*Proceedings of the Asia-Pacific Advanced Network* ***2012*** *v. 34, p. 1-10.*
*http://dx.doi.org/10.7125/APAN.34.1*
*ISSN 2227-3026*

# Evaluation of OpenFlow's Enhancements

**Othman Othman M.M. [1], and Koji OKAMURA [2]**

1 Department of Advanced Information Technology, Graduate school of Information Science and Electrical Engineering, Kyushu University / 6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan. +81-92-802-2722

2 Research Institute for Information Technology, Kyushu University / 6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan. +81-92-642-4030

E-Mails: omo_5@ale.csce.kyushu-u.ac.jp; oka@ec.kyushu-u.ac.jp

**Abstract:** This paper shows our proposed design and discusses in more details about the evaluation of our two enhancements to the current OpenFlow technology [11]. OpenFlow is a promising future internet enabling technology that has a great potential to improve the current Internet by providing new functionalities and a new control scheme, and thus, enabling new smarter applications to be created. Our study aims to provide OpenFlow with two new features; *network equipment to equipment flow installation (Ne-Ne FI)*, and a new type of *proactive flows*. Through which we aim to extend OpenFlow's usability, by making it more self-aware and traffic-aware, by relieving some of the load off the OpenFlow controller, and by providing OpenFlow controller with a relaxed method to support flows with strict timing requirements.

**Keywords:** OpenFlow; Flow-Based Networks; Network Control.

## 1. Introduction

The Internet; is one of the greatest means of communication over the current and past centuries. The Internet plays an important role in our lives as it delivers information through the World Wide Web, and helps people to communicate using E-mail. More over the Internet plays an important complementary role to the traditional communication methods like telephone by providing an alternative audio and video calling service. Not to mention the wide variety of contents that it provides. All of those capabilities of the Internet promoted its use as a promising ground for many trends of commerce like the e-trade, and many other types of businesses that depends on the Internet for making profit. And for the previously mentioned reason and many other reasons, the Internet has become an indispensable part of our daily lives. And due to the

importance of the Internet, researchers have been studying ways to improve the Internet and to provide new services and capabilities to it.

One of the concepts studied by researchers is the flow concept, where a flow is a sequence of packets from a source computer to a destination; which may be another host, a multicast group, or a broadcast domain, and could consist of all packets in a specific transport connection. Grouping sequence of packets into flows is very reasonable, since that different services (applications) on the Internet have different characteristics and require different levels of support by the network, and thus flows can be used to group communications according to the type of service they belong to and then apply some rules for each group. As an example flows are used in [9] to assure different levels of Quality of Service (QoS) for different types of applications depending on the application's needs, while in [10] flows are used as a mean to apply security policies.

In 2008, OpenFlow [1] was first introduced. OpenFlow is a part of Stanford University's clean slate project. OpenFlow provides a specially designed way to control flows on the network equipment by the OpenFlow controller (which is a dedicated entity for managing flows) through using the OpenFlow Protocol. Also, it enables more freedom and flexibility in by splitting the decision making or routing from packet forwarding. According to OpenFlow; decision making can be done and modified freely by the OpenFlow controller according to layer 2, 3 and VLAN headers while the forwarding or routing is still done by routers or switches, in addition to, their original functionality. Moreover, OpenFlow defines actions to be performed on flows that can be either collection of statistics, forwarding packets, dropping packets, or manipulating packet's headers. This freedom, flexibility, and the wide range of actions performed on packets enable it to play a crucial role in developing the future Internet along with its main target which is running researchers' experiments on production networks. Despite this great flexibility of OpenFlow, there have been many concerns about the scalability of OpenFlow due to the way that the OpenFlow controller controls the OpenFlow network equipment, which forces a tight coupling between the controller and the network equipment. This would mean that the controller can be one of the bottlenecks in the system. This can be inferred by the number of flows that can be installed by the NOX controller as shown in [5] that are 30K flows per second, while, maintaining a sub-10ms install time, and the flow arrival rate in [6] that is 100K flow per second. This was also confirmed by, Michael Jarschel et al. who concluded in [12] that "When using OpenFlow in high speed networks with 10 Gbps links, today's controller implementations are not able to handle the huge number of new flows".

There have been many efforts to solve this problem. However, many of those solutions focus on the controller side, as in [13], which aims provide a distributed event-based control plane for OpenFlow. While our study tackles this problem from a different side. In this study; we make use of; the network equipment, and a new type of flows, as methods aimed to participate in solving the previously mentioned problem.

The remainder of this paper is organized as follows. We first show and overview of the motivation behind those two enhancements, and some design details; of the Network equipment to equipment flow installation, and the proactive flows in Section 2, where the full design in shown in detail in [11]. While, in Section 3, we discuss in some detail about the evaluation of those two enhancements, and conclude in Section 4.

## 2. Design

### *2.1. Network Equipment to Network Equipment Flow Installation (Ne-Ne FI) .*

According to the current OpenFlow design; flows must be installed by the controller. This means that the controller is the only entity that is responsible for installing and maintaining flows on the network equipment.

And so, we propose a new method for installing flows, that is, the "network equipment to equipment flow installation" (Ne-Ne FI) method. Through using this method, the controller does not have to program (install) flows to each one of network equipment one by one; instead it can ask the equipment to spread this flow to other equipment on behalf of the controller, this can be useful in cases where the controller needs to program non critical-start up time flows. And thus relieving some load off the controller. Also, the network equipment to equipment flow installation method can be used to make the OpenFlow network more self-aware by having the network equipment cooperate and carry loads for each other upon the need and traffic situation by having the overloaded equipment delegating some of its flows to another network equipment.

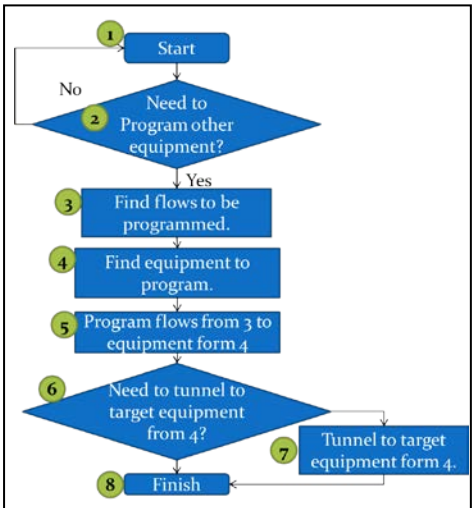


**Figure 1.** Main skeleton for the network equipment to equipment flow installation method.

The main method used by the network equipment to network equipment flow installation shown in Fig.1.

Where step 2 can be either; a r equest form the controller to spread a flow, or network equipment is overloaded and wants to delegate some of its flows to another network equipment.

Step 3 will lead to finding the flows that the controller requested to spread in case that the controller asked to spread those flows, or can lead to find an aggregation flow that aggregates other flows (it must aggregate one or more flows that are currently handling a specific percentage of the traffic according to the Traffic-Aware Flow Aggregation Algorithm (TA-FAA)). Where the network equipment that possesses those aggregated flows will delegate them to another network equipment, which means

that the delegating network equipment will delegate a number of flows responsible for a specified percentage of the traffic and replaces those flows with one delegation flow in order to tunnel the traffic of the delegated flows to the delegated network equipment.

While step 4 can be easily carried out by identifying neighbor equipment, through using the Link State Database or any other method.

Step 5 will be done using the newly proposed type of packets in the OpenFlow Protocol, which we designed to enable the network equipment to equipment flow installation.

### *2. 2. Proactive Flows*

On the original OpenFlow, whenever a flow is installed into a network equipment it will start matching against this flow and carrying out its actions. This means that the flow is active and used as soon as it is installed. However, according to this model, there will be difficulty in dealing with cases that require precise timing, since the controller must install those flows on the exact pre-specified time.

Having a centralized control would be of a greater advantage, if it can be coupled with the capability to operate precisely timed actions or flows. And to be able to provide OpenFlow with the combination of those capabilities we designed a new type of flows "The Proactive Flows" that are installed into the network equipment as inactive flows, which means that those proactive flows will not be used by the equipment that they are installed unless a certain condition activates them and enables the network equipment to use them. By having flows pre-installed into the network, and by using them in the right time; is the method used by proactive flows to tackle the precision timing difficulty.

As explained before, that the initially inactive proactive flows have to be activated in order to make use of them. We designed three conditions that can be used separately or as a combination to activate proactive flows. First condition is to receive a dedicated activation packet that contains a special activation token that can be sent by the controller, or a host, or another network equipment. The second activation method is to have an activation flow, through which a flow can be set as a condition to activate a proactive flow, and so whenever this activation flow is matched then the associated proactive flow will be activated. The third activation method is a specific time, through which a specific time is set upon which the flow will be activated.

## 3. Evaluation

This section describes the scenarios, parameters, results that we are working on, in order to assess those two enhancements. For this purpose we are using; OMNet++ [8] simulator, in addition to using Java programming language for our evaluation.

### *3.1. Network Equipment to Network Equipment Flow Installation*



#### *3.1.1. Traffic Aware Flow Aggregation Algorithm*

We have implemented the Traffic-Aware Flow Aggregation Algorithm (TA-FAA). That is a key element for using the network equipment to network equipment flow installation by overloaded equipment, in order to test its effectiveness. We implemented TA-FAA using Java programming language, we also randomly generated a flow table to resemble OpenFlow's flow table of equipment (router / switch). Then in order to assess the effectiveness of the TA-FAA, we used TA-FAA to aggregate the randomly generated flow table, and calculated its success rate in aggregating flows responsible for a specified portion of the input traffic of that equipment holding this flow table. Figure 2 shows the success rate of the TA-FAA. Our experiment showed that the average success rate of the TA-FAA was 79.7%.

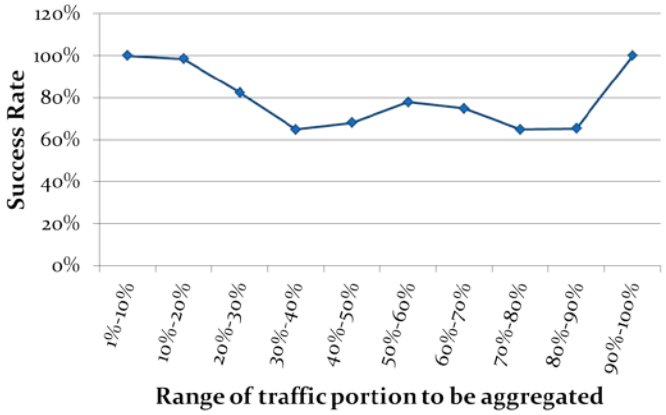


**Figure 2.** Success Rate of the TA-FAA.

*3.1.2. Ne-Ne FI for distributing flows on behalf of the controller.*

As described in Section 2.1, that Ne-Ne FI can be used to make equipment install identical flows on other equipment on behalf of the controller, as shown in Figure 3, in addition to, the regular OpenFlow's flow installation. According to this scenario we assume that the number of equipment that flows need to be installed to is (N), this means that in the regular OpenFlow method the controller needs to send (N) flow installation messages, each is destined for one equipment, while in the case of Ne-Ne FI method the controller needs to send only (one) flow installation message.

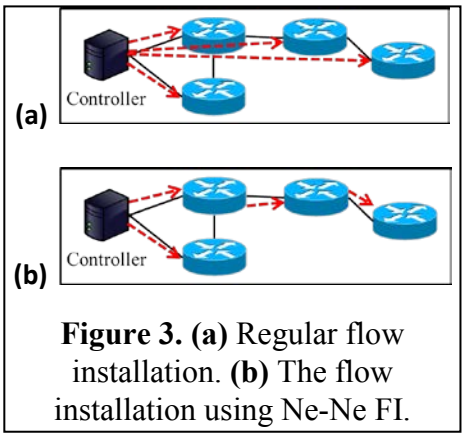


**Figure 3. (a)** Regular flow installation. **(b)** The flow installation using Ne-Ne FI.

Furthermore, to be able to judge the effectiveness of the Ne-Ne FI method and its messaging scheme for installing flows in behalf of the controller, we aim to measure the total time needed for the flows to be installed on the whole set of required equipment as shown in equation (1) as "*Total _ NeNeFI _ install _ time*". Assuming that letter E represents the set of targeted equipment that the new flows needs to be installed on. While, T(e) represents the time at which the flow installation reaches equipment e where $e \epsilon E$ , and T(0) is the time at which the controller initiated the Ne-Ne FI method.

$$Total\_NeNeFI\_install\_time = \max\{T(e) - T(0), \forall e \in E\} \quad (1)$$

Another factor that need to be assessed in order to judge the effectiveness of this method is the total number and size of packets exchanged in order to enable the Ne-Ne FI installation on behalf of the controller as shown in equation (2) as "$Total\_number\_of\_NeNeFI\_messages$" and equation (3) as "$Total\_size\_of\_NeNeFI\_messages$". Where MNe-Ne FI(e) is the number of all messages belonging to the Ne-Ne FI sent by equipment e where e∈E. And SNe-Ne FI (e) is the size of all messages belonging to the Ne-Ne FI sent by equipment e, where e∈E.

$$Total\_number\_of\_NeNeFI\_messages = \sum \left\{ \mathrm{M}_{\text{Ne-Ne FI}}(e), \forall e \in E \right\} \quad (2)$$

$$Total\_size\_of\_NeNeFI\_messages = \sum \left\{ \mathrm{S}_{\text{Ne-Ne FI}}(e), \forall e \in E \right\} \quad (3)$$

For purpose of comparison to regular OpenFlow flow installation method, we will evaluate similar parameters that are the "$Total\_regular\_install\_time$" as shown in equation (4), "$Total\_number\_of\_regular\_messages$" in equation (5), and "$Total\_size\_of\_regular\_messages$" in equation (6). Where Mregular(e), and Sregular(e) represents the number of all messages required by regular OpenFlow to install a the designated flows into equipment e by the controller, and the size required by those messages respectively.

$$Total\_regular\_install\_time = \max\left\{ T(e) - T(0), \forall e \in E \right\} \quad (4)$$

$$Total\_number\_of\_regular\_messages = \sum \left\{ \mathrm{M}_{\text{regular}}(e), \forall e \in E \right\} \quad (5)$$

$$Total\_size\_of\_regular\_messages = \sum \left\{ \mathrm{S}_{\text{regular}}(e), \forall e \in E \right\} \quad (6)$$

Based on t he previously mentioned two factors, we will run different sets of simulation where each set differs by the total number of equipment to be installed: E , ranging from 3 up to 100. Where each set of simulation will contain two simulations, one for the regular OpenFlow method and the other is for the Ne-Ne FI method. After that we will compare the parameters of the regular OpenFlow method with that of the Ne-Ne FI method in order to judge its effectiveness.

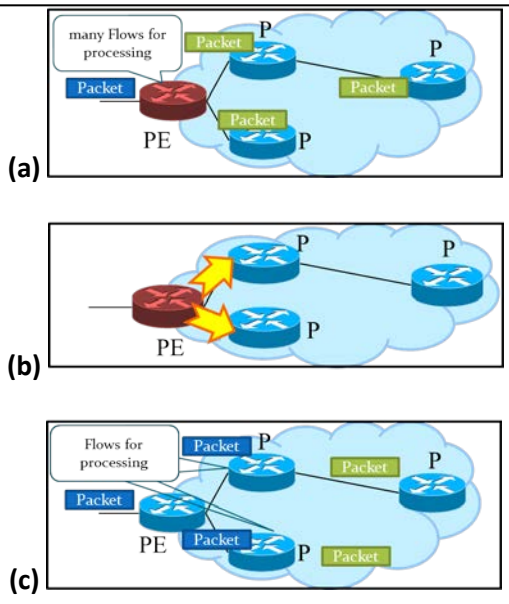


**Figure 4. (a)** Equipment overloaded, due to many flows to carry out. **(b)** Overloaded equipment delegates some flows to other equipment. **(c)** Reduced load off the overloaded

*3.1.3. Delegating Flows off the Overloaded Equipment using Ne-Ne FI.*

Ne-Ne FI can be used for a different purpose than that described in the previous Section (3.1.2). That is to use the Ne-Ne FI to delegate flows from an overloaded network equipment to other network equipment without requiring the interference of the controller. In order to assess the use of Ne-Ne FI for delegating flows off the overloaded equipment, we will use the scenario shown in Figure 4, in which an edge equipment being overloaded (due to the fact that all incoming packets to the network and outgoing packets from the network will pass through the edge, and so it might be the preferred choice to install many flows compared to other equipment).

For the purpose of this evaluation we will measure the following parameters, assuming that letter P represents the set of equipment in the network other than the overloaded edge equipment, the letter N represents the set of all network equipment in the network, and the letters pe represents the overloaded edge equipment. First parameter will be the ratio of load of the edge equipment over the average load of the other network equipment as shown in "$Ratio\_of\_overloading\_pe$" in equation (7). Where Lavg(e) represents the average load of equipment e , where e∈N.

$$Ratio\_of\_overloading\_pe = \frac{L_{avg}(pe)}{\{Average(L_{avg}(e)), \forall e \in P\}} \tag{7}$$

The second parameter will be the time needed to reduce load off the overloaded equipment as shown in "$Time\_to\_reduce\_eq\_load$" in equation (8). Where Te(load value) represents the instance of time at which equipment e, where e∈N, has reached the specified load value.

$$\begin{aligned} &Time\_to\_reduce\_eq\_load \\ &= T_{eq}(Ratio\_of\_overloading\_pe_{\geq 1}) - T_{eq}(Ratio\_of\_overloading\_pe_{\leq 1}) \end{aligned} \tag{8}$$

Based on the previously mentioned two factors, we will run different simulations where each set differs by the total number of equipment: N , ranging from 3 up to 100. After that we will assess the parameters, specially the Time_to_reduce_eq_load to check if their values are suitable for real realistic usage.

*3.2. Proactive Flows*

In order to assess the effectiveness of proactive flows, we aim to run the scenario shown in Figure 5, which shows a server migrating from one host machine to another, and an OpenFlow controller that installs a redirection flow on the OpenFlow equipment in order to redirect packets to the server using its old IP address to the new location of the new host machine, without having to wait to for routing table updates. In this scenario we will evaluate Tredir which represents the time at which the controller installs the redirection flows, and compare it with that Treq, which is the time at which the first request will arrive to the server after it completes the migration. After

that we will count the number of ignored requests under different initial loads of the controller using the OpenFlow model described in [12].

After that we will run the same scenario while using proactive flows for the redirection flows. During this scenario we will measure also Treq, and Tproactive while also counting the number of unanswered request. And finally we will compare the number of unanswered requests in the cases of regular OpenFlow and the proactive flows.

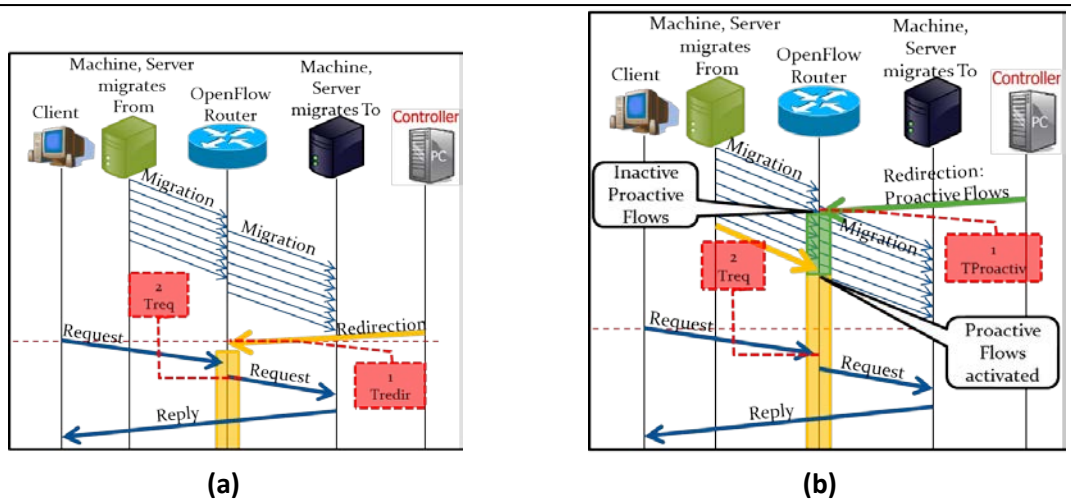


**Figure 5. (a)** Migration and Redirection using regular OpenFlow. **(b)** Migration and Redirection using OpenFlow and Proactive Flows.

## 4. Conclusions

Providing future Internet with technologies that enable it to play its role is extremely important. Because of that, many researchers are studying technologies to be the future Internet enabling technologies. OpenFlow is one of the future Internet enabling technologies, as it provides compelling functionalities that enable smarter applications to be built. However, there have been many concerns regarding the scalability of OpenFlow especially due to its dependence on a central controller.

For those reasons, our main goal in this study is to have a deeper understanding and evaluation of our efforts [11] to tackle those concerns in a different manner than others. In more details, our study aimed to relieve some load off the controller, improve the usability of OpenFlow, make it more self-aware and able to react to situations like overloading some of its equipment in a manner that takes into account the current status of the traffic, and to aid OpenFlow's ability to perform accurately timed operations. And to be able to achieve our aims we designed two enhancements to OpenFlow that are; the network equipment to equipment flow installation, and a new type of flows that is the proactive flows. In this paper, we showed our design of the protocols and algorithms needed by those enhancements to work, also showed some cases where those enhancements can play a significant role.

In order to assess the efficiency or our study; we are currently preparing simulations that we will use to measure the amount of traffic our enhancements will generate, delays needed to perform operations that are critically timed, the efficiency of our enhancements in reducing load on the controller, and the efficacy of our enhancements in introducing self-reactiveness to cases where equipment get overloaded. Our Preliminary evaluation of crucial part of the network equipment to equipment flow installation, that is the TA-FAA shows that TA-FAA success rate was 79.7%.

**References**


1. Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. O penFlow: enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev. 38, 2 ( March 2008), 69-74. DOI=10.1145/1355734.1355746 http://doi.acm.org/10.1145/1355734.1355746 .
2. W. Kim, P. Sharma, J. Lee, S. Banerjee, J. Tourrilhes, S.-J. Lee, and P. Yalagandula. Automated and Scalable QoS Control for Network Convergence. In Proc. INM/WREN, San Jose, CA, Apr. 2010.
3. Othman Othman M.M., Koji Okamura 2010. " Improvement of Content Server with Contents Anycasting Using OpenFlow". In Proceedings of the 30 APAN of Network Research Workshop.
4. S. Kandula, S. Sengupta, A. Greenberg, and P. Patel. The Nature of Datacenter Traffic: Measurements & Analysis. In Proc. IMC, 2009.
5. Tavakoli, A., Casado, M., Koponen, T., & Shenker, S. (n.d.). Applying NOX to the Datacenter. Proc. HotNets (October 2009).
6. Kandula, S., Sengupta, S., Greenberg, A., Patel, P., & Chaiken, R. (2009). The nature of data center traffic: measurements & analysis. Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference (p. 202–208). ACM.
7. Heller, B. (2009). Openflow switch specification, version 1.0.0. Wire. December.
8. OMNeT++ Network Simulation Framework. (n.d.). http://www.omnetpp.org/.
9. Song, J., Lee, S. S., Kang, K.-C., Park, N., Park, H., Yun, S., et al. (2008). Scalable Network Architecture for Flow-Based Traffic Control. ETRI Journal, 30(2), 205-215. doi: 10.4218/etrij.08.1107.0035.
10. Hong, J. W. (2004). A flow-based method for abnormal network traffic detection. 2004 IEEE/IFIP Network Operations and Management Symposium (IEEE Cat. No.04CH37507) (Vol. 1, pp. 599-612). Ieee. doi: 10.1109/NOMS.2004.1317747.
11. Othman Othman M.M., Koji Okamura 2010. " Wider Adaptation and Enhancement of OpenFlow". In Proceedings of the 33 APAN of Network Research Workshop.
12. Michael Jarschel, Simon Oechsner, Daniel Schlosser, Rastin Pries, Sebastian Goll, and Phuoc Tran-Gia. 2011. "Modeling and performance evaluation of an OpenFlow

architecture". In Proceedings of the 23rd International Teletraffic Congress (ITC'11). ITCP 1-7.
13. Amin Tootoonchian and Yashar Ganjali. 2010. "HyperFlow: a distributed control plane for OpenFlow". In Proceedings of the 2010 internet network management conference on Research on enterprise networking (INM/WREN'10). USENIX Association, Berkeley, CA, USA, 3-3.