As-RaD System as a Design Model of the Network Automation Configuration System Based on the REST-API and Django Framework

by Adian Fatchur Rochim

As-RaD System as a Design Model of the Network Automation Configuration System Based on the REST-API and Django Framework

Article Info
Keywords:
Network Automation, REST API, Django, Software Defined Network

Article history:
Received 17 August 2018
Revised 15 February 2019
Accepted 4 April 2019
Available online 4 April 2019

Cite:
Rochim, A. F., Hamaminata, A. R., Fauzi, A.. & Martono K. T. (2020). As-RaD System as a Design Model of the Network Automation Configuration System Based on the REST-API and Django Framework. Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 4(3). doi:

* Corresponding author.
Adian Fatchur Rochim
E-mail address:
adian@ce.undip.ac.id.

Abstract
The usage of information technology became very high because of COVID-19 pandemic. From business through education activities tend to use this technology most of the time. Information technology uses computer networks for integration and management data. To avoid communication problems, the number of network devices installed requires a manageable network configuration for easier maintenance. Traditionally, each of network devices has to be manually configured by network administrators. This process takes time and inefficient. Network automation configuration system exist to overcome the repetitive process. Unfortunately, network automation methods are relatively slow. In this research, we propose an alternatif model of network automation system. The model system implemented with a controller application that used REST API (Representational State Transfer Application Programming Interface) architecture and built by Django framework with Python programming language to increase the performance of network automation. Design model uses a web-based application for maintenance and automates networking tasks wih easy GUI. The design model namely As-RaD System. The network devices used in this research are Cisco CSR1000V because it supports REST API communication to manage its network configuration and could be placed on the server either. The As-RaD System provides 75% faster performance than Paramiko and 92% than NAPALM.

1. Introduction

In this era, computer networks have become dynamic and complex [1]. The availability and reliability of network devices then become a challenge for computer network providers. To configure network devices, network engineers use well-known tool as secured shell (SSH). by manually is ineffective because it is consuming a lot of time for configuring each device. Repetitive tasks, i.e. login and logout, entry user and password are done for every device that takes a time consume.

To reduce time, repetitive work of operation network maintenances used network automation by application programming interface (API) [2]. It can be overcome by automating computer networks. The tasks include monitoring network to prevent vulnerability network [3]. The automation network also to modified static routing and dynamic routing. In the other hand it also use to configure users [4]. Therefore, we can say that the network automation is a method that uses programming logic to manage network devices, so that network administrators can make an automatically configure network devices [5].

Network automation (NA) is done using Python programming language [6]. Paramiko and NAPAL are implementation of the NA concept that was coded by the Python language. Paramiko is a Python implementation library of SSH protocol and could providing network automation [7]. Then, NAPALM or Network Automation and Programmability Abstraction Layer with Multivendor support are a Python library that implements a set of functions to interact with different router vendor devices using a unified API [6]. The REST API (Representational State Transfer Application Programming Interface) has recently become popular in the design of a network protocol [8]. REST in the development of computer networks is an architecture that allows applications to send configurations to other applications, which in this case are virtual computer network devices [9]. To enable network automation, the creation of Python script is still needed, thus, to improve network administration it is necessary to develop web-based applications that have a display or GUI and can be accessed centrally [10].

Rheza et al., in 2014 explained how an application could be designed using the Python programming language to automate network device administration such as routing, and backup restore device configuration. This approach could reduce the complicated and repetitive tasks of a network administrator [11]. Zhou et al., in 2014 explained various issues regarding the RESTful protocol for computer network design needed with programming approaches and how the HTTP protocol can be used to control a computer network device with the advantages of RESTful [8]. Mihaila, in 2017, made comparative automation comparisons using several methods between NAPALM, Netmiko and Paramiko methods [12]. He demonstrated the methods to configure network devices by using each method. However, the comparison of the time requirements of each method required has not been disclosed, although how to implement network automation in a framework.

So that the research questions arise are as follows: 1) how to create a framework for network automation; 2) how to compare the performance of the proposed framework with existing methods i.e., NAPALM and Paramiko. both methods take a long time for access on network devices [13][14]. Focus of the research question is time consume of each method to access the network devices. This research question is important, because, longer requirement time it takes to execute one command on a network device, it will have an impact on the accumulation of subsequent commands for other devices in a network infrastructure. So, it has an impact on the whole institution that is currently using the network infrastructure.

Through this paper, authors proposed: 1) building an application to control network configuration with Django framework; 2) comparison of performance testing network automation framework based on time requirements.

There are four sections in this paper, the first section describes the background, purpose of network automation using REST API and previous research. Next section describes the methodology used. The third section describes result and discussion of the system implementation. And finally, conclusion is described in fourth section.

2. Research Method

The research starts from the study of literature from several journals and books as a source of research references. To develop the network automation uses the Design Science Research (DSR) methodology [15] which consists of system definition, system specifications, system configuration, evaluation and results.

First step is to define the system. This step includes identification of system requirements, the goal and benefit of the system, how the system works, and the network topology used. Next step identifies the requirement specification. This determines the specification of the requirements that match the system definition. Third step is system configuration. Basically, this step is aimed to design specifications for predetermined requirements. These requirements cover the network topology or design that is applied as a whole system or system unit that makes the system run.

And final step is to evaluate the system implementation and makes performance comparison with other methods to get the percentage of effectiveness of the REST API method. The proposed for a design model is namely the network automation system based on Rest-API and Django framework (As-RaD System). Figure 1 illustrate each steps of the research methodology.

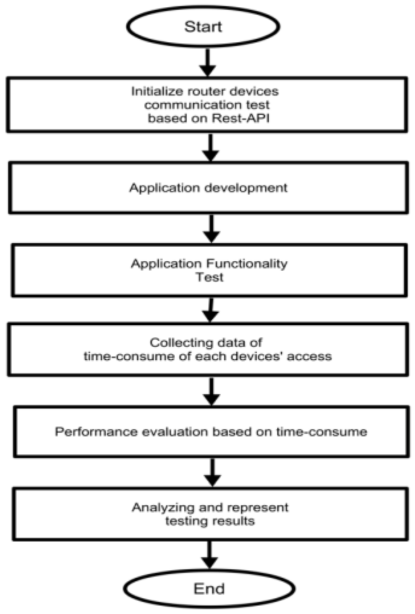

Start
Initialize router devices communication test based on Rest-API
Application development
Application Functionality Test
Collecting data of time-consume of each devices' access
Performance evaluation based on time-consume
Analyzing and represent testing results
End

Figure 1 Flowchart System

2.1 Functional and Non-functional Requirements

The functional requirements of the system are controller that could automate common network management such as adding IP Address, static route, dynamic route (OSPF and BGP), custom configuration, exporting syslog, and validating the configuration. While the non-functional requirement of the network automation system is a controller in a form of an application system that has a user-friendly and responsive interface, it can be run on several types of browsers and operating systems either.

2.2 Hardware Requirements

Hardware requirement is used to run the network automation controller. The controller is implemented on a server by cloud provider as an instance. Table 1 shows the specification of the instance.

Table 1 Instance Specification

Component	**Server**
Instance Type	Amazon EC2 T2 Micro
CPU	1 x Intel Xeon vCPU 2.20 Ghz
RAM	1024MB DDR4
Storage	1 GB Amazon EBS

2.3 Software Requirements

The As-RaD System requires several software tools to build application controller and other components. IOS XE as the operating system of the Router used to support REST API architecture communication [16]. Network automation controller application is made to implement As-RaD System in a tested model. The application is developed using the Python programming language and Django framework [17]. Django framework uses the MVT (Model View Template) to build a web application faster because it has several built in templates [18].

The other software used i.e. Ubuntu as a server, Python 3.7.5 version used in application development, Django 3.0.3 as web framework, AWS Cloud [19][20] as the cloud system, and Pop OS 19.10 based on Debian Linux as the operating system used in system development.

Three units of Cisco CSR1000V Routers [16] as the devices connect to the network automation system. One laptop to configure the system and create the application.

2.4 Topology Design

The network device is used with cloud-based virtual computing that supports the scenario for testing the model. The test includes automatic configuration features and validation from the topology created. Figure 2 shows the topology used for this research.

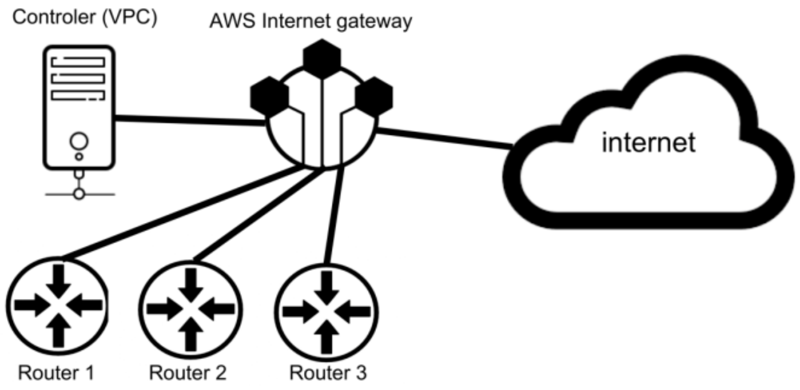


Figure 2 Network Topology

The VPC (Virtual Private Cloud) [21] is used as a local network area in the cloud system to put the components that will be tested. These components include the network automation controller, three Cisco CSR1000V Routers, and an Internet Gateway. Figure 3 illustrates the As-RaD system design.

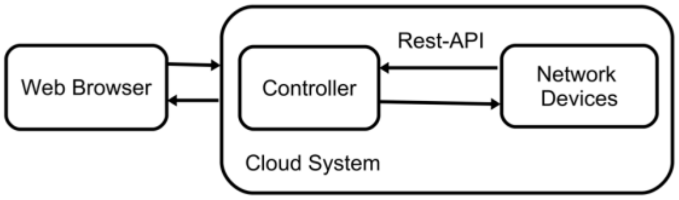


Figure 3 Network Automation System Design

3. Results and Discussion

This chapter discusses in detail about system evaluation such as the communication testing with the device, the implementation of the controller created, the testing of the application system of a controller, the comparison of the As-RaD System with other methods, and the discussion of application development of the controller.

Details of the As-RaD network automation process to communicate and configure the devices are presented as a flowchart in Figure 4.

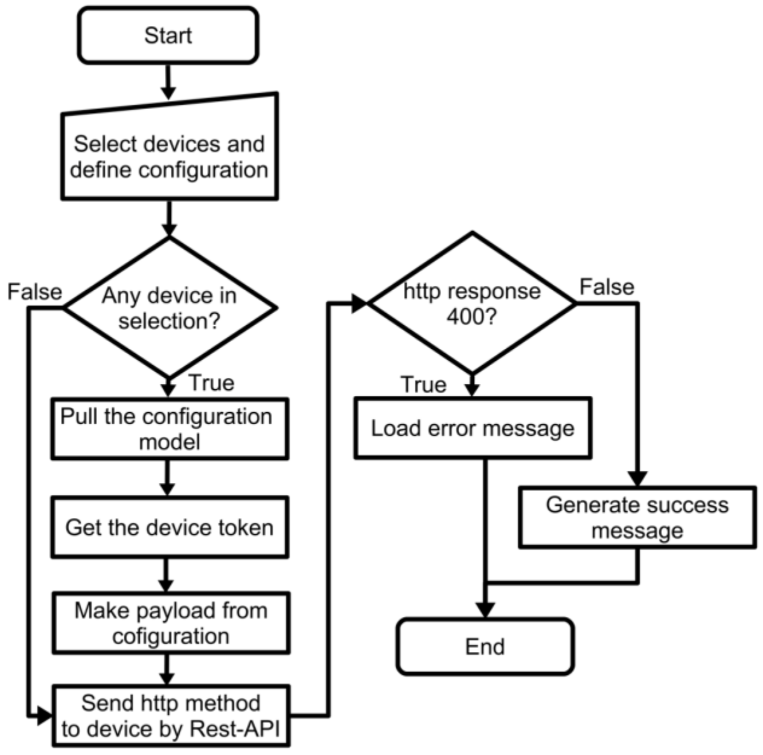

Start
Select devices and define configuration
Any device in selection?
False
True
Pull the configuration model
Get the device token
Make payload from cofiguration
Send http method to device by Rest-API
http response 400?
False
True
Load error message
Generate success message
End

Figure 4 Network Automation Flow Chart

The following is the pseudocode of the As-Rad System network automation process:
Input: configuration parameter
Output: configured device

```
1: select devices to configure
2: input configuration parameter
3: for each device in selected devices do
4:     pull the configuration model
5:     get token of the device
6:     make payload from the configuration model
7:     send payload to devices through http methods
8:     if devices http response > 400
9:         load error message
10:    else
11:        generate success message
12:                endif
13: endfor
```

3.1. REST API Evaluation

Using cURL [22] An HTTP Method request is tested for each API endpoint [23] to make sure the connection established using the REST API architecture with a Cisco CSR1000V Router. Table 2 shows communication of each request made. Each HTTP method is sent with JSON Properties according to the parameters requested by the client [24].

Table 2 REST API Requests

Configuration	HTTP Method	Endpoint	Response
Add IP Address	PUT	https://ip_address:55443/api/v1/interfaces/	204 No Content
Static Route	POST	https://ip_address:55443/api/v1/routing-svc/static-routes	201 Created
OSPF Route	POST	https://ip_address:55443/api/v1/routing-svc/ospf/ospd_id/networks	201 Created
BGP Route	POST	https://ip_address:55443/api/v1/routing-svc/bgp/bgp_id/networks	201 Created
Validate Configuration	PUT	https://ip_address:55443/api/v1/global/cli	200 OK
Export Syslog	GET	https://ip_address:55443/api/v1/global/syslog	200 OK
Custom Configuration	PUT	https://ip_address:55443/api/v1/global/cli	200 OK

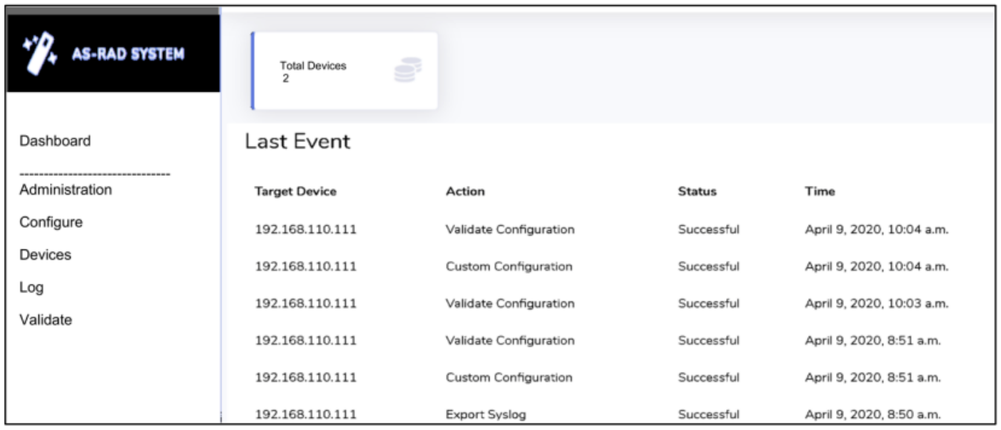
3.2. Home Dashboard Evaluation

Home Dashboard displays devices that were registered to the controller. This dashboard also displays the last event that occurred in the controller application such as the time of the automation, the target device's IP (Internet Protocol) address, automation status, and the user who did the automation.

Figure 5 shows the results of the Home Dashboard test which shows the controller doing an activity/event in the form of an experimental configuration of the registered device. The information obtained shows the success of network automation which is carried out.

3.3. Configuration Menu Evaluation

The Configuration Menu offers options for performing network automation actions on network devices such as Add or Update IP Addresses, Static Routes, Dynamic Routes, and Custom Configuration. Users are given a form to fill in the parameters of each configuration they need. For each configuration has its own parameters (i.e., updating IP address on the registered devices could be done simultaneously applies to all devices with each of their own configuration contained in this menu).



Last Event

Target Device	Action	Status	Time
192.168.110.111	Validate Configuration	Successful	April 9, 2020, 10:04 a.m.
192.168.110.111	Custom Configuration	Successful	April 9, 2020, 10:04 a.m.
192.168.110.111	Validate Configuration	Successful	April 9, 2020, 10:03 a.m.
192.168.110.111	Validate Configuration	Successful	April 9, 2020, 8:51 a.m.
192.168.110.111	Custom Configuration	Successful	April 9, 2020, 8:51 a.m.
192.168.110.111	Export Syslog	Successful	April 9, 2020, 8:50 a.m.

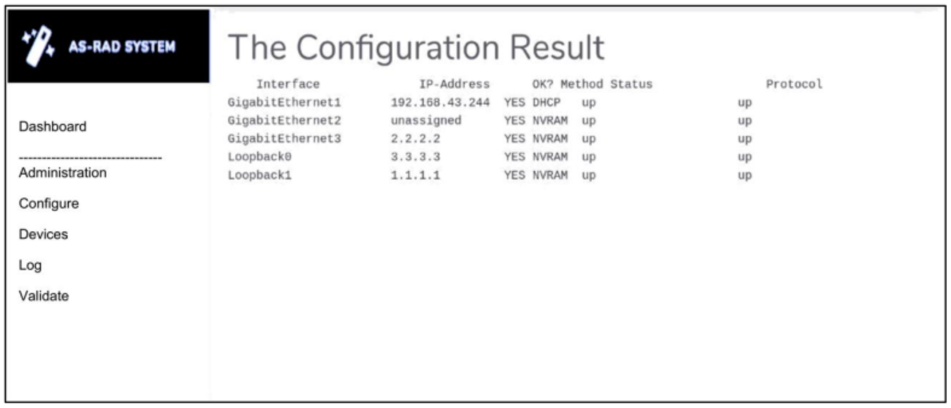
Figure 5 Home Dashboard

3.4. Log Menu Evaluation

This Log Menu offers users to view the activities of the controller application which is configuration information. Besides, there is a menu for exporting Syslog generated by network devices. The detailed log shown information related to the configuration made by the user so that the configuration has occurred.

3.4. Application Evaluation

Application evaluation explains the results and workflow of the controller application that has been developed. The initial work process contained in the application is the login page where the user is authenticated to enter the system. After being authenticated, users will be redirected to the main page that contains the Dashboard Home. On the home dashboard, there is the last event that occurs in the application. This page also has a navbar that presents options to configure, view devices, log, and configure validation.



The Configuration Result

```
    Interface            IP-Address      OK? Method Status                Protocol
GigabitEthernet1       192.168.43.244  YES DHCP   up                    up
GigabitEthernet2       unassigned      YES NVRAM  up                    up
GigabitEthernet3       2.2.2.2         YES NVRAM  up                    up
Loopback0              3.3.3.3         YES NVRAM  up                    up
Loopback1              1.1.1.1         YES NVRAM  up                    up
```

Figure 6 Validation Menu

The configuration validation menu displays the results of the configuration applied to network devices by entering command-line commands so that the configuration is applied accordingly. Figure 6 shows the configuration results in the configuration validation menu. Figure 6 shows that the device has been successfully configured with the information carried that indicates the running configuration on the network device.

3.5. Performance Evaluation

To perform a performance evaluation test of As-RaD system, a comparison of the execution time scale taken for each method in implementing network automation. The two methods applied in the previous research (Paramiko and NAPALM) were taken to provide comparative variables with the methods applied. Data collected from the execution time to complete network automation in the time scale of the millisecond using As-RaD System, Paramiko, and NAPALM. Retrieval of data is to determine the performance of the methods applied.

The test scenario is to add an IP address to the loopback interface and be applied to the same topology. From testing 100 times, the dataset obtained was filtered using the Peirce classification to eliminate data outliers that affect the logging of actual performance [25].

After having datasets, the next step is to compare the performance of each method. The first method compared to As-RaD System is Paramiko. From the results of obtained execution time data, visualization of Figure 7 shows the comparison in the form of a graph.

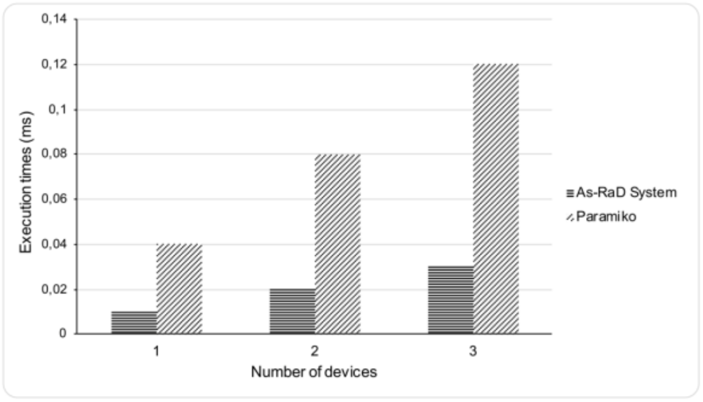


Figure 7 As-RaD System vs Paramiko

The data visualization shown in Figure 7 shows that the As-RaD System has ability to complete execution time faster than Paramiko with the various condition of the number of devices. This gives a marker that As-RaD has better performance than Paramiko.

Furthermore, As-RaD System is compared with other methods, namely NAPALM. From the results of the execution time obtained a comparison as in the data visualization shown in Figure 8.

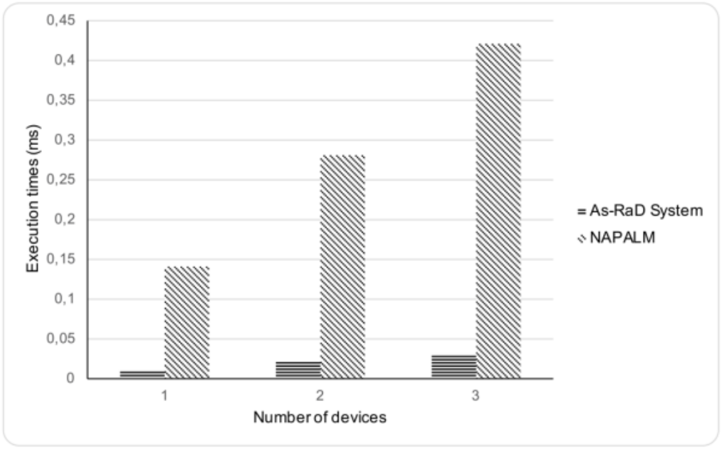


Figure 8 As-RaD System vs NAPALM

The data visualization shown in Figure 8 shows that the As-RaD System has the ability to complete execution time faster than NAPALM. Even the execution time of As-RaD System is relatively far superior compared to the previous testing. Finally, Figure 7 displays the comparison of the three methods tested.

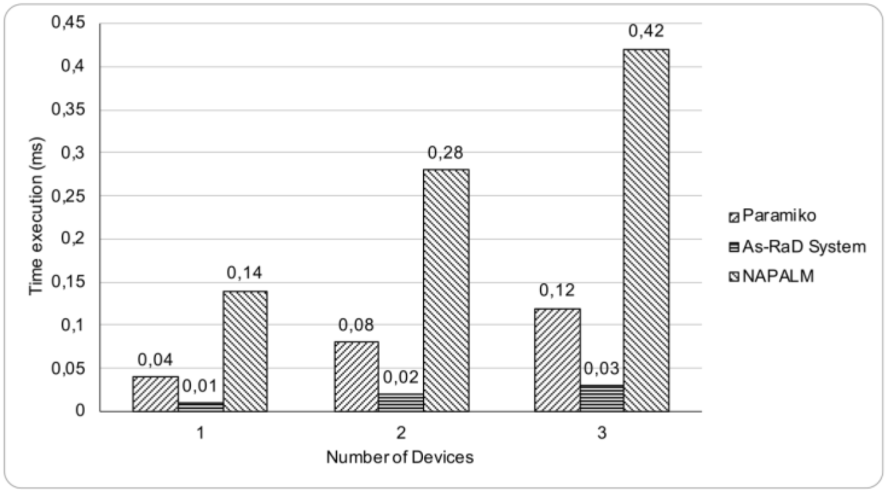


Figure 9 As-RaD System vs Paramiko and NAPALM

From the results obtained in the graph shown by Figure 9, a calculation is made to find the percentage of the results of the comparison of the As-RaD System with the Paramiko and NAPALM methods. The formula approach is as follows.

$$Result = \frac{Time\ Method - As\_RaD\ System}{Time\ Method} x\ 100\ \% \qquad (1)$$

The result is that the As-RaD System is 75% faster than Paramiko and 92% faster than NAPALM. From the result, we conclude that As-RaD has better performance to apply network automation based on time execution compare to the previous methods applied (Paramiko and NAPALM).

4. Conclusion

As-RaD System has the web-based application that is used to automate Cisco CSR1000V devices. This application applies the REST API method that has the ability to communicate with computer network devices, especially those that already have REST API architecture support. When communicating using the REST API, which in this case is network device management, this method will return data in the JSON (Javascript Object Notation) [26] format. This is different from the command line interface (CLI) which returns information that is intended for humans [27] but not for an application. In this work, we obtained results of network automation tests using the As-RaD System has a faster performance of 75% compared to the Paramiko method and 92% faster than the NAPALM method in completing network management tasks.

Acknowledgement

This research was financially supported by The Faculty of Engineering, Diponegoro University, Semarang, Indonesia through Strategic Research Grant 2020 number: 2496/STK05/UN7.5.3.2/PP/2020.

References

[1] H. Kim and N. Feamster, “Improving network management with software defined networking,” *IEEE Commun. Mag.*, vol. 51, no. 2, pp. 114–119, 2013. https://doi.org/10.1109/MCOM.2013.6461195
[2] J. Thielens, “Why APIs are central to a BYOD security strategy,” *Netw. Secur.*, vol. 2013, no. 8, pp. 5–6, 2013. http://dx.doi.org/10.1016/S1353-4858(13)70091-6

[3] A. F. Rochim, M. A. Aziz, and A. Fauzi, “Design Log Management System of Computer Network Devices Infrastructures Based on ELK Stack,” *ICECOS 2019 - 3rd Int. Conf. Electr. Eng. Comput. Sci. Proceeding*, pp. 338–342, 2019. https://doi.org/10.1109/ICECOS47637.2019.8984494

[4] A. P. Paramitha, A. F. Rochim, and A. Fauzi, “Design and Implementation Network Administrators Account Management System Based on Authentication, Authorization, and Accounting Based on TACACS and LDAP,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 803, no. 1, 2020. https://doi.org/10.1088/1757-899X/803/1/012040

[5] A. Ratan, *Practical Network Automation*, Second. Mumbai: Packt Publishing Ltd, 2018.

[6] E. Chou, *Mastering Python networking*. Packt Publishing, 2018.

[7] Brandon Rhodes and John Goerzen, *Foundations of Python Network Programming, Third Edition*, Third edit. Apress, 2014.

[8] W. Zhou, L. Li, M. Luo, and W. Chou, “REST API design patterns for SDN northbound API,” *Proc. - 2014 IEEE 28th Int. Conf. Adv. Inf. Netw. Appl. Work. IEEE WAINA 2014*, pp. 358–365, 2014. https://doi.org/10.1109/WAINA.2014.153

[9] F. Mehmood, I. Ullah, S. Ahmad, and D. H. Kim, “A novel approach towards the design and implementation of virtual network based on controller in future iot applications,” *Electron.*, vol. 9, no. 4, 2020. https://doi.org/10.3390/electronics9040604

[10] Aaron Rohyans, A. Shaikh, C. B. Rajaram, and et.al, “Cisco SD-WAN Cloud Scale Architecture,” Cisco, 2019.

[11] R. A. Wiryawan and N. R. Rosyid, “Website-based Network Administration Automation Application Development Using the Python Programming Language,” *Simetris J. Tek. Mesin, Elektro dan Ilmu Komput.*, vol. 10, no. 2, pp. 741–752, 2019. https://doi.org/10.30743/infotekjar.v5i1.2758

[12] P. Mihaila, T. Balan, R. Curpen, and F. Sandu, “Network Automation and Abstraction using Python Programming Methods,” in *MACRo 2015*, 2017, vol. 2, no. 1, pp. 95–103. https://doi.org/10.1515/macro-2017-0011

[13] Snatch, “NAPALM’s initial connection is to slow,” *Github*. [Online]. Available: https://github.com/napalm-automation/napalm/issues/747. [Accessed: 20-Apr-2020].

[14] Sturmf, “Performance of paramiko is pretty slow,” 2013. [Online]. Available: https://github.com/paramiko/paramiko/issues/175. [Accessed: 17-Apr-2020].

[15] R. Hevner Alan, “A Three Cycle View of Design Science Research,” *Scand. J. Inf. Syst.*, vol. 19, no. 2, pp. 87–92, 2007. https://doi.org/10.1007/978-3-642-29863-9_23

[16] S. Lota and M. Markowski, “Performance Analysis of Virtual Computer Network Based on Cisco Cloud Services Router 1000V in a Private Cloud Environment,” *J. Appl. Comput. Sci. Methods*, vol. 7, no. 2, pp. 117–132, 2016. http://doi.org/10.1515/jacsm-2015-0013

[17] O. Karnalim and M. Ayub, “The Use of Python Tutor on Programming Laboratory Session: Student Perspectives,” *Kinetik*, vol. 2, no. 4, p. 327, 2017. https://doi.org/10.22219/kinetik.v2i4.442

[18] Django Software Foundation, “Design Philosophies,” 2017.

[19] K. R. Jackson *et al.*, “Performance analysis of high performance computing applications on the Amazon Web Services cloud,” *Proc. - 2nd IEEE Int. Conf. Cloud Comput. Technol. Sci. CloudCom 2010*, pp. 159–168, 2010. https://doi.org/10.1109/CloudCom.2010.69

[20] M. A. A. Samiha Islam Abrita, Moumita Sarker, Faheem Abrar, “Benchmarking, VM Startup Time in the Cloud,” in *Lecture Notes in Computer Science*, 2018, pp. 53–64. https://doi.org/10.1007/978-3-030-32813-9_6

[21] O. Hrebicek and L. Chung, “Virtual private cloud that provides enterprise grade functionality and compliance.” Google Patents, 2014.

[22] M. T. Jones, “Conversing through the Internet with cURL and libcurl,” pp. 1–10, resource: https://www.ibm.com/developerworks/library/os-curl/os-curl-pdf.pdf 2009.

[23] D. Jacobson, D. Woods, and G. Brail, *APIs: A Strategy Guide*. O’Reilly Media, 2011.

[24] M. Amundsen, *RESTful Web Clients: Enabling Reuse Through Hypermedia*. O’Reilly Media, 2017.

[25] S. M. Ross, “Peirce ’ s criterion for the elimination of suspect experimental data,” no. September 2003, 2016.

[26] C. Severance, “Discovering JavaScript Object Notation,” *Computer (Long. Beach. Calif).*, vol. 45, no. 4, pp. 6–8, Apr. 2012. https://doi.org/10.1109/MC.2012.132

[27] B. Claise, J. Clarke, and J. Lindblad, *Network Programmability with YANG: The Structure of Network Automation with YANG, NETCONF, RESTCONF, and gNMI*. Pearson Education, 2019.

As-RaD System as a Design Model of the Network Automation Configuration System Based on the REST-API and Django Framework

ORIGINALITY REPORT

3%	%	3%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Paul Mihăilă, Titus Bălan, Radu Curpen, Florin Sandu. "Network Automation and Abstraction using Python Programming Methods", MACRo 2015, 2017 Publication	1%
2	Sara Lazarevic, Tamara Zuvela. "Design Of Information System To Support Movie Ticket Booking And Cinema Operations", 2022 21st International Symposium INFOTEH-JAHORINA (INFOTEH), 2022 Publication	1%
3	Lailis Syafa'ah, Agus Eko Minarno, Fauzi Dwi Setiawan Sumadi, Dwi Anggraini Puspita Rahayu. "ESP 8266 For Control And Monitoring In Smart Home Application", 2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE), 2019 Publication	<1%

4	Ni Made Satvika Iswari, Eko K. Budiardjo, Zainal A. Hasibuan. "Integrated e-Business System Architecture for Small and Medium Enterprises", Proceedings of the 2nd International Conference on Software Engineering and Information Management, 2019 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		