Vulnerability Study of FlowVisor-based Virtualized Network Environments

Victor T. Costa and Luís Henrique M. K. Costa
GTA/COPPE - Universidade Federal do Rio de Janeiro - Rio de Janeiro, Brazil
{torres,luish}@gta.ufrj.br

***Abstract*—In this paper, we make a vulnerability evaluation of virtualized OpenFlow network environments based on FlowVisor. We analyze the deployment of FlowVisor in our OpenFlow testbed, verifying the consistency of the isolation mechanisms between multiple virtual networks in the presence of a malicious controller. Our analysis shows that FlowVisor's isolation can be broken and different attacks are made possible.**

I. Introduction

In the field of next-generation networks, OpenFlow[1] has appeared as an open-standard for vendor-independent networking. An OpenFlow network is composed of OpenFlow switches, which are simple forwarding elements, and an OpenFlow controller, a control element that manages the network.

In order to share the same network infrastructure among different virtual networks, such as running a test network in parallel with the production network, a special-purpose controller called FlowVisor[2] was developed. It acts as a transparent proxy between switches and controllers, and rewrites control messages according to user-defined policies in order to guarantee isolation between the multiple virtual networks, called slices. FlowVisor is currently being deployed in a variety of future Internet testbeds, where the same OpenFlow network is divided in multiple slices, such as FITS[1][3]and OFELIA[4]. The address space of each slice may be defined differently, for example by using VLAN tags or specific IP address ranges.

This work investigates FlowVisor's isolation mechanisms in order to verify the consistency of its address space isolation mechanism. We conclude that based on the slice definition, different types of vulnerabilities appear, allowing a malicious controller to break isolation and to manipulate the forwarding of packets from other slice. A similar study done by [5] discuss vulnerability assessment in OpenFlow networks in general, and this paper contributes in investigating vulnerabilities in virtualized network environments.

II. Vulnerabilities and Attacks

In an OpenFlow switch, all forwarding rules are stored in a Flow Table. A Flow Entry in this table is composed of a match, the characteristics of the packet it matches (such as IP/MAC addresses, TCP Ports); counters, that store statistics of that specific Flow Entry; and actions, which are the actions (such as forward to port or change header field) to be applied to the packets matching that Flow Entry.

Although FlowVisor is intended to provide address space isolation, some of its implementation details hinder this property. In addition, FlowVisor does not implement action isolation, meaning that there is no control over which types of actions a controller may set on a Flow Entry. Problems with this isolation mechanism were first observed in our FITS testbed, because VLAN IDs are used to define slices (each slice have a different VLAN ID). The scenarios and consequences of each vulnerability are described below under three use cases.

The **VLAN ID Access Problem** happens when Flowvisor allows the creation of a Flow Entry whose action changes the VLAN ID of the packet, when the controller is denied access to any VLAN ID. This allows a malicious controller to steal packets from or to inject packets into another slice.

The **Field Rewrite Problem** relates to when a controller has access to a specific VLAN ID tag, but can create Flow Entries with actions that change the VLAN ID of its own packets, giving to a malicious controller the opportunity to inject packets into another slice. The exact same problem repeats for other header fields, such as IP or MAC source/destination addresses or Transport source/destination numbers, so a virtual network environment using these header values (or a combination of them) to define slices would have the same problem.

The **Wildcard Rewrite Problem** happens when a controller only has access to packets with Transport source port A. If a controller tries to create a Flow Entry with an unspecified Transport source port (wildcard), FlowVisor should rewrite the wildcard value to the valid one(A), but that does not happen and the Flow Entry is created with the wildcard field, matching any Transport source port. This repeats for a other header values, such as Transport destination and Protocol Type.

III. Conclusions and Future Work

Even though FlowVisor is present in many future Internet testbeds, there are vulnerabilities that could allow a malicious controller to break isolation and interfere with other slices. We are currently examining FlowVisor's source code (version 1.4-MAINT) in order to track and fix all found vulnerabilities. Our next step is towards the creation of an action slicing mechanism so as to enforce isolation and also make the definition of slices more flexible, forcing each slice to obey its own set of allowed actions. never be able to make a certain action, for example, changing a packet's IP ToS bits in a network where quality of service is based on that header field.

Acknowledgment

This work was partially supported by CAPES, CNPq, FINEP and FAPERJ.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[2] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Flowvisor: A network virtualization layer," *OpenFlow Switch Consortium, Tech. Rep*, 2009.

[3] G. de Teleinformática e Automação (COPPE/UFRJ), "Future Internet Testbed with Security," http://www.gta.ufrj.br/fits, 2011, [Online; accessed 01-October-2013].

[4] A. Köpsel and H. Woesner, "Ofelia–pan-european test facility for openflow experimentation," in *Towards a Service-Based Internet*. Springer, 2011, vol. 6994, pp. 311–312.

[5] K. Benton, L. J. Camp, and C. Small, "Openflow vulnerability assessment," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 151–152.

[1] This work was developed under the SecFuNet project.