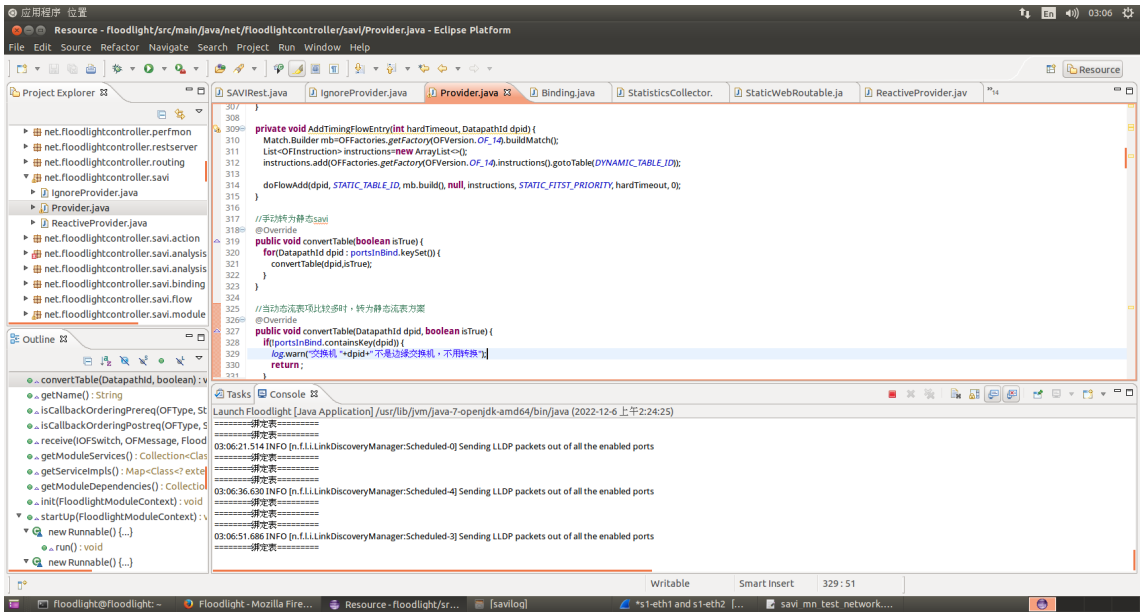

图 2: Floodlight 开发环境

Floodlight 控制器为 JAVA 实现，所以这里使用 Eclipse IDE 作为开发环境，如图 2所示。

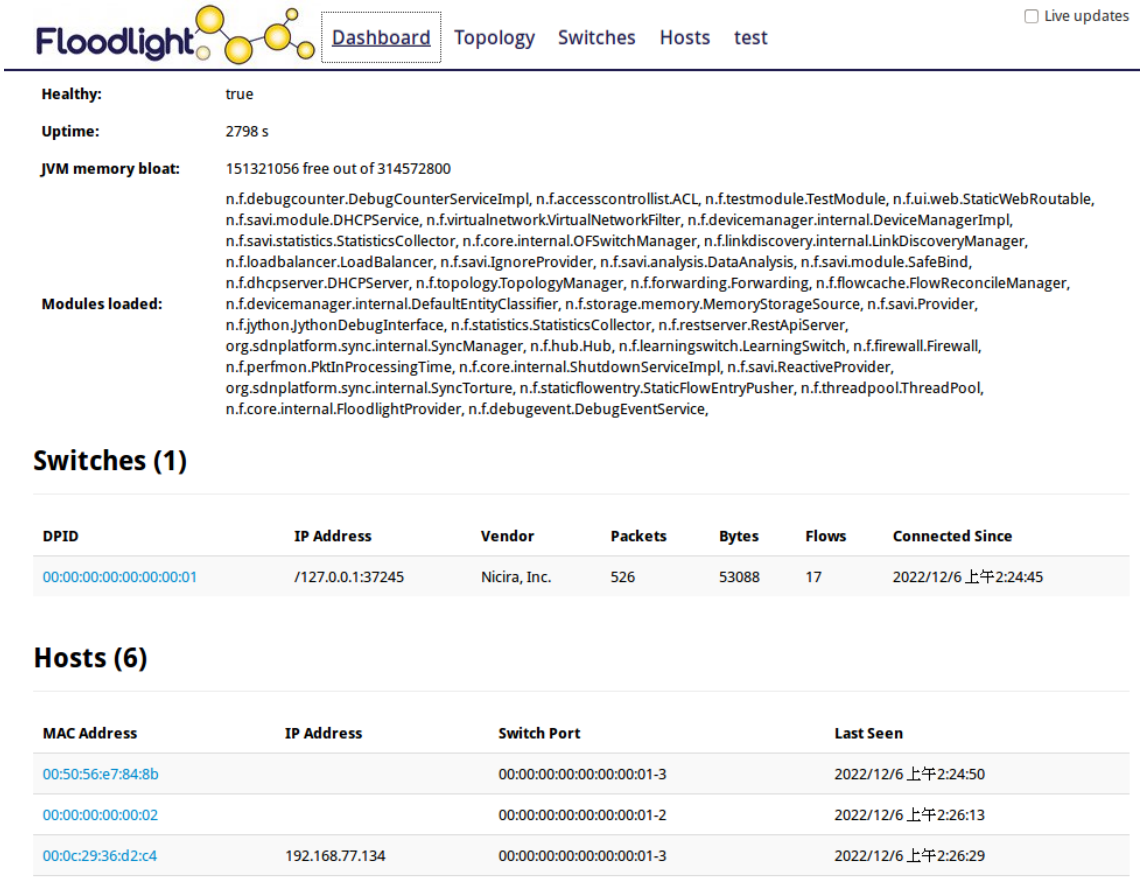

图 3: Floodlight 网页界面

同时 Floodlight 还实现了一个基于 RestAPI 的网页界面，如图 3所示，该页面可以直观的查看已加载的模块、已连接的交换机与主机等。

4.3.2 Mininet

真实的 SDN 网络需要很多 OpenFlow 设备包括控制器、交换机、功能虚拟化等一系列设备共同依赖来搭建并应用，但因条件有限，每一位实验者不可能有这样真实的环境来验证或者实验，此处将主要在没有 OpenFlow 硬件设备的条件下通过模拟搭建 Mininet 网络拓扑。

```
[sudo] password for floodlight:
*** Adding controller
*** Add switches
*** Add hosts
*** Add links
*** Starting network
*** Configuring hosts
h1 h2 h3
*** Starting controller
c1
*** Starting 2 switches
s1 s2 ...
*** h1 : ('dhclient h1-eth0',)
*** h2 : ('dhclient h2-eth0',)
*** h3 : ('dhclient h3-eth0',)
*** Starting CLI:
mininet> h1 ping fd15:4ba5:5a2b:1008:200:ff:fe00:3
ping: unknown host fd15:4ba5:5a2b:1008:200:ff:fe00:3
mininet> h1 ping6 fd15:4ba5:5a2b:1008:200:ff:fe00:3
PING fd15:4ba5:5a2b:1008:200:ff:fe00:3(fd15:4ba5:5a2b:1008:200:ff:fe00:3) 56 data bytes
64 bytes from fd15:4ba5:5a2b:1008:200:ff:fe00:3: icmp_seq=1 ttl=64 time=2.62 ms
64 bytes from fd15:4ba5:5a2b:1008:200:ff:fe00:3: icmp_seq=2 ttl=64 time=0.197 ms
^C
--- fd15:4ba5:5a2b:1008:200:ff:fe00:3 ping statistics ---
```

图 4: Mininet 运行界面

启动 Mininet 后界面如图 4所示，在命令行界面可以对拓扑结构进行简单修改、操作交换机与主机等。

4.4 创新点

所有功能均在控制器上实现，而无需修改 SDN 交换机。为了在数据平面中的数据包上实施 SAVI，控制器使用现有的 SDN 技术（如 OpenFlow）在交换机中安装绑定表。有了 SDN-SAVI，网络管理员现在只需在控制器上集成一个模块，就可以在其网络中实施 SAVI，而不必购买支持 SAVI 的交换机并更换旧的交换机。

5 实验结果分析

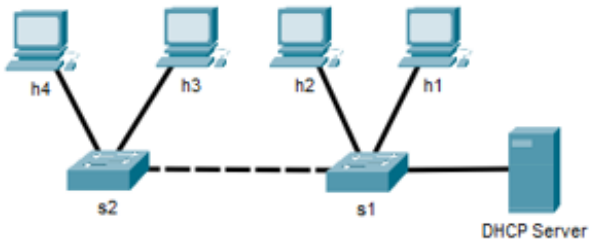

图 5: 实验拓扑图

```
========绑定表=========
SwitchPort：[s1, 1]; MAC:00:00:00:00:00:01; IP:fd15:4ba5:5a2b:1008:200:ff:fe00:1
SwitchPort：[s1, 2]; MAC:00:00:00:00:00:02; IP:fd15:4ba5:5a2b:1008:200:ff:fe00:2
SwitchPort：[s2, 1]; MAC:00:00:00:00:00:03; IP:fd15:4ba5:5a2b:1008:200:ff:fe00:3
SwitchPort：[s2, 2]; MAC:00:00:00:00:00:04; IP:fd15:4ba5:5a2b:1008:200:ff:fe00:4
```

图 6: 程序绑定表打印图

对 Floodlight 进行编译后运行，指定 Floodlight 作为 Mininet 的控制器，建立如图 5的拓扑，其中：s1-s2 与 s1-DHCP Server 所连接的端口定义为安全端口（信任所有源地址数据包）。主机 h1、h2、h3、h4 通过 s1 所连接的 DHCP 服务器获取动态 IPv6 地址，通过 floodlight 控制器识别后绑定表结果如图 6。

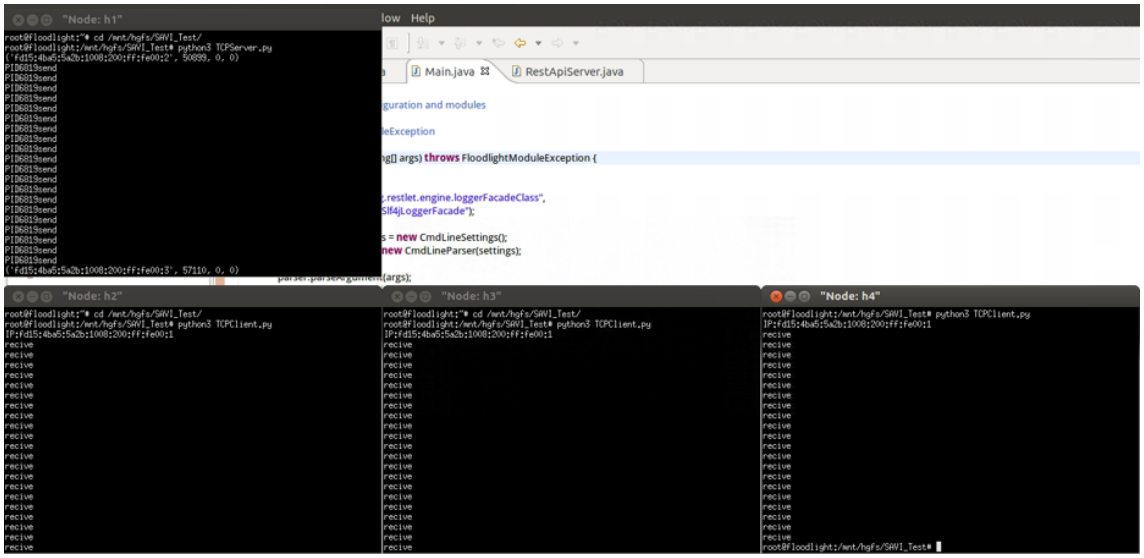

图 7: 正常通行实验图

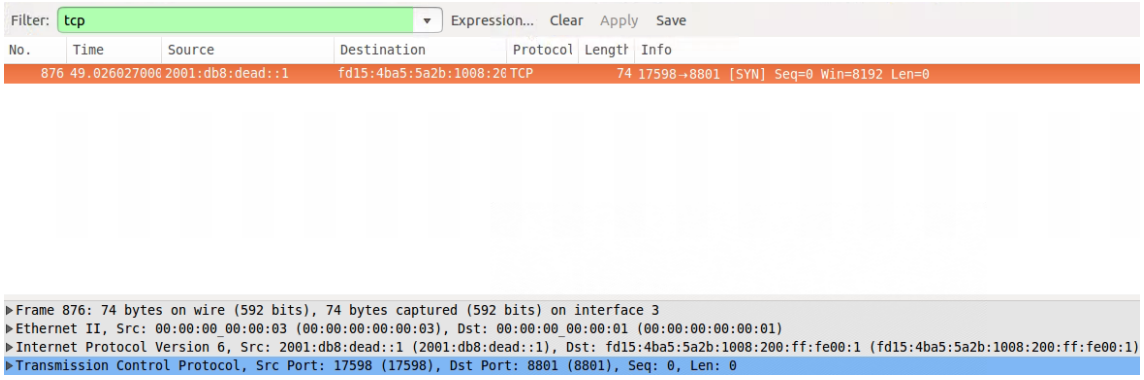

图 8: 伪造源地址通信实验图

在 h1 建立 TCP 服务端，监听 8801 端口，其他主机建立 TCP 客户端，通过服务端与客户端的信息显示可见经过绑定的源地址可以正常通信，如图 7所示。在任一客户端中使用 Python 中的 Scapy 模块封装带有伪造源地址的 SYN 包并发出，可以在 WireShark 中看到带有伪造源地址的在客户端数据包发出了，但是没有被交换机转发（被丢弃），服务端主机也未收到任何相关内容，如图 8所示。

这里采用宿主机作为 DHCP 服务器，图 5显示了交换机中的流表。SAVI 模块使用了一个优先级为“0”、操作为“发送到控制器”的流表项来转发数据包至服务器；随后是 SAVI 模块生成的优先级为“7”的流表项，用于放行一些路由协议包，使客户机可以正常获取 IP、向其他客户机发送 ICMP 等操作；优先级为“5”的作为源地址认证的规则流表下发；优先级为“6”的作为安全端口不做源地址认证的任何操作，一律放行。

```
02:28:00.940 INFO [n.f.s.s.SAVIProviderService:Scheduled-3] BIND fd15:4ba5:5a2b:1008:200:ff:fe00:1  00:00:00:00:00:00:00:01
02:28:01.941 INFO [n.f.s.s.SAVIProviderService:Scheduled-1] BIND fd15:4ba5:5a2b:1008:200:ff:fe00:2  00:00:00:00:00:00:00:01
========绑定表==========
SwitchPort：[s1, 1]; MAC:00:00:00:00:00:01; IP:fd15:4ba5:5a2b:1008:200:ff:fe00:1
SwitchPort：[s1, 2]; MAC:00:00:00:00:00:02; IP:fd15:4ba5:5a2b:1008:200:ff:fe00:2
```

图 9: 绑定日志提示

在识别到 AAM 报文后会产生绑定操作，将 < 地址, 交换机, 端口 > 元组数据加入到绑定表中，并

下发相应的通行流表给 SDN 交换机，如图 9所示。

6 总结与展望

本次课程的论文介绍了 SDN-SAVI 的初步设计和实现，这是 SDN 控制器上的一个模块/应用程序，用于在 SDN 网络中实现 SAVI 功能。在本提案中，所有功能均在控制器上实现，而无需修改 SDN 交换机。为了在数据平面中的数据包上实施 SAVI，控制器使用现有的 SDN 技术（如 OpenFlow）在交换机中安装绑定表。有了 SDN-SAVI，网络管理员现在只需在控制器上集成一个模块，就可以在其网络中实施 SAVI，而不必购买支持 SAVI 的交换机并更换旧的交换机。

本次复现可以粗略实现 SAVI 的部分功能与原理，实验结果符合预期。但是对于 IPv4 源地址认证部分的内容还未实现完整，后续可以按照该方向继续改进该项目。

参考文献

[1] 贾溢豪, 任罡, 刘莹. 互联网自治域间 IP 源地址验证技术综述[J]. 软件学报, 2018, 29(1): 20.

[2] WU J P, REN G, LI X. Building a next generation Internet with source address validation architecture[J]. 中国科学：F 辑英文版, 2008, 51(11): 11.

[3] BAGNULO M, GARCIA-MARTINEZ A. SAVI: The IETF standard in address validation[J]. IEEE Communications Magazine, 2013, 51(4): 66-73. DOI: 10.1109/MCOM.2013.6495763.