Security Testing SDN Controllers

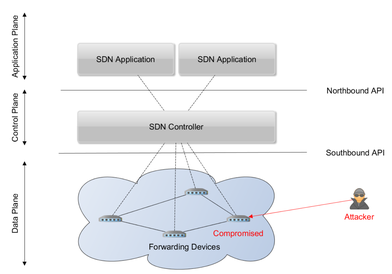
Andi Bidaj | Tuomas Aura| Aalto University, School of Science

Goal

Improve the software quality of open source SDN controllers

Attack Scenario

- Switches communicate with controllers using the OpenFlow protocol.
- Attackers compromise at least one switch or connect to the SDN controller using their own malicious switches.



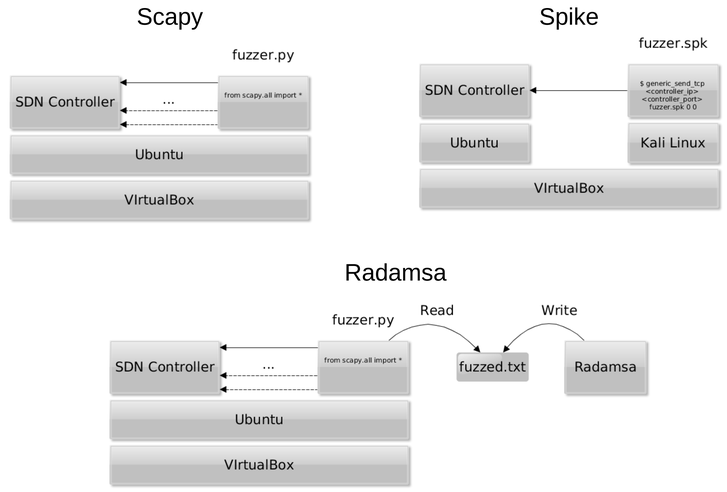
Project Overview

The goal of this project is to apply open-source security testing tools such as fuzzers different implementations of the software-defined networking (SDN) controllers such as OpenDaylight (ODL) and ONOS. The focus is on fuzz testing, which makes random mutations to previously recorded test cases or ones from model-based testing. Challenges arise from the questions of test coverage.

Fuzz Testing

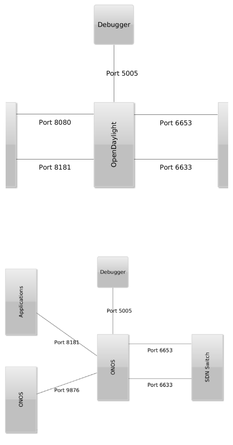
- There is no standard procedure how to apply fuzzing techniques, but generally the following steps might be valid:
- 1. *Identify target:*
- 2. *Identify inputs*
- 3. *Generate fuzzed data*
- 4. *Execute fuzzed data*
- 5. *Monitor for exceptions*
- 6. *Determine exploitability*

Testing Environment



Threat Modeling

Understanding the system



Identify the entry points and attack surface of OpenDaylight and ONOS controllers.

Identifying Threats

STRIDE Methodology:

- Spoofing
- Tampering
- Repudiation
- Information disclosure
- Denial of Service
- Elevation of Privilege

CAPEC Mitre attack library:

- Communication
- Software
- Hardware

Test implementation

Fuzzing methods:

a) Mutation based

- Random
- Increment byte
- Drop byte
- Flip byte
- Insert byte
- Swap byte
- Permute byte

b) Generation based. Protocol modeling in:

- Spike
- Scapy

Results

- Fuzz testing discovered several critical Denial of Service vulnerabilities
- Minor vulnerabilities that leak information
- Fewer vulnerabilities in ONOS than in OpenDaylight
- Threat modeling and careful test design needed to find deep bugs.

Target	Version	Type	Description
OpenDaylight	3.3, 4.0	Denial of Service	Unlimited number of threads created. ODL crashes.
OpenDaylight	3.3, 4.0	Denial of Service	Denied access to legitimate user to add new flows.
OpenDaylight	3.3	Spoofing	Insert poisoned information to the controller about a legitimate switch.
ONOS	1.5	Denial of Service	Increased resource consumption by 30-50%
OpenDaylight ONOS	3.3, 4.0 1.5	Denial of Service	Java Heap Space out of memory
OpenDaylight ONOS	3.3, 4.0 1.5	Unhandled exceptions	Unhandled exceptions when parsing several type of incorrect packets.

References

- Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.
- Michael Sutton, Adam Greene, and Pedram Amini. *Fuzzing: brute force vulnerability discovery*. Pearson Education, 2007.

Acknowledgement

- The work has been done within the scope of the CyberTrust SHOK project