Verifying Network Configurations at Scale

Presenters: Nikolaj Bjørner, Karthick Jayaraman, Jitu Padhye, Andrey Rybalchenko.

Affiliation: Microsoft Azure, Microsoft Research

Abstract

This demonstration illustrates network verification in Microsoft Azure leveraging a tool called SecGuru. SecGuru is currently being used to statically verify correctness of hundreds of thousands of access-control policies and router configurations on network devices in real time and produces reports that helps assure the correctness of network devices or root-cause issues that can be remediated pro-actively. SecGuru uses the Z3 theorem prover from Microsoft Research. This demonstration walks through the architecture of SecGuru, underlying theorem proving technology provided by Z3, and finally several of its uses cases in Azure. SecGuru showcases Microsoft Azure's deployment of Network Verification tools.

Introduction

The core technology underlying SecGuru is Z3, a state-of-the-art theorem prover. Z3 solves *Satisfiability Modulo Theories* formulas. Thus, it allows users to formulate logical formulas that use various background theories that are commonly found in software and hardware domains. SecGuru uses Z3's theory of bit-vectors that are able to directly represent header spaces.

SecGuru consumes (1) low-level network configurations (2) a statement of intent specified as a set of contracts, and outputs either that the configuration preserves the contracts or a set of rules in the configuration that violate the contract. SecGuru implements parsers for all the low-level configurations to translate them into formulas. The intent is specified using JavaScript object notation (JSON).

This demonstration will show how we encode policies into bit-vector logic formulas. These techniques and other Z3 features are also used in tools from the network community, such as MineSweeper, SyNet and Network Optimized Datalog [2].

We have developed a monitoring system for continuously verifying routing tables from network devices against the intent. For a structured network such as Microsoft Azure, the intent for the routing tables can be derived from the network architecture and metadata about the address ranges hosted in the datacenter. We leverage this aspect to perform local checks on a per router basis. These local checks together assure the end-end reachability invariants that assure our availability and performance.

Figure 1 contains the high-level architecture of this service. The service comprises of four micro services. The contract generator service creates per-device contracts for each network device and is configured to create them every few hours. The scheduler service periodically schedules the retrieval of routing tables from the network devices and pushes these work items into a queue. The device command orchestrator services pull the routing tables based on the work queued by the scheduler. Finally, the validator service validates the routing tables against the contracts and pushes the results of validating the devices into a streaming analytics system that enables summarizing the results of validations in a dashboard.

The demonstration will showcase some sample reports of validations (**Figure 2**), and the dashboard we internally use for summarizing the validation results.

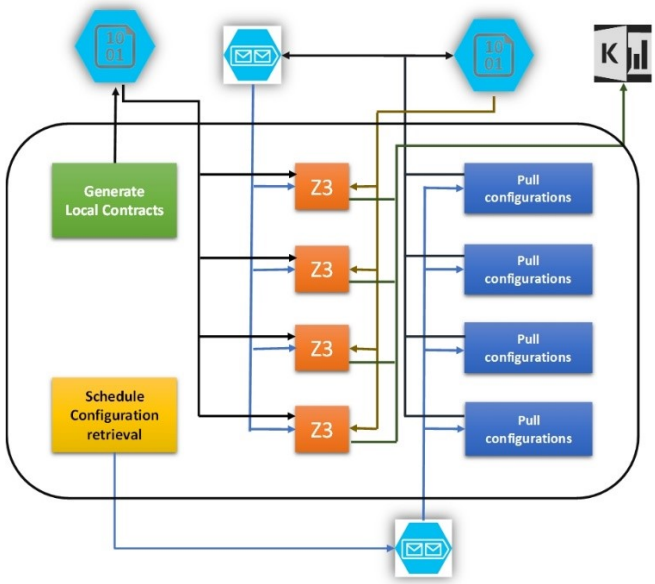

Figure 1 : Continuous Verification of Data Plane Against Intent

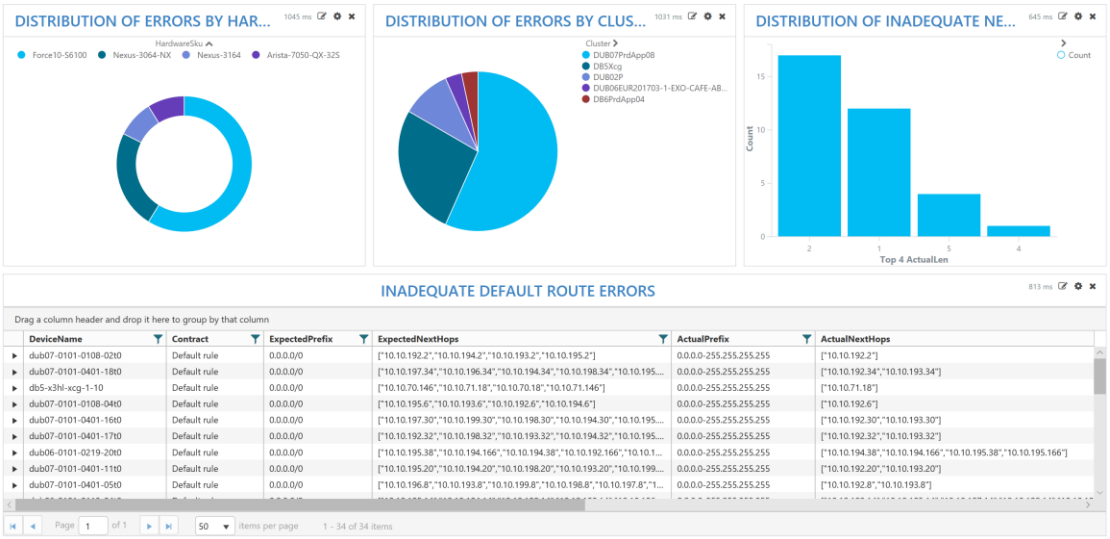

Figure 2 Sample Validation Report

References:

[1] Automated Analysis and Debugging of Network Connectivity Policies. Karthick Jayaraman, Nikolaj Bjørner, Geoff Outhred, Charlie Kaufman

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/secguru.pdf

[2] Checking Beliefs in Dynamic Networks. Nuno P. Lopes, Nikolaj Bjørner, Patrice Godefroid, Karthick Jayaraman, George Varghese

https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/lopes