IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727
PP 46-52
www.iosrjournals.org

Enhanced Malware Monitor in SDN using Kinetic Controller

Jiphi T S, Simi Krishna K R
Department of Information Technology, Government Engineering College, Barton Hill, India
Department of Information Technology, Government Engineering College, Barton Hill, India

Abstract: *Software Defined Networking introduced significant granularity, visibility and flexibility to networking, but at the same time brought forth many security challenges. So, many of the security measures have to be brought to SDN which are used in traditional networks. One such concept is Malware Monitor, which combines the advantage of a distributed detection performing NIDS with SDN. It allows for integration into existing hardware. It is faster and more flexible in threat blocking. But one of the fundamental challenges is to build and deploy effective detection systems in highly dynamic environments where maintaining host and network state information is very difficult. The objective of this work is to formulate a mechanism to build an effective system that can cope with highly dynamic environments. This will ensure high security in those networks.*

Keywords: *Malware, Openflow, Programmable Networks, SDN*

I. Introduction

Software Defined Networking (SDN) has quickly emerged as a promising technology for future networks. It introduced significant granularity, visibility and flexibility to networking, but at the same time brought forth many security challenges. Many legacy security appliances such as Firewalls and Intrusion Detection and Prevention Systems (IDS/IPS) have been migrated to these OpenFlow-based networks by re-designing and implementing these systems as compatible security applications. Large networks such as enterprises and campuses have increasingly come to adopt Network Intrusion Detection Systems (NIDS) for mitigating malware. A framework called Malware Monitor is available which act as a comprehensive, distributed malware detection system for networks, but is less efficient in dynamic network conditions. The purpose of this work is to develop a mechanism to build an effective detection system in highly dynamic environments.

II. Motivation

With the rapid growth of SDN, a larger number of customizations can be done at data plane level and control plane level. Policies can be introduced or updated dynamically for handling real time events. Some policy changes are automated to adapt to the network condition changes and this might cause the malware monitor to work less efficiently. To address the above issue this paper takes into account the features of kinetic controller which inherits Pyretic's language [8] and runtime and also provides a verifiable dynamic control. Kinetic provides a structured language for expressing a network policy in terms of finite state machines (FSMs), which both concisely capture dynamics and are amenable to verification. Kinetic's use of computation tree logic (CTL) and its ability to automatically verify policies with the NuSMV model checker can allow network operators to verify the dynamic behavior of the controller before the control programs are ever run. And it would be beneficial to include the feature of kinetic in malware monitor.

III. Literature Survey

i. Software Defined Networking

SDN [2] is defined as new approach to designing, building, and managing networks that separates the network's control (brains) and forwarding (muscle) planes to better optimize each. The basic characteristics of SDN are Plane Separation, Centralized Control, Network Automation, Virtualization and Openness. Basic components of SDN are SDN devices, SDN controller and SDN applications.

The SDN devices contain data, represented by flows, which drive forwarding decisions and a forward functionality that decides what do to with the received packet. The flow table in the device has a series of flow entries and details of actions to perform when a packet matching that flow is received at the device. When a packet arrives at the device, it searches the flow table for a match and performs the corresponding action.

The SDN controller is the brain of the SDN network. It abstracts the network of SDN devices that it controls and presents an abstraction of those resources to SDN applications that it interfaces with. The controller

also allows the applications to define flows on devices and to respond to packets sent to controller from SDN devices.

The SDN applications interface with SDN controller. These are used to define flows on the devices and to respond to packets received by the controller. There are proactive as well as reactive flows. Proactive flows can be predefined static flows or flows modified based on network traffic load through the device during a certain period. Reactive flows are defined in response to a packet received at the controller or in response to other factors such as IDS.

ii. Malware Monitor

SDN security is an area that requires further development. Large networks such as enterprises and campuses have increasingly come to adopt Network Intrusion Detection Systems (NIDS) for mitigating malware. It involves Deep Packet Inspection (DPI). Centralized control together with DPI is incapable of coping with high data rates and also lacks in effective and speedy threat detection and mitigation. The solution is Malware Monitor.

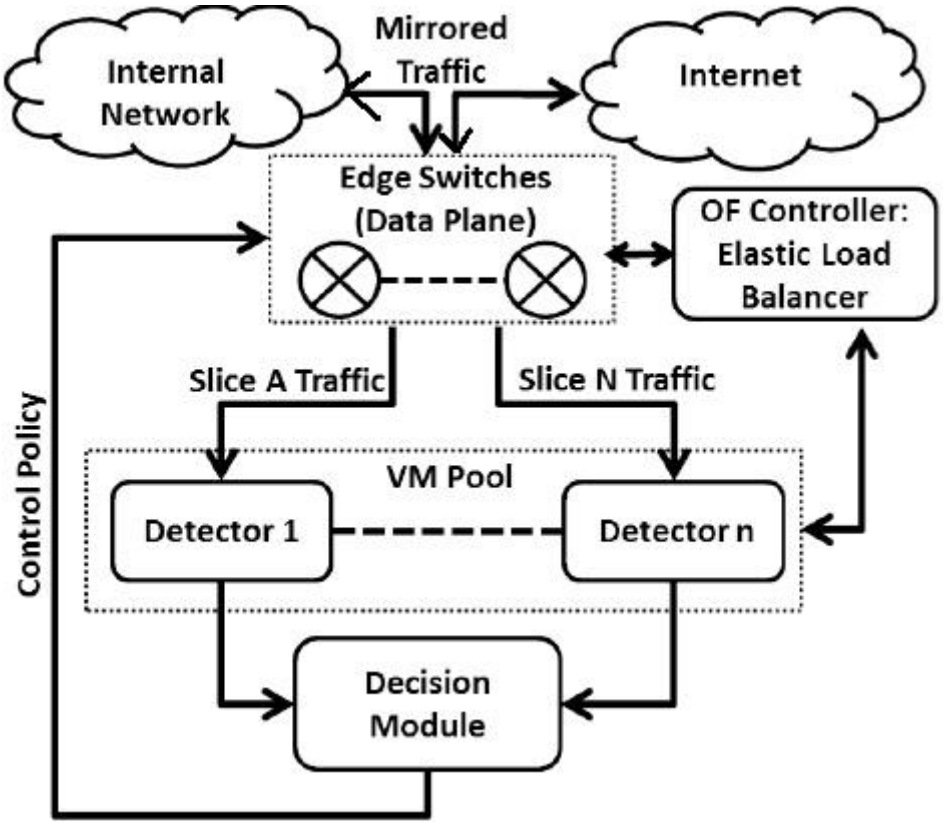

Fig.1 Architecture of Malware Monitor [1]

Malware Monitor [1] is an SDN based conceptual framework for securing large networks. It proposes elastically portioning network traffic to enable distribution of detection load across a range of detectors; further, a centralized SDN controller allows for network wide threat correlation as well as speedy control of malicious flows.

Malware Monitor Modules include:

1. Elastic Load Balancer: The load balancer is an SDN application running on an Open Flow controller. It consists of a virtual machine (VM) controller and network slicing logic. Depending on network load, the VM controller issues n number of logic detection machines (detector nodes) from its VM pool, thus ensuring elasticity. The network slicing logic will then divide the network into n slices, where a slice is all traffic with a particular characteristic. Then a copy of the packets from each slice is forwarded to the corresponding detector nodes by creating appropriate flow rules.
2. Detector Nodes: Each detector node has two main functions. First, the inspection of received packets to detect malware infections and second, the computation of flow statistics of received flows. The first function can be achieved by using any of the tools available for detection of malicious events. The second

function can be achieved by running the detector node on an Open Flow (OF) controller. In an OF, the switches maintain flow statistics for each flow passed through them. So, the node can easily pull the flow statistics from the switch. The detector node then merges the flow identification information with any detected malicious activity and the computed flow statistics, and forwards it to the decision module.

3. Decision Module: The decision module, after examining the flow data received from the detector nodes, computes a maliciousness score for each flow. It analyzes the connections between the flows to identify suspicious patterns or combinations of events that imply a malicious attack. Thus the score of a flow is allotted based on not only the events associated with it, but also on its similarity with other malicious flows. This helps in effectively identifying those malicious flows that are undetectable individually.

iii. Flowguard

Flowguard [4] is a comprehensive framework used to facilitate accurate detection and effective resolution of firewall policy violations in dynamic OpenFlow-based networks. It checks network flow path spaces to detect firewall policy violations when network states are updated. In addition, it also conducts automatic and real-time violation resolutions with the help of several innovative resolution strategies designed for diverse network update situations.

iv. Kinetic Controller

Kinetic [3] is a domain specific language and SDN controller for implementing dynamic network policies in a concise, verifiable language that can handle continually changing network conditions, adapt the network configuration whenever these conditions change, and also provides a means to verify that the changes will be correct. Kinetic exposes a language that allows operators to express network policy in an intuitive language that maps directly to a CTL-based model checker.

IV. Problems

The following are the problems/challenges associated with the Malware Monitor framework.

1. There is no way to devise a formal mechanism to decide how much load a single detector VM can take and when a new one should be spawned. (This is out of scope of this work)
2. It is difficult to maintain host and network state information at the detector and decision modules in highly dynamic environments such as a campus network, and deploy effective detection algorithms. There should be an effective mechanism for handling the highly dynamic policies in the SDN and for verifying the correctness of the dynamic control.

V. Solutions

Improve detection mechanism by proper and correct handling of dynamic network changes to effectively work in dynamic environments. The main challenge of dynamic networks is the discovery and resolution of flow policy violations as the network policies are updated frequently.

1. There is a framework called FlowGuard [4] that addresses this issue in firewalls. One solution is to use a similar approach to introduce an improved detection mechanism in Malware Monitor.
2. Another approach is to use kinetic [3] which has a verifiable dynamic network control, a feature that other controllers don't have [5]. Kinetic provides a structured language for expressing a network policy in terms of finite state machines (FSMs), which both concisely capture dynamics and are amenable to verification. States correspond to distinct forwarding behavior, and events trigger transitions between states. Kinetic's event handler listens to events and triggers transitions in policy, which in turn update the data plane. (In this paper the focus will be on this method)

VI. Proposed System

This paper proposes a malware monitor which uses kinetic controller instead of pox controller or floodlight controller. Kinetic is a domain specific language (DSL) and SDN controller that enables creation of network control programs that are capable of capturing responses to dynamic network conditions. Using kinetic, network policies can be expressed in terms of finite state machines (FSMs) which capture dynamics accurately and also allows easy verification of those policies. Kinetic evaluates the correctness of policies by using NuSMV model checker. When a Kinetic program is executed, it automatically creates an FSM model for the model checker. The proposed system, thus, not only increases the ability of malware monitor to respond effectively to dynamically changing network and its policies, but also allows to verify those changes to ensure that those are in line with the requirements.

The Detector Nodes perform deep packet inspection for identifying malwares. It can be done by using snort IDS rulesets of BotHunter [10] or by using layered framework of trained neural network.

The Decision Module detects distributed attacks through the overall analysis of five tuples and alert messages send by the detector nodes. Provenance-aware Security Risk Analysis approach [9] can be used. In Fig.2 there are two controllers, one for managing and assigning of the detector nodes from Virtual Machine pool as well as changing the policies in the network to make proper slicing of traffic to the detector nodes and other for managing the network policies based on the authentication and intrusion detection system events. In this paper, the elastic load balancer and the OF controller associated with it is not considered. If considered it is preferable to use kinetic controller. Both the controllers have to synchronize their policies which are updated on the network, to detect the network policy violations and conflicts brought by new changes. It can also be implemented using one kinetic controller with decision module and load balancer application thereby avoiding complexity of synchronizing policies, that is, controller to controller policy synchronization methods are not necessary. But it might slightly impact the performance of controller. Here we are only considering the decision module which decides on policies and sends json event messages to the kinetic controller to set those policies.

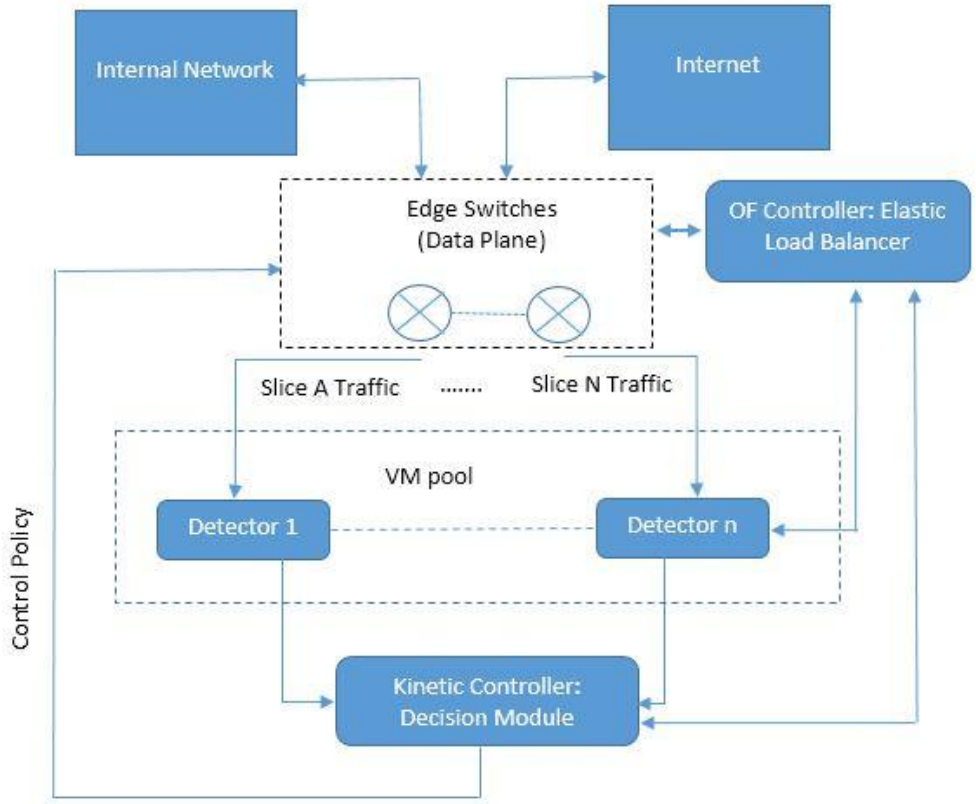

Fig.2 Architecture of Proposed System [6]

VII. Evaluation

The proposed system has been implemented in a simulated environment by using the kinetic controller and mininet [7] configured on an ubuntu server virtual machine. Pox and Pyretic Controllers were used to compare flow rule update time with kinetic controller.

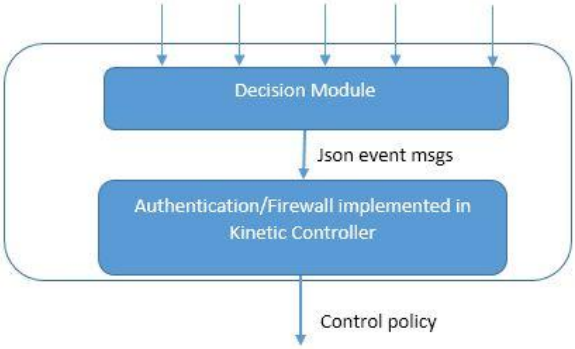

Fig.3 Modules implemented in Kinetic Controller [6]

The decision module takes decision based on the output of the detector nodes, that is, based on flow tuple, alert messages and flow stats given by detector nodes. For the test, static capture flow outputs of detector nodes were given as input to the decision module. The decision module will cross check the different flow tuples, that is, the output of different detector nodes, for distributed type of attacks, and for distributed attack pattern which are not detected by the detector nodes. The decision module then calculates the malicious score based on these criteria. Based on this malicious score json event messages are send to the kinetic controller for updating intrusion detection system policies and authentication policies.

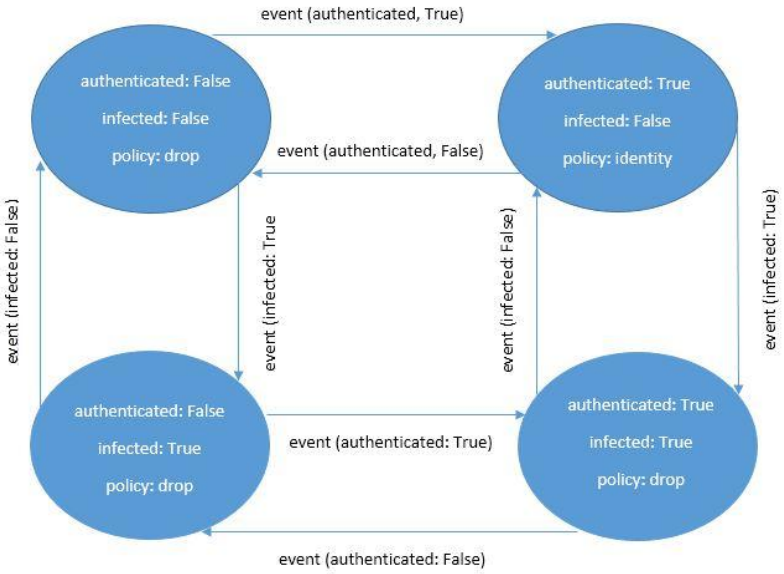

Fig.4 State representation of authentication and intrusion detection system [6]

In this simulation, kinetic controller has two state variables, Authenticated and Infected. The policies are set based on the value of these Boolean state variables. All the hosts and open flow switches will have a corresponding finite state machine which has number of states limited by Located Packet Equivalent Classes to prevent state explosion. When network boots up, all the devices will have both the state variables, Authenticated and Infected, set to False. To enable flow of a device in the network, it has to be authenticated by the kinetic controller. Once authenticated or the device is allowed to use the flow space of the network, then the policy is set to Identity. When an infected event occurs, the state of that device is changed from infected False to True and policy to Drop. For those devices which are of universal blacklisted IPs or have high probability of compromising the network based on the outcome of decision module, the state variable Authenticated is set to False and Infected to True. The simulation was conducted with different number of hosts and different type of topologies namely single, linear and tree. For the different topologies, flow rule update time increases with the increase in the number of hosts in a linear fashion.

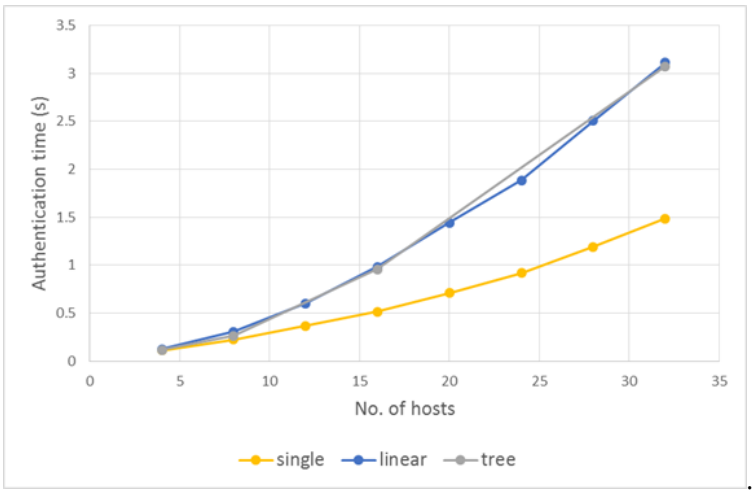

Fig.5 Authentication flow rule update time in Kinetic Controller based on topology

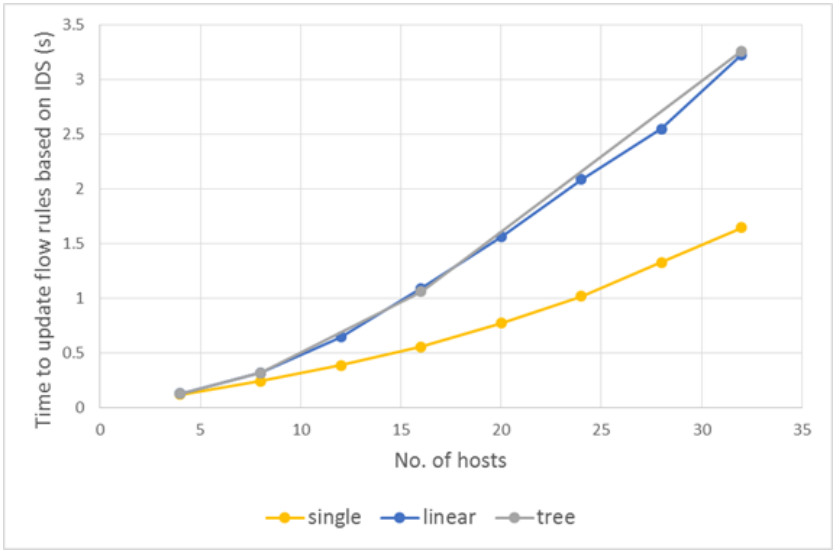

Fig.6 IDS flow rule update time in Kinetic Controller based on topology

The simulation was conducted with pox controller and pyretic controller to compare their flow rule update time. For different SDN controllers, there is not much variation in flow rule update time.

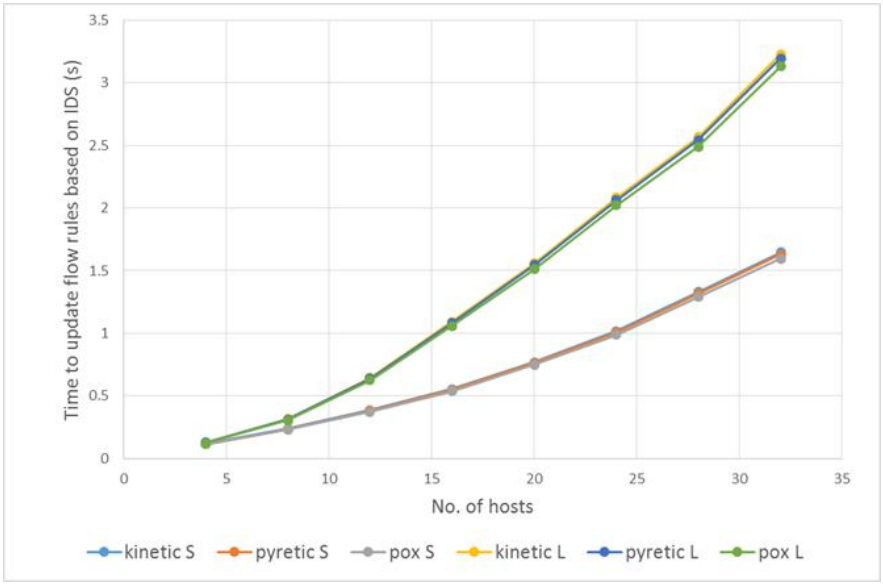

Fig.7 IDS flow rule update time in different SDN controllers.

VIII. Conclusion and Future Work

Kinetic Controller helps to mitigate network policy violations and conflicts better compared to other SDN controllers like pox or pyretic. Introduction of Kinetic Controller in the malware monitor will make it effective in handling threats in ever changing dynamic environments. Enhanced malware monitor using kinetic controller, can handle the dynamic policy changes that occur continuously and provide a mechanism that can verify whether the system can adapt to the network condition changes to produce the exact behavior as intended.

The future scope of this work is to analyze the proposed system with complex networks governed by different SDN applications communicating through northbound API. Another idea is to evaluate its performance in large networks with huge amount of policies arriving newly and changing continuously based on automated SDN applications. Furthermore, a mechanism to automate the resolution of the policy violations and conflicts, or a recommender system for recommending new network policies which would not conflict with other existing policies, can be implemented. Another area for future work is the implementation of a self learning neural network for the decision module.

References

[1] Zainab Abaid, Mohsen Rezvani and Sanjay Jha, MalwareMonitor: An SDN-based Framework for Securing Large Networks, *CoNext Student Workshop'14, 2(12), 2014.*

[2] Chapter 4 - How SDN works, *2014 Elsevier Inc*.

[3] Hyojoon Kim, Joshua Reich, Arpit Gupta, Muhammad Shahbaz, Nick Feamster and Russ Clark, Kinetic: Verifiable Dynamic Network Control, *12th USENIX Symposium on Networked Systems Design and Implementation*, 2015, 59-72.

[4] Hongxin Hu, Wonkyu Han, Gail-Joon Ahn, and Ziming Zhao, FLOWGUARD: Building Robust Firewalls for Software-Defined Networks, *HotSDN'14, 22(8),* 2014.

[5] Jiphi T S and Simi Krishna K R, A Survey on SDN Controllers, *Proc. 17th National Conference on Technological Trends*, Trivandrum, Kerala, 2016, 863-866.

[6] Jiphi T S and Simi Krishna K R, Enhanced Malware Monitor in SDN, *Proc. 5th National Conference on Emerging Technologies*, Trivandrum, Kerala, 2016, paper id: 62.

[7] mininet, [online]. Available: http://mininet.org/walkthrough

[8] pyretic [online]. Available: http://www.frenetic-lang.org/pyretic

[9] Mohsen Rezvani, Aleksandar Ignjatovic, Elisa Bertino and Sanjay Jha, Provenance-aware security risk analysis for hosts and network flows, *Network Operations and Management Symposium (NOMS)*, 2014 IEEE, 1-8.

[10] Guofei Gu, Phillip A Porras, Vinod Yegneswaran, Martin W Fong, and Wenke Lee, BotHunter: Detecting malware infection through IDS-driven dialog correlation, *USENIX Security*, 2007, 1-16.