

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: February 12, 2016

W. Eddy
G. Clark
J. Dailey
MTI Systems
August 11, 2015

Customer-Controlled Filtering Using SDN
draft-eddy-sdnrg-customer-filters-01

Abstract

In order to reduce unwanted traffic and make efficient use of limited access link capacity or other network resources, it is advantageous to filter traffic upstream of the end-networks that the packets are destined to. This document describes filtering within access Internet Service Provider (ISP) networks. The ISP's end-network customers are given control over ISP filtering of traffic destined to their own prefixes, since each customer's definition of desirable versus undesirable traffic may change over time (e.g. as new network services and protocols are introduced). In this document, we describe an SDN-based means for customers to express flow definitions to their ISPs in order to distinguish between desirable and undesirable inbound traffic. These rules can be dynamically and securely updated within the running ISP network, with full automation. One use case for this capability is in mitigating denial of service attacks. Even if such filtering is only implemented in an ISP's access network, it preserves capacity on the customer access links for desirable traffic. If implemented at the ISP's edge connections to other providers, or prior to ingress to their core, it can also preserve the ISP's own network capacity and other resources that may be threatened by attacks.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 12, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
1.1.	Requirements Language	4
2.	Perceived Requirements	4
3.	Architecture	5
4.	Customer Network	8
5.	ISP Network	10
5.1.	Sub-Controller Configuration	10
5.2.	Sub-Controller Operation	11
6.	Discussion	17
7.	Acknowledgements	19
8.	IANA Considerations	19
9.	Security Considerations	19
10.	References	20
10.1.	Normative References	20
10.2.	Informative References	20
	Authors' Addresses	21

[1.](#) Introduction

At the edge of the Internet, end-network customers purchase access through Internet Service Providers (ISPs). The ISPs offer a limited amount of access link capacity to each customer, and have their own capacity limitations within their access networks, core networks, and peering to other providers. Traffic coming in from other networks to

the ISP's customers is normally forwarded through the ISP's infrastructure and to the customer access links based only on the destination IP addresses within packets.

Customers generally require reliable Internet access, and critical business operations functions rely on availability of the ISP's network resources. However, in many cases, customers are also receiving substantial amounts of undesirable traffic, including port-scans, intrusion attempts on vulnerable services in their networks, denial-of-service (DoS), and distributed DoS (DDoS) attacks. These undesirable flows are able to use up network capacity and disrupt or interfere with desirable flows.

A normal end-network customer only requires a limited set of traffic to be forwarded to them from the outside, and all other traffic can safely be filtered. In fact, a common and highly recommended practice that many end-networks already employ is to firewall undesirable incoming traffic as it comes in from the ISP's access link. This protects the end-network, but still leaves the access link itself and capacity within the ISP's network subject to abuse. Since customer networks already execute logic to distinguish between desirable and undesirable traffic, then there is benefit to both them and the ISPs in sharing that logic and pushing it upstream to increase the scope of protected resources. This allows the ISP to only devote its resources to packets that have potential value to its customers, and to quickly discard undesirable traffic.

During a DoS or DDoS attack, these upstream filters can be implemented at points of ingress from other networks and save the ISP's core and access network resources from being wasted or abused while carrying the attack traffic. Since a high-volume attack on one of its customers can have collateral impact to the ISP and its other customers, this filtering is beneficial to the ISP and its other customers, in addition to the intended victim. For instance, in 2010 an attack on a company attempting to shutdown the Mariposa botnet took down not only that company, but also several other networks using the same ISP, including a Canadian university and a few government agencies in Ottawa, according to news articles. This could have been avoided with upstream defenses.

As Software Defined Networking (SDN) technology becomes more prevalent in both customer and ISP networks, among many other capabilities, it enables the type of upstream filtering that would be beneficial in these cases. An interdomain usage of SDN allows the filter rules to be managed by the customer themselves, dynamically, and to be distributed to the ISP through automation without human intervention.

This document describes a usage of SDN that enables a customer to have all the necessary control over filtering functionality implemented within a service provider network. This description includes:

Eddy, et al.

Expires February 12, 2016

[Page 3]

Internet-Draft

Customer-Controlled Filtering Using SDN

August 2015

- o Perceived Requirements
- o Functional Architecture including ISP and Customer Components
- o Configuration and Operation of ISP Components
- o Configuration and Operation of Customer Components

This is an informational document, and not a standard. The intention of the document is to engage with the Internet community as SDN technology continues to transfer from research to increased operational use, and to discuss new ways of utilizing SDN-based network infrastructure to enhance defensive capabilities for DDoS mitigation.

We attempt to use the SDN terminology previously defined in the IRTF SDNRG [[RFC7426](#)] [[RFC7149](#)].

One of the interesting areas of SDN research is in inter-network, inter-domain, and inter-provider SDN concepts, use cases, and mechanisms. This is a challenging area because of the differing requirements between cooperating networks and their separate administrative domains and trust relationships. The configurations described in this document implement one use-case for inter-domain SDN based on OpenFlow signaling, and do so in a way that respects both customer and provider requirements on the interface.

A key goal of the architecture described in this document is for ISPs to be able to securely delegate control of their network filtering

configurations without creating new weaknesses or exposing any additional information or capabilities beyond the filtering configuration. This means that an individual customer's insight and abilities must be limited to only traffic destined for their own prefix(es).

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [[RFC2119](#)].

2. Perceived Requirements

The requirements in this section are only applicable to ISPs that choose to offer a service for customers to control upstream packet filters. They are not meant to apply to other ISPs.

The following requirements are what we perceive to be constraints on configuration of provider and customer network devices, in order to support powerful customer-controlled filtering behavior within the ISP network while still keeping the administrative boundary of the provider network intact:

1. The customer MUST be able to control traffic destined for their own prefix(es).
2. The customer MUST NOT be able to control any traffic destined for other prefixes outside their end-network.
3. The customer's ability to control filtering within ISP equipment MUST NOT change their level of access to the provider's network (e.g. it should not offer them additional insight into traffic passing through the provider that would not be available otherwise, should not expose internal or sensitive details of the provider's network architecture and internal configuration, etc).
4. The customer SHOULD be able to specify logic based on packet contents that leads to a "drop" or "accept" decision on inbound traffic, but very little or no more. An "accept" decision should

lead to provider-defined forwarding logic to select the ultimate output ports, tunnels, or functions applied to traffic towards the customer; the customer should not have control over selecting these details.

5. The customer MUST be able to access statistics on its own traffic or the filtering logic it controls from the view of the provider's network (e.g. per-filter packet match counters). This allows the customer to manage its rule set over time and remove or replace logic that is no longer needed or effective.
6. The customer MUST NOT be able to access statistics on other flows or aspects of the ISP network not related to the customer's own flows.

3. Architecture

Functionally, the entities involved in defining and implementing customer-specific filters can be divided into:

Filter management application - run by the customer to determine desired filtering rules at a given point in time. The desired filtering rules are dynamic and change as new information becomes available. For instance as attacks are detected or subside, filter rules may be added or deleted. Also, as details of an attack become known, filters may be revised to either generate

less false positives (desirable packets lost) or false negatives (undesirable packets not dropped).

Customer switching - SDN hardware and software switches within the end-network that the customer has decided to implement ingress filtering policies within.

Customer controller - SDN controller software within the end-network that instantiates filter rules provided by the filter management application. This controller is capable of managing switches within the customer end-network, at least. This controller must convey relevant limitations of any managed hardware or software switches to the filter management application so that rules can be expressed at the proper granularity given the available resources for implementing filter logic.

ISP switching - SDN hardware and software switches within an ISP network, at least some of which are made available to the customer for implementation of customer-controlled filtering rules.

ISP controller - SDN controller software in an ISP's network may be responsible for handling all switch configuration within the ISP network, other than the filtering rules provided by the customer. For instance, this controller may implement the ISP's initial ingress policies and actions, switching decisions within the ISP's network, QoS policies, and other egress functions and policies desired by the ISP.

ISP sub-controller - SDN software in an ISP's network that acts as an intermediary between switches below it, and instances of other controller software above it. The sub-controller is responsible for presenting appropriate views of the underlying network to each controller above, and for enforcing policy rules on the types of actions that each controller can perform on the underlying network.

These are all functional elements, and in some cases might either be combined together or with other concrete elements. For instance, the customer controller and filter management application could be run on the same host(s) and be implemented within a single codebase.

The ISP sub-controller is the most important element for enabling inter-domain SDN, and the majority of this document describes it.

There may be additional elements that are part of a fully detailed real-world system. For instance, detection and classification of attack traffic can be a complex task, and may involve multiple other systems before input is provided to the filter management

application. This may involve monitoring, analysis, and other functions provided by software or appliances in the customer network, coordination with cloud services and other mitigation techniques, operator alerting and possibly operator interaction and approval of a suggested response. In a basic implementation, however, the filter management could be a manual, operator-driven process. None of this specifically matters to this architecture, as ultimately filtering rules will be defined, monitored, and managed over time, no matter

how those rules are determined or coordinated with other facets of system operation.

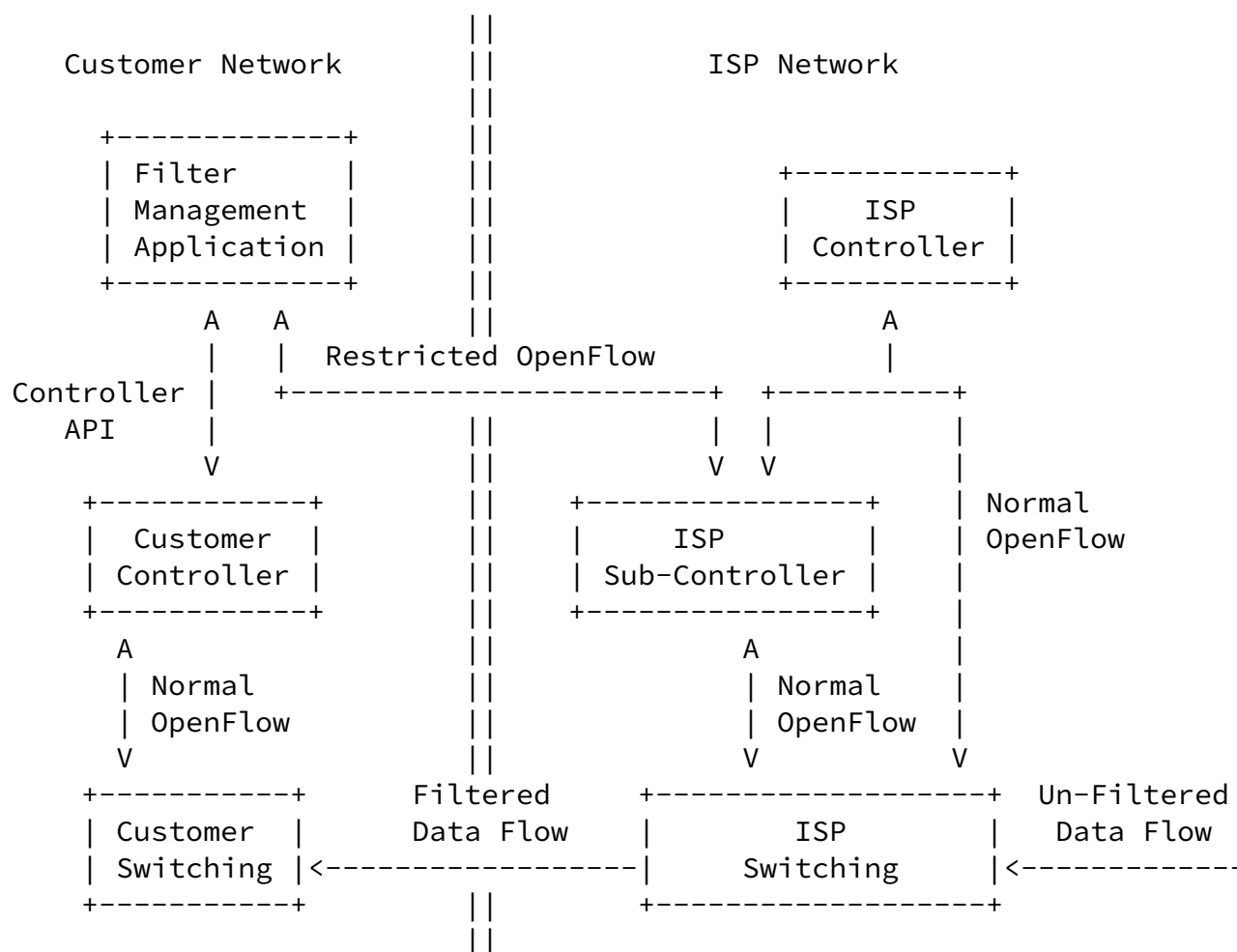


Figure 1

As illustrated in Figure 1, interfaces between elements of this architecture are fundamentally supported using the OpenFlow protocol. This is a key factor in making it possible to implement using existing customer and ISP switching elements, as OpenFlow has become commonly available. However, OpenFlow is generally used only within an administrative domain, and not inter-network between different administrative domains. Including the sub-controller element in this architecture enables OpenFlow to be utilized inter-network. Without

requiring SDN protocol modifications, the sub-controller defined here

provides one method to advance from the current single-domain state-of-practice in SDN, towards giving SDN software a wider interdomain reach and scope of view and influence. This will enable better global optimization of traffic flows, while continuing to respect the policies desired by each network's operators.

We refer to the northbound interface of the ISP Sub-Controller as "Restricted OpenFlow", because the vocabulary of OpenFlow messages that can be passed on this interface is limited to only those that conform with the requirements listed previously. Restricted OpenFlow is not a new protocol or standard, but just a subset of OpenFlow that is safe to use for implementing customer-controlled filtering functionality in an interdomain scenario. The specific subset and per-message restrictions are detailed later in this document.

Within its own network, a Filter Management Application can use any controller API that local Customer Controllers make available in order to program and monitor filter rules within the Customer Switching elements.

In the figure, we show "Normal OpenFlow" being used between many elements, but this could actually be any southbound interface protocol supported between controllers and switching elements in an SDN architecture, and does not have to be OpenFlow specifically. It is simply shown here because at the present time, it is widely available and has been used in early implementations of this architecture.

It is not illustrated in the diagram, but as an alternative, a Customer Controller element could interface with the ISP Sub-Controller using Restricted OpenFlow, instead of the Filter Management Application making this control connection into the ISP's network. Depending on the specific design of the Filter Management Application, this may be desirable to simplify its codebase and operations.

The following sections of this document discuss configuration and operation of the elements in the ISP and customer networks respectively.

[4.](#) Customer Network

End-network architectures vary widely, for instance from SOHO to enterprise campuses. Some may have multiple sub-networks, be multi-homed to different ISPs, support multiple address families (e.g. IPv4 and IPv6), use VPNs and tunnels to other networks, etc. The filter management application within the end-network may need to

understand these aspects of the network configuration, but it should not be relevant to other entities in the architecture.

The filter management application interfaces with both controller software within the customer end-network as well as the sub-controller within an ISP (the "upstream" inter-domain interface).

The purpose of the filter management application is to take a series of rules or logic that identifies undesirable traffic (as input from other software or an operator) and synthesizes them into a list of filters for upstream implementation, and another list of filters for local use within the customer network. These lists may be the same, or different, based on the capabilities of the switches, such as the number of rules they can simultaneously support or their ability to match certain packet fields.

The filter management application maintains connections with sub-controllers at upstream ISPs using a restricted variant of OpenFlow. It uses this to add, remove, and modify upstream filtering logic.

Managing filters within both the customer and ISP network switches may be useful, for instance, to have coarse-grained logic that removes the bulk of unwanted traffic upstream, and fine-grained logic that further sifts traffic at ingress to the customer network. It may also be possible to exploit more functionality within the end-network to do stateful filtering or employ other techniques that may not be possible or available in upstream ISP filters. However, it is possible that the customer network includes legacy equipment and does not support SDN at all. In this case, the filter management application only controls upstream filters.

The filter management application (or SDN controller) within the customer network listen for restricted OpenFlow connections from the sub-controllers in the ISP networks. For security reasons, connections should only be accepted from designated sub-controller addresses, otherwise an attacker might be able to extract filter logic and changes from the application and determine ways to dynamically evade filtering. Operators will need to exchange IP address (and possibly port number) information to be used for these restricted OpenFlow connections so that they can be made securely, allowed through firewalls, etc. Connecting to multiple customer filter management applications (e.g. with OpenFlow master/slave roles) is possible in order to support redundancy and failover, and is not complicated, but details are not discussed in this document.

[5.](#) ISP Network

There are a wide range of sizes and architectures for ISP networks, for instance serving different types of customers (residential, commercial, government, etc), and with different types of access networks and technologies (e.g. cable, fiber, wireless/cellular, satellite, etc). The specific way that an individual ISP configures, manages, and controls its switching elements is intended to be unaffected by implementation of customer-controlled filtering rules as described here. We provide a generic description of how the sub-controller element can be introduced and coexist with other control structures and systems within the ISP network, either based on SDN or other technologies.

There are many different specific ways the technology can be instantiated, but the same general rules apply. For instance, an ISP may create virtual routers for each customer and dedicate resources of those customer-specific virtual routers for instantiating filtering rules. In this case, there could be one sub-controller per customer and the sub-controller would be able to work almost directly with the designated virtual routers for that customer. Alternatively, another ISP might have a single sub-controller that all customers connect to, and that sub-controller could be responsible for limiting customer filtering rules to only occupy some fraction of the hardware flow table resources available within ISP switching elements common to all customers. The same concepts and configuration rules apply in all cases, even though there is a range of complexity, scaling, and performance requirements for the sub-controller element in different ISP networks.

[5.1.](#) Sub-Controller Configuration

Since a sub-controller proxies for the switching elements, it needs to behave like them and initiate OpenFlow connections upward to controllers that it has been configured to connect to.

The sub-controller has different security considerations on its intradomain OpenFlow connections with switches versus its connections to other controllers that may be interdomain. Intradomain

connections to switches and other intradomain controllers are likely to stay entirely within the ISP's infrastructure and may be on portions of the network engineered for isolation from all customer traffic and logically or physically separated from customer accessible resources. In this case, subcontroller to switching component connections can be configured in any way that meets the ISP's security requirements.

For interdomain customer controller connections, packets may traverse sections of the network that are shared or could be visible to other parties. The OpenFlow specification [[OF1.3](#)] notes "The OpenFlow channel is usually encrypted using TLS, but may be run directly over TCP". If an ISP permits sub-controller to use TCP without TLS when making connections to customer controllers, it may be exposing both parties to privacy risks [[RFC6973](#)] and other types of attacks. TLS SHOULD be used for all interdomain sub-controller to controller connections.

It is not expected that client certificates are useful for either switches or for the sub-controller when acting in its client role, however, they might be used in any cases deemed appropriate.

Since updates to rule sets are latency sensitive and may happen in a relatively hostile setting (e.g. a customer trying to push drop rules to a sub-controller while suffering from a large DDoS attack), it is beneficial for sub-controller communications to be out-of-band from normal data services to customers. For instance, this could be via a separately provisioned (e.g. assured forwarding) service class using some codepoint agreed between the provider and its customers. If this the interdomain control traffic is forwarded in-band with data traffic, some types of attacks will be capable of disrupting the control flow and could prohibit mitigations from being effectively triggered.

It is important to not have switches queueing packets while waiting for responses to new flow event notifications to a customer controller outside the ISP network. The latency and potential for packet loss (especially under attack conditions) make this undesirable. We assume instead that switches have been configured to suppress event notifications to customer controllers, and that events

are handled either by the sub-controller itself or by the ISP controllers. The customer-designated flow tables can be configured with default drop or forwarding rules, as desired so that a match is always found, and no "packet in" events need to be generated to customer controllers.

[5.2.](#) Sub-Controller Operation

When starting, an ISP sub-controller needs to establish OpenFlow connections upwards to controllers above it. These controllers may belong to the ISP itself, as well as multiple customers. The number of connections may be relatively large, compared to typical SDN usages, since an ISP may have hundreds or thousands of customers that it is enabling to control filtering. A sub-controller implementation SHOULD include features to rate-limit outgoing connections to customer controllers. This can be under operator control.

Intradomain connections to ISP controllers may be made preferentially, prior to customer connections.

An ISP sub-controller will accept OpenFlow connections from the ISP's switching elements. It will maintain an internal view of the state of those switch resources, as directly reported to it.

When processing events coming up from ISP switching elements, the sub-controller determines whether to handle the event internally, or route the event through to an upper ISP controller. Events can be routed directly to the ISP controllers without modification by the sub-controller, or they may be modified in order to provide a simplified or virtualized view of the underlying resources to the ISP controllers, depending on ISP desires and configuration.

Note that the sub-controller SHOULD NOT forward events to the customer OpenFlow connections. This prevents switching elements in the ISP network from performing poorly due to waiting upon responses from the customer, and helps to alleviate concerns about a customer's controller being able to degrade performance in the ISP network or impact the customer's own SLAs (e.g. for latency), either inadvertently or on purpose.

There are different possible implementations of a sub-controller in terms of the number of switches that it can manage. A simple sub-

controller might manage only a single switch, and in this case it will act only as a proxy and filter for OpenFlow messages. It may even modify and relay most messages directly without performing its own queries, inventing its own responses, etc. More complex sub-controllers will manage multiple switches, and will have more complex logic for processing messages, because simple manipulation and relaying will not be possible with multiple switches. For instance, an OpenFlow echo request message expects a single reply, not multiple replies.

Specific message types in the OpenFlow 1.3 specification and the proper responses to them on sub-controller reception either from a switch or a controller is described below. These are all written for the more complex situation of a sub-controller that manages multiple switches, as the single-switch situation is trivial.

Immutable Messages

OFPT_HELLO (0) - This message is processed by the sub-controller and not relayed.

OFPT_ERROR (1) - This message is processed by the sub-controller and not relayed.

OFPT_ECHO_REQUEST (2) - This message is replied to with an echo reply whenever it is received, and it is never passed through.

OFPT_ECHO_REPLY (3) - This message is processed by the sub-controller and not relayed.

OFPT_EXPERIMENTER (4) - Treatment and generation of these messages is left to be determined by each implementation.

Switch Configuration Messages

OFPT_FEATURES_REQUEST (5) - This message should not be received from a switch. When received from a controller, a reply is generated indicating the capabilities of the sub-controller, and the functionalities that it can synthesize across the set of switches that it manages. The sub-controller creates its own datapath ID representing the sub-controller, and does not feed through the datapath IDs or features of the individual

switches, but only a synthesized view of them.

OFPT_FEATURES_REPLY (6) - This message should not be received from a controller. When received from a switch, it is processed by the sub-controller and not relayed to other controllers above it.

OFPT_GET_CONFIG_REQUEST (7) - This message should not be received from a customer controller, though it might be received from an ISP controller. If the sub-controller homogenizes configuration across managed switches, then it can generate a proper reply to an ISP controller for this message.

OFPT_GET_CONFIG_REPLY (8) - This message may be sent to an ISP controller and provide indication of a homogenous configuration across the managed switches below the sub-controller.

OFPT_SET_CONFIG (9) - This message should not be received from a customer controller, though it might be received from an ISP controller. If the sub-controller homogenizes configuration across managed switches, then it can take appropriate action and generate a proper reply to an ISP controller for this message.

Asynchronous Messages

OFPT_PACKET_IN (10) - These messages may be received from switches, if the sub-controller has configured the switches to generate them. They can be relayed to ISP controllers, but should not be relayed to customer controllers (e.g. according

to the "master", "equal", or "slave" OpenFlow controller roles).

OFPT_FLOW_REMOVED (11) - These messages may be received from switches, if the sub-controller has configured the switches to generate them. They can be relayed to ISP controllers, but should not be relayed to customer controllers (e.g. according to the "master", "equal", or "slave" OpenFlow controller roles).

OFPT_PORT_STATUS (12) - These messages may be received from

switches, if the sub-controller has configured the switches to generate them. They can be relayed to ISP controllers, but should not be relayed to customer controllers (e.g. according to the "master", "equal", or "slave" OpenFlow controller roles).

Controller Command Messages

OFPT_PACKET_OUT (13)

OFPT_FLOW_MOD (14) - These are the primary messages from a customer controller that will be validated, translated, and conveyed into additional messages to managed switches. When received from an ISP controller, on the other hand, they may be passed through without change. In either case, they may be passed through by replicating each message to all managed switches, or by conveying specific messages translating the contents of the message as appropriate for each individual switch.

OFPT_GROUP_MOD (15) - These messages are treated in the same way as the OFPT_FLOW_MOD messages.

OFPT_PORT_MOD (16) - These messages should not be received from a customer controller, and when received from an ISP controller require special processing by the sub-controller in translating the port view that the ISP controller has to particular ports across the set of managed switches.

OFPT_TABLE_MOD (17) - This message does not seem to be particularly useful, and we do not expect it to be received from either ISP or customer controllers.

Multipart Messages

OFPT_MULTIPART_REQUEST (18) - These messages might be received from any type of controller, and their contents should be

processed according to the other guidelines in this list. These messages may be generated by the sub-controller in any case where a multipart request is needed to convey information to switches.

OFPT_MULTIPART_REPLY (19) - These messages might be received from switches, and their contents should be processed according to the other guidelines in this list. These messages may be generated by the sub-controller in any case that they're needed to convey information that it generates internally for controllers.

Barrier Messages

OFPT_BARRIER_REQUEST (20) - This can be generated for operations going to switches from the sub-controller. It might be received from either customer or ISP controllers and applies to the indicated messages in either case.

OFPT_BARRIER_REPLY (21) - This message might be received by the sub-controller from switches. It can also be generated by the sub-controller in response to ISP or customer controllers barrier requests.

Queue Configuration Messages

OFPT_QUEUE_GET_CONFIG_REQUEST (22) - Queue configuration setting is outside of the OpenFlow protocol scope, and may be difficult to synthesize across multiple managed switches, so these messages might not be accepted or processed by a sub-controller. They may be generated by the sub-controller itself in learning about the queue configuration of managed switches.

OFPT_QUEUE_GET_CONFIG_REPLY (23) - These messages may be received by the sub-controller from switches, but it does not seem useful to create a way to generate them from the sub-controller to either ISP or customer controllers.

Controller Role Change Request Messages

OFPT_ROLE_REQUEST (24) - Role requests may be received from either customer or ISP controllers. In either case, the sub-controller should process them normally, with the exception that the customer controller roles influence only their capabilities within the envelope of control delegated to the particular customer, and not a global role. The sub-controller may generate role request messages to managed switches in order to control its role relative to other sub-controllers or ISP

controllers accessing the same switches. Generally, only an "equal" or "master" role is sensible for a sub-controller, since it would not be able to carry out obligations to customer or ISP controllers otherwise.

OFPT_ROLE_REPLY (25) - This message is received from managed switches and also generated in response to role requests from ISP or customer controllers.

Asynchronous Message Configuration

OFPT_GET_ASYNC_REQUEST (26) - There does not seem to be a need to accept or process these messages from customer controllers, though they might be received from an ISP controller. They can be generated by the sub-controller for managed switches.

OFPT_GET_ASYNC_REPLY (27) - This message might be generated in response to an ISP controller's request, or received from a switch in response to a request from the sub-controller.

OFPT_SET_ASYNC (28) - This message might be received from an ISP controller, though does not seem useful to process from customer controllers. It can be generated by the sub-controller and sent to managed switches.

Meters and Rate Limiters Configuration Messages

OFPT_METER_MOD (29) - Meter control may be useful for both customer and ISP controllers, and the sub-controller should translate these into appropriate messages to the managed switches. The view of meters provided to controllers from the sub-controller will need to be synthesized from aggregated view of meters on the managed switches, and this requires specific logic in the sub-controller.

When validating flow table modifications from customer controllers, they need to first be checked against the particular flow table resources designated for modification by the particular customer, or transformed in some way to customer-specific flow table resources on the individual managed switches. After this, the content of the modification needs to be checked in order to make sure that the format of the rule only supports a Drop action or a GoTo-Table action for a table designated by the ISP to that customer as a Stage-3 flow table (described below). The sub-controller may ensure this by transforming the indicated GoTo-Table target into values it controls and tracks for the individual switches being managed by it.

Internet-Draft Customer-Controlled Filtering Using SDN August 2015

In OpenFlow, rules are organized into flow tables, and sets of rules across flow tables are linked via GotoTable actions

In order to ensure that the ISP retains capabilities to control traffic through its network, and that the only thing being granted to the customer is an ability to specify a drop preference on offending traffic, the sub-controller views the underlying switch flow-table organization as three stages.

Stage-1 (ISP Controlled): These contain the rules initially applied to packets ingressing the ISP's network, and is where the ISP implements its own ingress filtering and other actions. At the end of the Stage-1 flow tables, entries based on the destination IP address are matched against customer-specific prefixes, and Goto-Table actions specify the Stage-2 flow tables to be checked for customer-specified filtering rules.

Stage-2 (Customer Controlled): Customers have access to these flow tables. They MUST contain only Drop, Forward, or Goto-Table actions. The Goto-Table actions MUST indicate a Stage-3 table selected by the ISP. Each Stage-2 flow table corresponds to a particular customer and are dynamically managed by each customer. These can be safely managed without interaction with the ISP.

Stage-3 (ISP Controlled): These flow tables are used to implement the ISP's additional forwarding logic and other processing needed before an accepted packet is placed at an egress port. Since these flow tables are not reached unless Stage-2 logic has already indicated the traffic is desirable, the functionality accessed through Stage-3 tables (e.g. tunnels, etc) is able to be protected from DDoS attacks once they are detected and distinguished from desirable traffic.

The sub-controller permits only Stage-2 rules to be modified by the customer controllers, limits the flow table resources used in Stage-2 by each customer, and limits the format of rules permitted to be included in Stage-2.

[6.](#) Discussion

The FlowVisor architecture [[SGY09](#)] is similar to the one described in this document, but has some key distinctions. FlowVisor allows for the partitioning of a network into a series of logical "slices" via a series of logical flow classification rules. The rules are managed by a customized OpenFlow controller running the FlowVisor software, which acts as a transparent proxy for OpenFlow network traffic. Each unique slice has one or more OpenFlow controllers associated with it

(called "guests"). Messages to or from all guests are routed through the FlowVisor proxy, which is then responsible for examining OpenFlow messages it observes and rewriting (or discarding) them as necessary to ensure the messages only affect the pieces of the network assigned to be part of that slice.

FlowVisor could potentially be used to implement the ideas defined in this document. The caveats to this include:

Scaling the Proxy In order for the rules to be properly applied in the FlowVisor system, all messages must pass through a single proxy, and this may have scaling implications compared to the more specific sub-controller functions we described that can be distributed even to per-customer sub-controllers, because customers do not require a unified consistent view of the ISP switching elements, nor do their directives interact with one another since the "stage-2" flow tables are isolated from one another.

Hierarchical Considerations - While FlowVisor supports hierarchical deployment through partitioning or overlap of flowspace, applying rules in the presence of certain types of network designs (e.g. multiple layers of NAT) could be a challenge

Generic - FlowVisor provides a superset of the functionality required to support upstream DDoS filtering. The additional functionality may be useful in many cases, but is more heavyweight than a focused solution for DDoS filtering.

Isolation - Rewritten rules need to be carefully audited to ensure the transformations have no adverse effects at any level of a managed network(s).

When there are multiple SDN applications controlling flow table entries, there is a potential for conflict, as dealt with by systems like FortNOX [[PSY12](#)]. By creating the three-stage approach to the flow table composition, and only allowing customer access to individual stage-2 flow tables (or entries), we have largely avoided the types of conflicts that require more complex analysis and enforcement systems.

While this document focuses on the sub-controller as a hierarchical SDN approach to filtering traffic, the idea could be extended to support other applications as well. For example, quality of service or traffic-shaping applications could be requested at the upstream provider in the same way as (D)DoS filtering mechanisms. This may be possible to extend with only moderate additional sub-controller

complexity, following a similar three-stage design to the flow table configuration.

The focus in this document is on implementation of DDoS mitigations between an attack target and its direct ISPs. Pushing defenses further upstream, to higher tier ISPs or to ISPs closer to the attack traffic sources, is something we consider to be a separate problem that other protocol mechanisms can deal with [[EDC15](#)].

Several position papers from the 2015 IAB CARIS workshop discuss inter-domain collaboration for DDoS defense and possible application of SDN technology towards this goal. [[BJD15](#)] notes that standards are needed for threat feed exchanges, service requirement profiles, and dynamic negotiation with service providers. The system described in this document shows that at least for immediate ISPs of an attack target, mitigations can be implemented without new protocols or standards. The delegation of limited flow table entry management to an ISP's customers could be viewed as a simple, fast, and effective means of the dynamic negotiation functionality. [[XHH15](#)] discusses use of "big data analysis" working in conjunction with an SDN-based network. Use of any type of attack identification or characterization technique is fully compatible with the architecture described in this document, including high-performance traffic analysis appliances and other on-premises equipment. Identification and characterization of an attack is performed by a functional entity that provides input into the filter management application that we

describe, and the way this input is determined is orthogonal to whether SDN or some other mechanism is used to signal and effect a mitigation function.

7. Acknowledgements

Work on some of the material discussed in this document was sponsored by the United States Department of Homeland Security (contract HSHQDC-15-C-00017), but it does not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred. Support and feedback from Dan Massey was helpful in assessing and describing this usage of SDN for DDoS mitigation.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

DDoS mitigation is a security functionality. The architecture described in this document improves on the current ability of end-

networks to survive attacks and of ISP networks to accurately filter DDoS traffic.

Security of the system described in this document depends heavily on security of the OpenFlow control connections themselves, which is discussed in the body of the document as part of configuring the switches, sub-controllers, and controllers.

Security flaws in implementations or applications of this architecture could be used to attack an end network by rerouting or dropping its traffic within its ISPs' networks, however, these would likely need to leverage underlying flaws in OpenFlow that would have other implications more serious than this.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", [BCP 14](#), [RFC 2119](#),
DOI 10.17487/RFC2119, March 1997,
<<http://www.rfc-editor.org/info/rfc2119>>.

10.2. Informative References

- [BJD15] Boucadair, M., Jacquenet, C., and L. Dunbar, "Integrating Hosted Security Functions with on Premises Security Functions – Joint Force to Mitigate Internet Attacks", 2015.
- 2015 IAB CARIS workshop
- [EDC15] Eddy, W., Dailey, J., and G. Clark, "BGP Flow Specification Validation Using the Resource Public Key Infrastructure", 2015.
- [OF1.3] Open Networking Foundation, "OpenFlow Switch Specification Version 1.3.0", June 2012,
<<https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf>>.
- [PSY12] Porras, P., Shin, S., Yegneswaran, V., Fong, M., Tyson, M., and G. Gu, "A Security Enforcement Kernel for OpenFlow Networks", August 2012,
<<http://www.cs.columbia.edu/~lierranli/coms6998-8SDNFall2013/papers/FortNox-HotSDN2012.pdf>>.

Eddy, et al.

Expires February 12, 2016

[Page 20]

Internet-Draft Customer-Controlled Filtering Using SDN August 2015

- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", [RFC 6973](#), DOI 10.17487/RFC6973, July 2013,
<<http://www.rfc-editor.org/info/rfc6973>>.
- [RFC7149] Boucadair, M. and C. Jacquenet, "Software-Defined Networking: A Perspective from within a Service Provider Environment", [RFC 7149](#), DOI 10.17487/RFC7149, March 2014,
<<http://www.rfc-editor.org/info/rfc7149>>.
- [RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S.,

Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", [RFC 7426](https://www.rfc-editor.org/info/rfc7426), DOI 10.17487/RFC7426, January 2015, <<http://www.rfc-editor.org/info/rfc7426>>.

[SGY09] Sherwood, R., Gibb, G., Yap, K., Appenzeller, G., Casado, M., McKeown, N., and G. Parulkar, "FlowVisor: A Network Virtualization Layer", October 2009, <<http://archive.openflow.org/downloads/technicalreports/openflow-tr-2009-1-flowvisor.pdf>>.

[XHH15] Xia, L., Fu, T., He, C., Gondrom, T., and D. He, "An Elastic and Adaptive Anti-DDoS Architecture Based on Big Data Analysis and SDN for Operators", 2015.

2015 IAB CARIS workshop

Authors' Addresses

Wesley Eddy
MTI Systems

Email: wes@mti-systems.com

Gilbert Clark
MTI Systems

Email: gclark@mti-systems.com

Justin Dailey
MTI Systems

Email: justin@mti-systems.com