Towards A Secure Controller Platform for OpenFlow Applications[*]

Xitao Wen[†] Yan Chen
Northwestern University
xw@u.northwestern.edu
ychen@northwestern.edu

Chengchen Hu
Xi'an Jiaotong University
huc@ieee.org

Chao Shi
Northwestern University
chaoshi1989@gmail.com

Yi Wang
Tsinghua University
wy@ieee.org

As the brain of the OpenFlow (OF) network, the controller possesses great flexibility to define network behavior. Behind the scene of the thriving market growth, the dramatic architectural reform also fosters fresh security issues, which remain almost untouched so far. Since the control plane manages critical assets, traditionally people have to put absolute trust in the reliability and goodwill of control-plane software. Yet the openness and the absence of security enforcement make it difficult to keep such trust on the OF controller, especially on the third party modules.

The OF applications (apps), which run upon OF controller and implement a majority of the functionalities of the control plane, are typically developed by third parties. When joining with the controller, the apps inherit the privileges on manipulating the network behavior, exposing potential threats to the entire network. Although OF controllers are usually well tested, it is technically challenging to verify the trustworthiness of a third-party app. As a result, either malicious app or vulnerability in an app could lead to full compromise of the entire network.

Our survey suggests all state-of-the-art OF controller implementations effectively expose the full capacity of OF protocol without any protection mechanism. In practice, the attacker could either attack existing apps or persuade users to install malicious apps, in order to exploit the control plane. As long as the attackers control an app, they have effectively full control of the OpenFlow switches, the controller itself as well as the resources provided by OS. Such capacities could facilitate the attacks that are both lucrative and stealthy. To provide an intuitive impression, we envision four classes of attacks as follows.

- **Class 1: Direct intrusion from control plane into data plane.** With the capacity of packet-in/-out messages, the attacker can sniff/inject arbitrary data-plane packet, and thus violate the isolation between data path and control path.
- **Class 2: Leakage of sensitive information.** An compromised app can access through the controller API and leak out a variety of sensitive information, such as network flow tables and device configurations.
- **Class 3: Manipulation of OF rules.** The attacker can manipulate flows stealthily through the manipulation of OF rules, resulting in various active network attacks, such as man-in-the-middle attack and blackhole attack.
- **Class 4: Deactivating other apps.** The attack can deactivate other apps, especially security apps, by overriding the their OF rules or issuing rules that bypass their policies.

Our work calls for immediate attention to the *privilege abuse problem* of OF apps. Although the attacks are more sophisticated and harder to success than host-based attacks, this battle is obviously too critical to lose for the network operators.

Right now we are working on designing a privilege control system, which works as the first line defense against malicious app behaviors. The main design concern is two-fold: 1) what is the most effective set of privileges? 2) what isolation mechanisms should be devised to enforce the privilege control between the apps and the controller or OS. We design the privilege set generation procedure according to our understanding of four aspects: the threat models, the control message set of OpenFlow standard, the controller implementations and the functional requirements of the apps. Also, we devise both code-level and OS-level mechanism for privilege enforcement.

As future work, we envision multiple levels of defense to combat those threats. First, the controller vendor may enforce a mandatory app censorship before the distribution, which is similar to iOS App-Store. Second, the controller runtime may enforce a permission system to reduce the privilege of running apps. Finally, behavior-based analysis techniques may also facilitate online malicious behavior detection or offline forensic analysis.

[*]We will set up a demo with the poster.

[†]Xitao Wen, Chao Shi and Yi Wang are students.