Demo: Adversarial Network Forensics in Software Defined Networking

Stefan Achleitner, Thomas La Porta, Trent Jaeger, Patrick McDaniel
Computer Science and Engineering, Pennsylvania State University
University Park, PA 16802
{sachleitner,tlp,tjaeger,mcdaniel}@cse.psu.edu

ABSTRACT

The essential part of an SDN-based network are flow rules that enable network elements to steer and control the traffic and deploy policy enforcement points with a fine granularity at any entry-point in a network. Such applications, implemented with the usage of OpenFlow rules, are already integral components of widely used SDN controllers such as *Floodlight* or *OpenDayLight*. The implementation details of network policies are reflected in the composition of flow rules and leakage of such information provides adversaries with a significant attack advantage such as bypassing *Access Control Lists (ACL)*, reconstructing the resource distribution of *Load Balancers* or revealing of *Moving Target Defense* techniques.

In this demo [4, 5] we present our open-source scanner *SDNMap* and demonstrate the findings discussed in the paper *"Adversarial Network Forensics in Software Defined Networking"* [6]. On two real world examples, *Floodlight's Access Control Lists (ACL)* and *Floodlight's Load Balancer (LBaaS)*, we show that severe security issues arise with the ability to reconstruct the details of OpenFlow rules on the data-plane.

CCS CONCEPTS

•**Security and privacy →Network security;** *Security protocols; Domain-specific security and privacy architectures;*

ACM Reference format:
Stefan Achleitner, Thomas La Porta, Trent Jaeger, Patrick McDaniel. 2017. Demo: Adversarial Network Forensics in Software Defined Networking. In *Proceedings of ACM Symposium on SDN Research, Santa Clara, CA, April 3–4, 2017 (SOSR'17),* 2 pages.
DOI: http://dx.doi.org/10.1145/3050220.3060599

SOSR'17, Santa Clara, CA

DOI: http://dx.doi.org/10.1145/3050220.3060599

1 INTRODUCTION

In comparison to legacy networks, SDN enables a central controller to dynamically program the data plane and implement applications and network policies with a fine granularity of information to steer and control traffic.

Since the internals of SDN-enabled switches are not accessible to network users, they appear as a black-box and the deployed system of flow rules is assumed to be invisible to users. This central assumption is especially important when flow rules are deployed to implement security-aware policies.

Our scanner *SDNMap* is able to precisely infer the composition details of OpenFlow rules on the data plane, independent of the controller platform, by applying traffic analysis techniques that are able to gather high granularity information about SDN-based packet forwarding. Using our approach, we demonstrate that by inferring flow rule details, adversaries can reconstruct deployed network policies and use this knowledge for crafting attack traffic to bypass ACLs, map load balancing strategies or defeat moving target defense implementations to name a few examples.

This new attack vector on analyzing the precise forwarding policy of SDN switches, gives an adversary detailed knowledge of what packet and meta information is used to match packets and which actions are applied. The following is a summary of our contributions:

- ***Demonstration of OpenFlow rule reconstruction with SDNMap on real-world scenarios***
  We show a number of existing real-world application scenarios where SDNMap can be used as a reconnaissance tool to reconstruct OpenFlow-based rules and exploit their composition to execute attacks.

Our open-source tool is available at [3], that can be used as a scanner for the reconstruction of OpenFlow rules on the data plane. In [6] we discuss the details of SDNMap.

2 APPLICATION SCENARIOS

2.1 Access Control List [4]

The Floodlight SDN controller includes a number of applications, such as the implementation of an Access Control List (ACL) which deploys rules on an OpenFlow enabled switch to control network access. For the deployment of flow rules

in this scenario, we follow the configuration examples as listed on the Floodlight webpage [2].

To demonstrate SDNMap's functionality in such a scenario we created a small network of six hosts (10.0.0.1-10.0.0.6) connected by an SDN switch. By following Floodlight's ACL manual we are using its REST interface to proactively deploy rules for denying access to the host at *10.0.0.2* for host *10.0.0.1* by executing the following commands:

"src-ip":"10.0.0.1/32","dst-ip":"10.0.0.2/32","action":"deny"

"src-ip":"10.0.0.2/32","dst-ip":"10.0.0.1/32","action":"deny"

We use SDNMap to perform a scan of the network *10.0.0.0/24* from the node with address *10.0.0.1*. The scanning results of SDNMap labeled host *10.0.0.2* as being online, reported that a learning approach is used on the paths from *10.0.0.1* to *10.0.0.3*, *.4*, *.5* and *.6*, and reconstructed the following rules for the path *10.0.0.1* to *10.0.0.2*:

match=type:ip,nw_src:10.0.0.1,nw_dst:10.0.0.2 actions=drop

match=type:ip,nw_src:10.0.0.2,nw_dst:10.0.0.1 actions=drop

The reconstructed flow rules by SDNMap are an exact match with the previously deployed access control rules. An attacker using SDNMap in such a scenario is now able to determine how packets have to be crafted to be successfully delivered to the host at *10.0.0.2*. For example, sending a packet with the correct destination address but a source IP address different than *10.0.0.1*, will be delivered to host *10.0.0.2* since the Floodlight controller is forwarding packets not matching ACL rules with the default learning approach. Based on SDNMap's result we sent a probing packet with a spoofed IP source address from *10.0.0.1* to *10.0.0.2*. Since traffic not matching the access control rules is delivered based on the default learning approach, the packets from *10.0.0.1* with a spoofed IP source address are forwarded to *10.0.0.2* and the response traffic is delivered back to *10.0.0.1*.

2.2 Load Balancing as a Service [5]

The Floodlight SDN controller provides a load balancer module which is defined by the OpenStack Quantum LBaaS (Load-Balancing-as-a-Service) API. Following the configuration manual on the Floodlight webpage [1], we specified the virtual network endpoint *10.0.0.100* which is assigned to a resource pool that is handled by the physical network endpoints *10.0.0.3* and *10.0.0.4* as visualized in Figure 1.

The generated flow rules to implement this load balancing function are deployed over multiple SDN network elements and forward traffic to their assigned physical network endpoint based on their IP source address. Traffic to the virtual endpoint *10.0.0.100* from the client endpoint *10.0.0.1* is handled by the following flow rules which are automatically generated and deployed by the controller:

nw_src=10.0.0.1 actions=mod_nw_dst:10.0.0.4,output:#port

nw_dst=10.0.0.1 actions=mod_nw_src:10.0.0.100,output:#port

The deployed flow rules show that traffic originating from

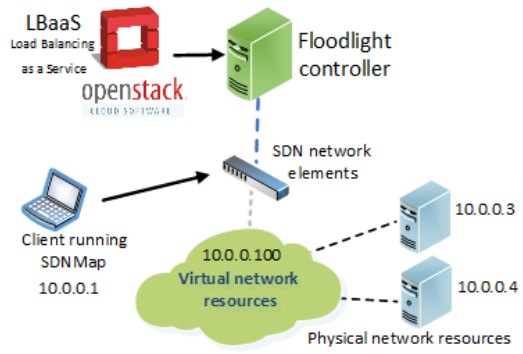

Figure 1: Load Balancing as a Service scenario

node *10.0.0.1* is forwarded based on its IP source address, and the destination address is translated to *10.0.0.4* to be handled by the physical network endpoint at this address. Return traffic to *10.0.0.1*, as defined by the second rule, has its IP source address rewritten to *10.0.0.100*, so that it appears to the client that it is communicating with *10.0.0.100* rather than *10.0.0.4*.

Using our SDNMap tool to scan host *10.0.0.100*, we are able to reconstruct the following rule which reveals the load balancing functionality:

match=type:nw_src:10.0.0.1,nw_dst:10.0.0.100

actions=mod_nw_dst:10.0.0.4,output:#OUT_PORT

This reveals to a user that address *10.0.0.100* is actually handled by a server at *10.0.0.4*. A malicious user who scans a range of virtual IP addresses is able to reconstruct the implemented load balancing policy, and launch a targeted denial of service attack on specific virtual IP addresses which all map to the same server.

ACKNOWLEDGMENT

The effort described in this article was sponsored by the U.S. Army Research Laboratory Cyber Security Collaborative Research Alliance under Cooperative Agreement W911NF-13-2-0045. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation hereon.

REFERENCES

[1] Floodlight lbaas. http://bit.ly/2d6gKUY.
[2] Floodlight sdn controller. http://www.projectfloodlight.org/floodlight/.
[3] Sdnmap repo. https://github.com/SDNMap.
[4] Video acl. https://youtu.be/rTMYvoRFc0U.
[5] Video lbaas. https://youtu.be/9v7mjMrkxHk.
[6] Achleitner, S., La Porta, T., Jaeger, T., and McDaniel, P. Adversarial network forensics in software defined networking. In *ACM Symposium on SDN Research (SOSR 2017)*.