Robustness of the detection of anomalies in a network control in case of parsimonious observation

Loïc Desgeorges, Jean-Philippe Georges, Thierry Divoux

Robustness of the detection of anomalies in a network control in case of parsimonious observation

Loïc Desgeorges, Jean-Philippe Georges, Thierry Divoux

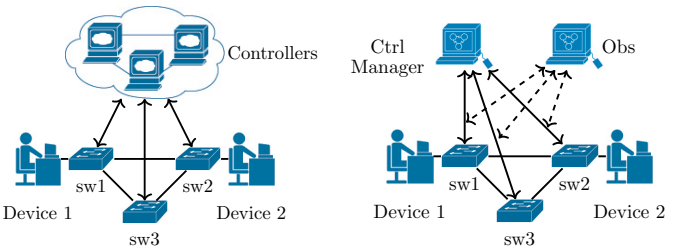
Université de Lorraine, CNRS, CRAN, F-54000 Nancy, France
(e-mail: firstname.name@univ-lorraine.fr).

Abstract: Software Defined Networking (SDN) is a networking architecture within the control is centralized through a software-based controller. Hence, the controller represents a single point of attack which makes the controller a preferred target in case of attack. In previous work, an observer has been introduced in order to detect any anomalies in the network control (in particular, in the data planes which are set up). The detection method is based on the assumption that the observer captures all the packets sent and received by the controller. This paper introduces an extension of the detection approach to improve the robustness in case of partial or parsimonious observation, more specifically when the complete data plane cannot be observed. The evaluation of the consistency of the data plane is adapted by definition of necessary condition which permits to conclude that the data plane is inconsistent. In the absence of evidence, a function which permits to compute the level of confidence in the consistency of the data plane is introduced. The efficiency and the limitations of such method is discussed on a case study.

Keywords: Detection, Security, Software-Defined Networking, Hidden Markov Models

1. INTRODUCTION

During the last decade, a huge activity in networking has focused the Software-Defined Networking (SDN) architecture, as presented in Farhady et al. (2015). Fundamentally, SDN separates the control from the network devices, by centralization of control through a software-based controller which takes all the decisions related to the network Kreutz et al. (2014). However, from a security point of view, the controller is a preferential target as summarized in Kreutz et al. (2013) and Scott-Hayward et al. (2013). In case of an attack of the controller, the attacker has access to the whole network and can damage the network (for example by flooding the tables of the switches or by modifying the content of the commands sent by the controller as developed in Lee et al. (2017) or Fonseca et al. (2012)). To solve this issue, Desgeorges et al. (2022) introduced a specific control architecture within one controller and one observer in charge of the detection of anomalies in the control as presented in Fig. 1.



(a) Classical mutli-controller architecture

(b) The proposed multi-controller architecture

Fig. 1. Classical multi-controller and observer based architecture

The topic of the paper is to study the limits of the methods to detect anomalies of the controller. Indeed, it is generally assumed that the observer has access to all the packets send and received by the controller. However, in a network context, this assumption might not be verified as shown by Petit et al. (2016) and Montanari and Aguirre (2020). Indeed, due to a burst during the observation or in case of a hybrid SDN architecture (as defined by Amin et al. (2018) and Sinha et al. (2017)), the observer does not have access to the whole decisions but only to parsimonious data. Also, it has to be mentioned the development of wireless southbound interface in the literature as presented in Dvir et al. (2019). In such a case, the capture of packets by the observer is unreliable and the activity of the control is not fully observed by the observer. Hence, the detection method developed cannot be applied anymore. Some works like Castillo et al. (2008) aim at determining the position of the sensors to observe enough information and being able to infer over the whole network. In this objective, they identified which are the link flows that can be calculated based on a subset of observed link flows. Hence, they determine the minimum link flows needed for the capture and the computation. However, parsimonious observation issues may have several origins. It can be a random lack of information (due to a miss in the capture) or it can be a technological issue (the impossibility to capture at some points). Depending on the source, the consequence is totally different. Regarding a hybrid SDN architecture, the hidden part of the data plane is fixed while in the case

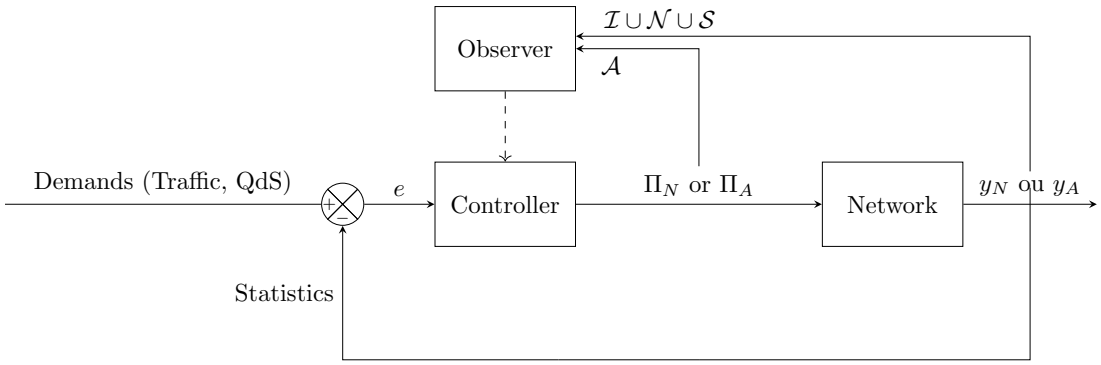


Fig. 2. Construction of the observer.

of a burst, the location, the duration and the number of packets lost may vary.

In this work, we do not make any assumptions about the observability issue. Classically, the aim of the detection approach is to determine a likelihood score for each data plane in order to determine if the data plane is abnormal or not. The problem here is to make robust the detection approach if the observer does not have a complete data plane, specifically in case of parsimonious observation.

Firstly, a reminder of our detection method is presented. Then, the observability problem is developed. A method to solve these problems is introduced and finally the method is applied on a case study.

2. ANOMALIES DETECTION APPROACH

The detection approach is represented in Fig. 2. Besides the controller in charge of the control (setting up a forwarding plane), an observer is added. It has only access to the activity of control, Σ, which corresponds to the messages exchanged at the southbound interface. The aim of the observer is to detect any anomaly in the control.

2.1 Activity of the controller

The messages exchanged at the southbound interface are normalized through a protocol name OpenFlow ONF (June, 2012). According to the specification, the packets can be classified in four types:

- $\mathcal{I}$: the set of "Packet_In" messages which are requests from the switches to the controller about what to do with an incoming flow.
- $\mathcal{A}$: the set of "Flow_Mod" messages which are the commands send by the controller to the switches. Such packets are used by the controller to set up a forwarding plane.
- $\mathcal{P}$: the set of "Port_Status" messages which are notifications from the switches of the state of their ports. This means that there is an evolution of the network topology (at the data plane level).
- $\mathcal{S}$: the set of "Multi_Part" messages which are statistics of the switches sent to the controller. These statistics are sent in response to requests from the controller though "MultiPartRequest" noted *stat*. According to ONF (June, 2012) there are several kinds of statistics given by the switch.

The activity of the controller corresponds to the set $\Sigma = \mathcal{I} \cup \mathcal{A} \cup \mathcal{P} \cup \mathcal{S}$. The list of demands $\mathcal{D}$ for which the controller decided to install active routes at time t is given by:

$$\mathcal{D}(t) = \{(s,d) \mid \exists \sigma \in \mathcal{A}, t_\sigma \in [t-\delta, t],\ s_\sigma = s,\ d_\sigma = d\}$$

δ consists of the time to live of the actions applied by the controller such that an action set up at $t-\delta-\epsilon$ is still valid at t. The active data plane $\mathcal{P}$ (the set of active routes) at a given time t is then defined by:

$$\mathcal{P}(t) = \overset{*}{\bigcup_{\forall (s,d) \in \mathcal{D}(t)}} \mu(t,s,d)$$

The aim of the detection method is to evaluate the likelihood score of every data plane $\mathcal{P}$ captured in order to determine if there is a deviation compared to the nominal behaviour.

2.2 Detection method

In order to evaluate if a data plane $\mathcal{P}$ is consistent, the observer firstly checks that each route in the plane satisfies three properties: no loop, no dead node and the destination is reached. This is necessary condition but not sufficient (the plane might not be optimal for instance). The observer needs to compare the running behaviour of the control to the unfaulty commands observed. We propose then to determine the likelihood of a data plane $\mathscr{L}(P)$ according to a multi-criteria approach:

$$\mathscr{L}(\mathcal{P}) = \Sigma_{i=1}^{n} \alpha_i \times p_i$$

with:

- $(\alpha_i)_{i \in [1,n]}$: the criteria weights such that $\Sigma_{i=1}^{n} \alpha_i = 1$
- $(p_i)_{i \in [1,n]}$: the likelihood of the criteria i

Hence, a data plane implemented by the controller is assumed to be consistent if and only if its likelihood is lower than a threshold noted TD, $\mathscr{L}(\mathcal{P}) < TD$. In this work, the limit is assumed to be fixed but it is possible to adapt the method and to consider a threshold which depends on the time.

In order to simplify the explanation, only one criterion is considered in this paper. It consists of comparing the routes of the observed planes to the ones known as consistent. It gives:

$$\mathscr{L}(\mathcal{P}) = p_1$$

with p_1 is the likelihood of the observed sequence of decisions of the control. There are several possibilities

to determine $\mathcal{L}(\mathcal{P})$. However, there is a constant: the decisions taken by the controller depend on its interns variables. We are assuming that the evolution follows a Markov Process. Thus, we propose to use the Hidden Markov Model (HMM) formalism introduced in Baum and Petrie (1966).

3. IMPACTS OF PARSIMONIOUS OBSERVATION

The previous approach was based on the assumption that every packet (requests $\mathcal{I}$ and commands $\mathcal{A}$) was observed by the observer. However, the observability in a network is still challenging Montanari and Aguirre (2020) and in a classical network context this assumption is not always verified. For example, in a SDN architecture with a wireless southbound interface, as the one presented in Ku et al. (2014), the observer may miss some packets. In such architecture, the placement of the observer may lead to the misscaputre of some packets. Hence, the activity of the control is not fully observed by the observer. Also, in a hybrid SDN approach like the one proposed in Amin et al. (2018), not all traffic pass by the southbound interface which means that the observer does not have access to the whole data plane. Also, in case of a burst transmission, some packets might not be captured by the observer (the issue may be due to memory saturation or CPU limitations). In such a case, the observer will raise an alarm while there is no specific threat. Hence, as it stands, detection methods are not robust to such situation. To be general, we do not have any assumptions about the behaviour of the observability problem and we will consider a full random observability issue. As a consequence, the set of packets involved in the control is divided in two parts:

$$\Sigma = \Sigma_O \cup \Sigma_{NO}$$

with:

- Σ_O: the set of packets observed
- Σ_{NO}: the set of packets unobserved. There is not particular assumptions about the number of packets missing and their contents.

To simplify, we assume that the controller is pro active (commands are installed periodically, not in response to specific requests $\mathcal{I}$). Hence, only the actions are considered such that:

$$\Sigma = \mathcal{A}_O \cup \mathcal{A}_{NO}$$

Consider the topology given in Fig. 3.

Let's focus on the installation of a route between the nodes 4 and 11. In case of a burst during the observation, the observer is not able to capture all the commands sent by the controller and it cannot recover the full data plane. A parsimonious observation of a nominal route (4-2-1-9-10) may lead to the two following incomplete planes:

$$\mathcal{P}_1 = \begin{pmatrix} 1-2 \\ 9-10 \end{pmatrix} \qquad \mathcal{P}_2 = \begin{pmatrix} 4-2 \\ 2-1 \\ 1-9 \\ 10-11 \end{pmatrix}$$

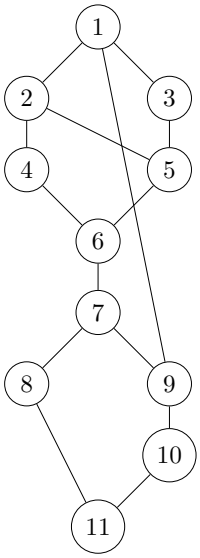
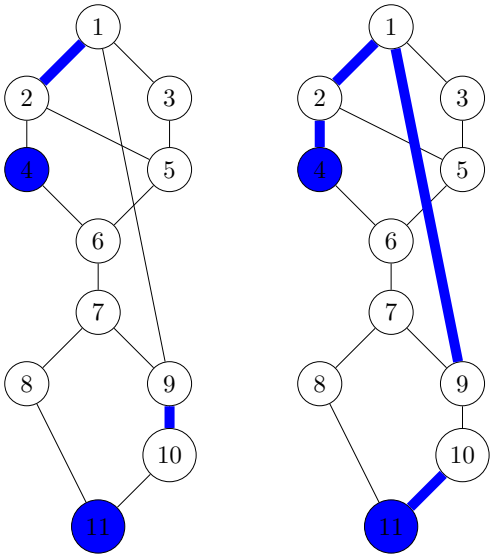


Fig. 3. Topology

They are graphically represented, respectively, on Fig. 4a and Fig. 4b. The blue lines represent the decisions contained in the Flow_Mod packets observed by the observer ($\mathcal{A}_O$).



(a) $\mathcal{P}_1$ (b) $\mathcal{P}_2$

Fig. 4. The two data planes captured

A first task of the observer might be to determine what is missing. Regarding Fig. 4b, there are not a lot of possibilities. It seems like just the last command is missing. However, regarding Fig. 4a, there are a lot of possible paths from this composition. In both cases, the objective is to find a method to determine that these captured planes ($\mathcal{P}_1$ and $\mathcal{P}_2$) are in fact an incomplete observation of the full plane and as a consequence, to not detect here an error of the controller. The challenge is to avoid accepting planes that appear to be incomplete when they are the result of a failure or an attack (like the *Control Message Drop* attack presented in Lee et al. (2017)).

It has to be mentioned that the captures given in $\mathcal{P}_1$ and $\mathcal{P}_2$ (as represented in Fig. 4) correspond to two

examples of parsimonious observation which justify the related problematic. However, the objective is not to be exhaustive over the possible parsimonious observations.

4. METHODOLOGY

Firstly, the detection method is based on *a priori* knowledge of the control logic. As presented in Desgeorges et al. (2021), the observer lets a fixed time (δ) to the controller to send all commands ($\mathcal{A}$) in response to a request ($\mathcal{I}$) or a time event. Based on the commands captures, the observer should then check the consistency of the data plane.

However, if there is observability issues, the data plane might not fully be observed and so the verification might lead to a non-consistent plane and the observer might raise an alarm. There are two possibilities, the data plane is fully captured, then a detection approach can be applied to determine its consistency. Otherwise, what follows defined the detection method applied.

4.1 Consistency

The first thing is to verify the consistency of the data plane. Here, we are assuming that the data plane is not fully captured as represented on Fig. 4 such that the paths are not full. Indeed, the aim of this step is to check that the data plane captured verifies a set of properties which is sufficient to assume that the full data plane is inconsistent. Regarding the properties introduced in Desgeorges et al. (2021) for a routing application, we can verify that there is no loop. However, it is not possible to conclude with the consistency regarding the two following properties:

- No dead node: we may have dead node in the part of the data plane observed because some packets (commands) are missing in the capture
- The destination has to be reached: the command in direction of the destination may not have been captured

Since the consistency of the data plane cannot be directly evaluated, we will assume that it is verified. As it is a strong assumption, the level of confidence in this assumption needs to be determined. In this objective, we introduce $i_{Conf}(\mathcal{P})$ as the level of confidence of the data plane $\mathcal{P}$. We propose here to define this function as the ratio between the number of commands $\mathcal{P}$ and the average number of commands in a data plane observed $\mathcal{P}_{Obs}$, as follows:

$$i_{Conf}(\mathcal{P}) = \frac{|\mathcal{P}|}{average_{\mathcal{P}_{Obs}}(|\mathcal{P}_{Obs}|)}$$

A limit TD of this ratio has to be fixed such that if $i_{Conf}(\mathcal{P}) > TD$, then we made the assumption that the data plane is consistent. Based on this assumption, the likelihood score of the data plane has to be determined.

4.2 Likelihood

As a reminder, the likelihood score of the data plane is determined using a HMM which analyzes the statistics of the sequence of data plane observed. The set of all possible unobserved actions which completes $\mathcal{A}_O$ in order to obtain a consistent data plane is defined by $\mathcal{A}_{NO,P}$:

$$\forall \mathcal{A}_{NO} \in \mathcal{A}_{NO,P} \implies \mathcal{A}_{NO} \cup \mathcal{A}_O = \mathcal{P} | \ \mathcal{P} \text{ is consistent}$$

Considering the set of actions observed $\mathcal{P}_1$, shown in Fig. 4b, we have:

$$\mathcal{A}_{NO,P} = \left\{ \begin{pmatrix} 9-10 \end{pmatrix}, \begin{pmatrix} 9-7 \\ 7-8 \\ 8-10 \end{pmatrix} \right\}$$

The correct data plane is part of this set. Since it is not possible to prove that the part of the data plane captured is inconsistent, a level of reliability should be given to the controller by tolerating more or less that information is missing and thus potentially reinforcing the likelihood. The most restrictive definition is to consider the worst case such that the likelihood score is computed as follows:

$$\mathcal{L}'(\mathcal{P}) = \min_{\mathcal{A}_{NO} \in \mathcal{A}_{NO,P}} \mathcal{L}(\mathcal{P}' = \mathcal{A}_{\mathcal{NO}} \cup \mathcal{A}_{\mathcal{O}}) \qquad (1)$$

It is also possible to be more flexible and considered other functions such as the *average* as follows:

$$\mathcal{L}'(\mathcal{P}) = \text{average}_{\mathcal{A}_{NO} \in \mathcal{A}_{NO,P}} \mathcal{L}(\mathcal{P}' = \mathcal{A}_{\mathcal{NO}} \cup \mathcal{A}_{\mathcal{O}}) \qquad (2)$$

It is clear that the length of $\mathcal{A}_{NO,P}$ has an impact over the tolerance let to the controller. As an example, let us consider the two captures $\mathcal{P}_1$ and $\mathcal{P}_2$. In the first capture, there are many more possibilities to obtain a consistent data plane and so being flexible would significantly increase the tolerance, and may lead to accept malicious plane.

5. CASE STUDY

The aim of this section is to evaluate the method proposed.

5.1 Scenario

The considered topology is the one of the network GEANT which is the European data network for research and educational communities. It gathers 23 nodes and 37 links as shown in Fig. 5. The network is here simulated in Mininet [1] .

In this case study, we are focusing on the multi-objective proactive routing strategy proposed in Casas-Velasco et al. (2020). It is based on a reinforcement learning technique aiming at optimizing the delay, the packet loss and the available link bandwidth. For the traffic, we are using the dataset TOTEM [2] which provides intra-domain traffic matrices for the GEANT topology. The control is attacked by setting up malicious data planes as developed in Lee et al. (2017) such that some demands might not be served.

To simulate control packet loss, we randomly deleted from the observed data plane some commands ($\mathcal{A}$), such that the determination of the cause of an incomplete plane remains ambiguous (might be due to a loss or an attack). The commands deleted are chosen here in order to retain

[1] `http://mininet.org/`

[2] `https://totem.info.ucl.ac.be/dataset.html`

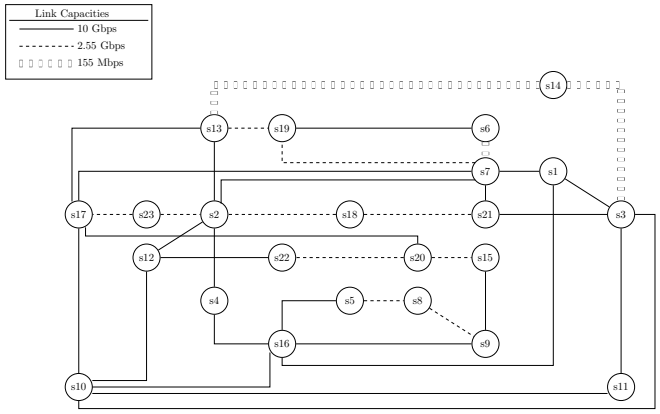


Fig. 5. Topology of the network GEANT with 23 nodes.

the choice between 10 other data planes without the likelihood being altered. To note that the performance of the function $i_{Conf}(\mathcal{P})$ is proper to the observation issue considered while the limit TD permits to define the scope of the possible unknown observation tolerated. This setup is now used to analyze the likelihood computation and, more generally speaking, the complexity of the network observability.

5.2 Analysis

Fig. 6 shows the evolution of the likelihood score $\mathcal{L}'(\mathcal{P})$ computed by the HMM. The HMM is configured with a depth of the considered sequence (the number of elements in a sequence) fixed to 3 (comprise between accuracy and complexity given that attacks are detected for depth > 2). The first curve, named "Nominal", represents the evolution of the likelihood score without any observation issue. The two other curves represent the likelihood score depending if the function used is Min or $Average$.

Consider first the $Average$ operator. At the observation number 42, it can be seen on Fig. 6 that the likelihood score using $Average$ is larger than the likelihood score of the nominal data plane (the one installed by the controller). This nominal data plane can be one of the rarest possibilities and so considering the $Average$ permits to consider all the possible data planes and so increases the likelihood score. Such operator might then overrate the likelihood score of the observation. There is indeed a threat of retaining an abnormal data plane and to raise false alarms but also false negative.

Secondly, Fig. 6 shows that the likelihood score calculated by the $Minimum$ operator remains lower than the nominal case. Here, the worst case is considered and it means that the likelihood of the installed data plane is equal or upper and so we are not exposed to a false negative. It has to be reminded some assumptions, in particular that the packet loss is due to observability issues and that the packets not captured by the observer corresponds to a consistent data plane. Considering the worst case is a way to minimize the potential to interfere with the detection of a control anomaly and as a consequence, it improves the robustness of the detection. To note also that such behaviour is conservative and there might be more false alarms considering the $Minimum$ operator rather than the $Average$.

6. CONCLUSION

In conclusion, the aim of this work is to study the limits of the detection approach based on the assumption that the observer has no observability issue. However, such assumptions are not always satisfied in a network context. This paper introduces an extension of the detection approach to improve the robustness in case of parsimonious observation. In particular, we showed that the evaluation of the likelihood of a (potential) incomplete data plane needs to be adapted and that such plane needs to be compared to the minimal likelihood of the non-faulty complete data planes.

The perspective of this work is to study the sensitivity of the method proposed depending on the number of packets missing. Also, we would like to extend the approach to the case of encrypted communication.

ACKNOWLEDGEMENTS

This work was supported partly by the French PIA project "Lorraine Université d'Excellence", reference ANR-15-IDEX-04-LUE.

REFERENCES

Amin, R., Reisslein, M., and Shah, N. (2018). Hybrid sdn networks: A survey of existing approaches. *IEEE Communications Surveys & Tutorials*, 20(4), 3259–3306.

Baum, L.E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6), 1554–1563.

Casas-Velasco, D.M., Rendon, O.M.C., and da Fonseca, N.L. (2020). Intelligent routing based on reinforcement learning for software-defined networking. *IEEE Transactions on Network and Service Management*, 18(1).

Castillo, E., Conejo, A.J., Menéndez, J.M., and Jiménez, P. (2008). The observability problem in traffic network models. *Computer-Aided Civil and Infrastructure Engineering*, 23(3), 208–222.

Desgeorges, L., Georges, J.P., and Divoux, T. (2021). A technique to monitor threats in sdn data plane computation. In *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*.

Desgeorges, L., Georges, J.P., and Divoux, T. (2022). Anomalies detection method for non-determinist sdn control. *IFAC-PapersOnLine*, 55(8), 57–63.

Dvir, A., Haddad, Y., and Zilberman, A. (2019). The controller placement problem for wireless sdn. *Wireless Networks*, 25, 4963–4978.

Farhady, H., Lee, H., and Nakao, A. (2015). Software-defined networking: A survey. *Computer Networks*, 81, 79–95.

Fonseca, P., Bennesby, R., Mota, E., and Passito, A. (2012). A replication component for resilient openflow-based networking. In *2012 IEEE Network operations and management symposium*, 933–939. IEEE.

Kreutz, D., Ramos, F.M., and Verissimo, P. (2013). Towards secure and dependable software-defined networks. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, 55–60.

Kreutz, D., Ramos, F.M., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S., and Uhlig, S. (2014). Software-

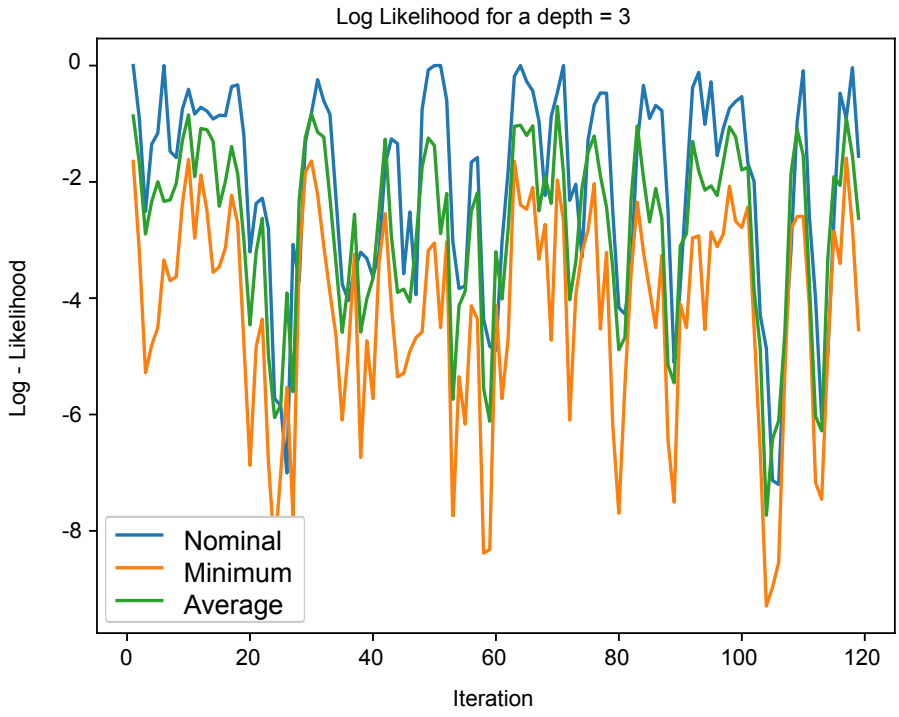


Fig. 6. Inference process

defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1), 14–76.

Ku, I., Lu, Y., and Gerla, M. (2014). Software-defined mobile cloud: Architecture, services and use cases. In *2014 international wireless communications and mobile computing conference (IWCMC)*, 1–6. IEEE.

Lee, S., Yoon, C., Lee, C., Shin, S., Yegneswaran, V., and Porras, P.A. (2017). Delta: A security assessment framework for software-defined networks. In *NDSS*.

Montanari, A.N. and Aguirre, L.A. (2020). Observability of network systems: A critical review of recent results. *Journal of Control, Automation and Electrical Systems*, 31(6), 1348–1374.

ONF (June, 2012). *OpenFlow Specification v1.3 https://opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf, last visited the 14/09/2021.*

Petit, D., Georges, J.P., Divoux, T., Regnier, B., and Miramont, P. (2016). Freshness analysis of functional sequences in launchers. *IFAC-PapersOnLine*, 49(30), 80–85.

Scott-Hayward, S., O'Callaghan, G., and Sezer, S. (2013). Sdn security: A survey. In *2013 IEEE SDN For Future Networks and Services (SDN4FNS)*, 1–7. IEEE.

Sinha, Y., Haribabu, K., et al. (2017). A survey: Hybrid sdn. *Journal of Network and Computer Applications*, 100, 35–55.