CrystalNet Review Report

Jiajin Liu

Feb 2022

1 Paper Reading

CrystalNet: Faithfully Emulating Large Production Networks [1]
Hongqiang Harry Liu, Yibo Zhu, Jitu Padhye, etc.

1.1 Motivation

The problem is that there are few tools to help to improve the overall reliability of large production networks.

High reliability of large production networks is attractive for cloud and online server providers, because in such complicated large-scale networks, even small changes or issues, such as bugs in device firmwares, new configurations and human type errors etc., may have a near-disaster impact on the network and cause serious financial loss.

A promising way to avoid these harmful issues is to proactively test and validate all network operations before deployments. But unfortunately, there are few efficient tools to emulate production networks and thus we cannot accurately test and validate operations in production networks. Traditional network testbeds and verification tools cannot catch software bugs, human errors, and hardware failures because they can only scale to a limited-size topology and present a different operator workflow from real situations.

1.2 Solution

This paper proposes a scalable and high-fidelity emulator called CrystalNet to address the problem. CrystalNet can accurately mimics the behaviors of the large-scale production network including the network topology, device firmwares and operator workflows, so operators can proactively validate network operations in CrystalNet before deployments to prevent potential incidences. CrystalNet has an orchestrator to get information from the production network, provision resources on clouds, configure emulation devices and process the emulation.

To achieve high scalability, CrystalNet uses the resources of public clouds. A production network emulation where there are thousands of routers requires a huge amount of resources, which is much more than the capacity of a small cluster of servers can provide. Fortunately, there are nearly unlimited available resources in a public cloud environment, from which CrystalNet can employ enough VMs in the cloud to run emulated networks.

To achieve high fidelity, CrystalNet accurately mock-up physical networks. CrystalNet creates virtual network interfaces, virtual links and virtual management networks to mimic the setting that a switch OS runs on top of a physical switch with multiple NICs. Device firmwares in the emulation network consider that they are on real devices and management tools visit the devices in the same way as in the real network.

As it is impossible to emulate the whole Internet, a boundary is needed for the emulation network. To achieve high fidelity, the emulation boundary of CrystalNet should also be transparent and CrystalNet accurately mock-up external network. CrystalNet assumes that the network outside the emulation boundary is static, so it can save emulation resources.

A safe boundary is required to ensure the consistency among the emulated network and the real network, even when the topology or configurations of emulated devices change. The paper proposes a lemma and several propositions as the sufficient conditions to guarantee the boundary safety in GBP and OSPF networks, and more details will be discusses in the second section of this report.

1.3 Discussion

Here is the issue that may happen in CrystalNet, and my solutions are presented in this part.

When `Prepare` is called by the orchestrator, a snapshot of the production environment to be emulated will be token. In the datacenter where the delay is small enough, snapshots on different devices can be considered to have been captured simultaneously. However, in the WAN where the delay cannot be neglected, because it has a geographically larger topology than the datacenter's, different delays from devices to the orchestrator can influence the accuracy of snapshots. For example, assume the time when the device with the minimum (or maximum) delay receives the `Prepare` signal and makes a snapshot is t_1 (or t_2), if there is any update of the production network between t_1 the t_2, the snapshots of these two devices cannot be considered have been captured simultaneously. The time inconsistency of snapshots will degrade the fidelity of the emulated network.

To address this issue, there are two possible solutions:

The first way is to install several distributed orchestrators in the WAN production network to collect snapshots from different areas to reduce the delay. In the `Prepare` stage, the main orchestrator will send signals to other orchestrators, and all orchestrators will notify devices in their control area to take snapshots. The delay of any device to its nearest orchestrator can be reduced to at most d, the threshold within which the fidelity of the emulated network will not be effected.

The second way is to set a common timestamp for all devices in the production network to take snapshots. Specifically, a timestamp field can be added into the `Prepare` notification packet. The value of the timestamp should be at least the result of the current timestamp plus the maximum delay of the device to the orchestrator. Snapshots will not be token until the time reaches the timestamp, and in this way, snapshots are captured simultaneously to guarantee the fidelity of the emulated network.

1.4 Questions

The questions I don't understand about the paper are:

Q1: As the emulation is processed in public cloud, are the connections between VMs stable enough during the emulation process? Also, as the cloud is public, is it safe to emulate the production network on the public cloud?

Q2: It seems that traffic pattern and some other data plane information may also influence the network reliability, so why it is enough to mock up only with the control plane snapshots in the production network?

Q3: In Figure7 of this paper, S1-2 are not connected but they are in the same AS. From what I knew before, every device in the same AS is connected with each other. But it is not true in the case of S1-2, so does it means I mistakenly buy the statement?

2 Math formulation

2.1 Definition and Notation

Let the graph $G(V, E)$ represent the whole topology in the data center. The vertex set V contains all devices in the topology, and the edge set $E = \{(v_1, v_2)|(v_1, v_2) \in V^2, v_1 \neq v_2\}$ contains all directly linked devices pairs. Note that $(v_1, v_2) = (v_2, v_1)$. We also define V_m to be the vertex set which contains those "must-have" devices to be emulated, V_e to be the emulated vertex set which contains all devices needed to be emulated, and $E_e = \{(v_1, v_2)|(v_1, v_2) \in V_e^2, (v_1, v_2) \in E\}$ to be the emulated edge set. The collection of devices which are in V_e with any directly link to any device in the external vertex set $\overline{V_e}$ is defined as the boundary vertex set V_b:

$$V_b = \{v|(v, v') \in E, v \in V_e, v' \in \overline{V_e}\},\ \overline{V_e} = V - V_e \tag{1}$$

The collection of devices which are out of V_e but with any directly link to any boundary device is defined as the speaker vertex set V_s:

$$V_s = \{v|(v, v') \in E, v \in \overline{V}_e, v' \in V_b\} \tag{2}$$

Note that we can easily get the relationships:

$$V_b \subseteq V_e,\ V_m \subseteq V_e,\ V_e \subseteq V,\ V_s \subseteq \overline{V_e} \tag{3}$$

A boundary vertex set is safe if it can guarantee the devices reachability consistency between $G(V, E)$ and $G_e(V_e, E_e)$. The consistency can also be written as:

$$\forall (v_1, v_2) \in V_e^2,\ if\ (v_1, v_2) \in E,\ then\ (v_1, v_2) \in E_e \tag{4}$$

The problem in CrystalNet of finding the static safe emulation boundary can be considered as: given $G(V, E)$ and V_m, find a safe V_e and its V_b.

2.2 Lemma and Proposition

Lemma 2.1 *V_b is safe if and only if there is at most one vertex in the set $\{v | (v_1, v_2) \in V_e^2,\ \exists (v_1, v_2) \in E, v \in V_b,\ v \text{ is on the path from } v_1 \text{ to } v_2\}$*

In BGP network, an AS can be considered as a sub-topology of the whole network topology, and any route update from any device in the AS will not come back to any device in the the same AS. Let V_{as} be the device collection in the AS, $\overline{V_{as}} = V - V_{as}$, and $E_{asb} = \{(v_1, v_2) | v_1 \in V_{as} \text{ and } v_2 \in \overline{V_{as}}\}$ be the border edge collection of it, we can get:

$$If\ \forall e_1 \in E_{asb}\ is\ on\ a\ path,\ then\ \forall e_2 \in E_{asb}, e_2 \neq e_1,\ is\ not\ on\ the\ same\ path. \tag{5}$$

We can propose the following two promotions in BGP network:

Proposition 2.2 *If $\exists V_b \subseteq V_{as}$, $V_s \subseteq \overline{V_{as}}$, then V_b is safe.*

Proposition 2.3 *Let $V_{as_{all}} = V_{as_0} \cup ... \cup V_{as_i}$, $k = 0, 1, .., i$, $\overline{V_{as_k}} = V_{as_{all}} - V_{as_k}$, $V_p = \{v | \forall v_1 \in V_{as_k}, \forall v_2 \in \overline{V_{as_k}},\ v \text{ is on the path between } v_1 \text{ and } v_2\}$, if $V_b \subseteq V_{as_{all}}$, $V_p \cap \overline{V_e} = \varnothing$, then V_b is safe*

In OSPF network, there are two special devices named DR and BDR, which are directly linked to any devices in the whole topology and whenever there is a route update, DR and BDR should be notified. Let devices DR and BDR be V_{dr} and V_{bdr} in $G\langle V, E\rangle$.

Proposition 2.4 *If $\forall e \in \{(v_1, v_2) | v_1 \in V_b, v_2 \in V_s\}$ always remains in the E, $V_{dr} \in V_e, V_{bdr} \in V_e$, then V_b is safe.*

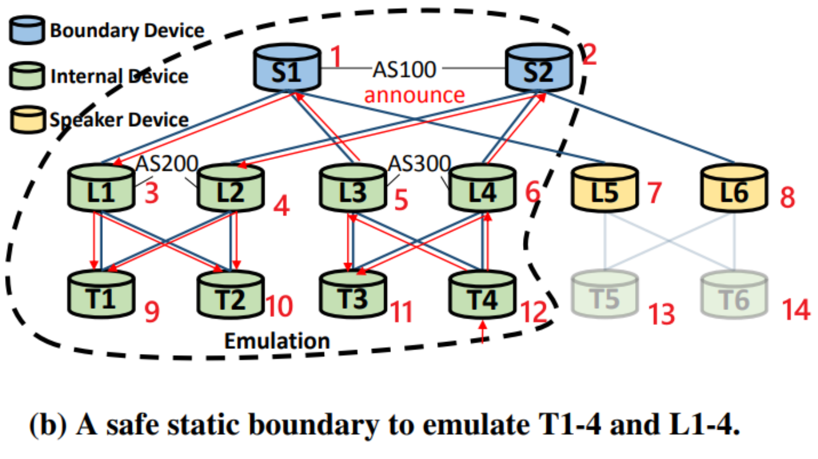
3 Programming

3.1 Implementation

The algorithm proposed to find a safe datacenter emulation boundary is based on the BFS algorithm. Here is the repository: `https://github.com/oamyjin/CrystalNet_FindSafeBoundary`

3.2 Testing

This is the input topology with tagged device ids (marked as red integers), which comes from the Figure7(b) in the paper:



The program will output the total number of emulated devices and the percentage of them among all.

Case1:
The input "must-have" devices are L1-4, T1-T4:
3 4 5 6 9 10 11 12
The output all emulated devices are:
10 71.4%
12 11 10 9 6 2 5 1 4 3

Case2:
The input "must-have" devices are L1-4:
3 4 5 6
The output all emulated devices are:
6 42.9%
6 2 5 1 4 3

References

[1] Hongqiang Harry Liu, Yibo Zhu, Jitu Padhye, Jiaxin Cao, Sri Tallapragada, Nuno P. Lopes, Andrey Rybalchenko, Guohan Lu, and Lihua Yuan. Crystalnet: Faithfully emulating large production networks. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, page 599–613, New York, NY, USA, 2017. Association for Computing Machinery.