Designing, Developing and Deploying an Enterprise Scale Network Monitoring System

Arakdip Basu
arkadip.basu@walmart.com
Walmart Global Tech
Bangalore, Karnataka, India

Rishi Singh
rishi.singh@walmart.com
Walmart Global Tech
Bangalore, Karnataka, India

Chenyang Yu
chenyang.yu@walmart.com
Walmart Global Tech
Sunnyvale, California, USA

Amarjeet Prasad
a.avdhesh.prasad@walmart.com
Walmart Global Tech
Bangalore, Karnataka, India

Kunal Banerjee
kunal.banerjee1@walmart.com
Walmart Global Tech
Bangalore, Karnataka, India

ABSTRACT

Walmart carries out its retail business across 27 countries both in the form of brick-and-mortar (~11,500 stores and clubs) and e-commerce. To ensure smooth customer experience across the globe, we need to monitor the health of all devices ranging from networking hardware, storage spaces to compute servers spread across geographies all the time. Specifically, we need to monitor which device is facing as issue, when did this happen and what kind of alert does it call for. Swift remediation is carried out in a pro-active manner, i.e., before a device fails, and sometimes in re-active manner, i.e., after a device has failed. Tackling this challenge at an enterprise scale requires various technologies working together in a seamless manner. In this work, we give an insight about how the problem of network monitoring is handled at Walmart and elaborate on the design decisions taken.

CCS CONCEPTS

• **Software and its engineering** → **Software design techniques**; *Maintaining software*; **System administration**; • **Networks** → **Network monitoring**; Network performance analysis.

KEYWORDS

network monitoring system, OpenNMS, service availability, re-active monitoring, pro-active monitoring

ACM Reference Format:
Arakdip Basu, Rishi Singh, Chenyang Yu, Amarjeet Prasad, and Kunal Banerjee. 2022. Designing, Developing and Deploying an Enterprise Scale Network Monitoring System. In *15th Innovations in Software Engineering Conference (ISEC 2022), February 24–26, 2022, Gandhinagar, India.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3511430.3511446

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ISEC 2022, February 24–26, 2022, Gandhinagar, India
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9618-9/22/02…$15.00
https://doi.org/10.1145/3511430.3511446

1 INTRODUCTION

Walmart stepped into the retail industry when it opened its first store in USA in 1962. It has come a long way since then and now has presence in 27 countries in the form of both physical stores and digital platforms. To provide the customers with the best experience, Walmart has truly adopted the big data strategy and collects multiple petabytes of data every hour [5]. This data may be used in the brick-and-mortar stores for improving the store checkout process for the customers or optimizing the product assortment for better display of the merchandise, or in e-commerce platforms for providing enhanced personalized shopping experience, or used for internal operations such as calculating routes for transporting merchandise [8]. Walmart is committed to its policy of *EveryDay Low Price (EDLP)* so that customers need not incur extra expenses. One way that Walmart ensures EDLP is by embracing *EveryDay Low Cost (EDLC)*, i.e., it tries to minimize its operational costs so that the resulting savings can be passed along to the customers. Thus, EDLC calls for innovations that deliver fast and efficient solutions.

Network monitoring system (NMS) forms an integral part of Walmart to assure uninterrupted business operations. It includes surveiling the availability, operation and performance of devices linked together through routers, switches, cables, etc. These networks are often spread across geographies but need to function in unison to offer a seamless experience to our customers and associates. In short, NMS is designed to ensure *service availability* and *service reliability* and in order to do so it needs to regularly monitor the health of a large number of devices, analyze their performance, predict device failures, generate alerts and prevent downtimes.

In this work, we present the network monitoring system that has been designed and developed in-house and now is in production for about two years. It is based on OpenNMS [7] which is an open source network monitoring platform built for scale. Our solution augments OpenNMS with enhanced search capability, advanced alert services via email and different business communication platforms, and sophisticated data storage facilities to aid in network analyses.

The paper is organized as follows. Section 2 describes the NMS solution developed at Walmart. Section 3 covers the experimental results that includes an engineer satisfaction survey. Section 4 concludes the paper and outlines some future work.

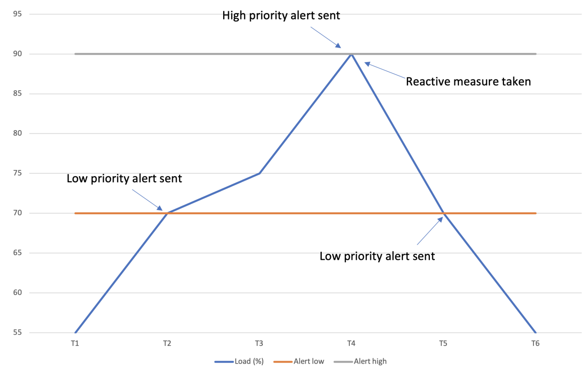


Figure 1: An example to illustrate the benefits of a network monitoring system.

2 NETWORK MONITORING SYSTEM AT WALMART

2.1 Benefits of an NMS

To briefly illustrate what is an NMS and what type of benefit it provides, let us consider the example shown in Figure 1. In this example, we are monitoring the health of a router by recording its bandwidth load at constant intervals (which typically varies from the order of seconds to a few minutes). As per the specification, NMS is supposed to send a low priority alert when the load reaches 70% and a high priority alert at 90% load level. At time T1, the device was operating at a normal level; however, at time T2, its utilization reaches 70% level and hence at this time a "low priority alert" is sent. Perhaps this alert is ignored by the engineer, and subsequently, the load on the router is further increased at T3 and finally, it reaches the 90% threshold at T4 – at this moment, a "high priority alert" is sent. This time, the engineer takes a quick reactive measure (possibly, by diverting some of the traffic to another router) and consequently, the utilization is brought down steadily. Another low priority alert is sent to the engineer at time T5 when 70% load is again reached; this alert assures the engineer that his/her remediation strategy is working. It is worth noting that instead of having a human-in-the-loop, an automated reactive measure may also be taken if our client team approves it and supplies us with the resolution steps. In fact, the benefit of having an automated system in place is magnified in case of taking pro-active measures; in this example, an automated system may have started diverting some of the traffic upon receiving the first low priority alert, and thus may have prevented reaching a critical utilization level altogether. Since the number of networking, telecommunication, and storage devices in Walmart are in tens of thousands, having an automated system for resolving critical events is important. We work with our client teams to have these solutions ready for deployment.

2.2 Architecture of Walmart's NMS

The OpenNMS [7] framework supports monitoring for all kinds of networking related equipment. In Walmart, we have developed and customized the design as per our infrastructure requirements. We need to support routers, switches, load balancers, etc. from various different hardware vendors for continuous monitoring.

The high-level architecture of Walmart's NMS is given in Figure 2. The OpenNMS system comes with a generic front end and a set of minions which work as agents between the devices and the server; the minions are tasked with performing scheduled jobs remotely on the devices and then communicating the information back to the server. When the server checks the device, it is called a *poll* in the jargon of Simple Network Management Protocol (SNMP), while when the device reports any issue to the server it is known as a *trap* in SNMP. To enable such communication, we have to push specific few lines of configuration in a device. Specifically, a secret community string is pushed to the device which is common between the device and the monitoring server, so that the server can find the device and check its status by poll mechanism. In order to send a trap, we have to define a Virtual IP (VIP) with a designated port number at the device level, so that for reporting any event in the device, it can communicate to the corresponding server. Since we need to work with streaming data at large scale, OpenNMS uses Apache Kafka [6]. To keep track of data, Kafka partitions messages across Kafka topics which have uniquely identifiable names in a Kafka cluster; once a producer writes a message to a topic, multiple subscribers can read from it. The poll and the trap information are streamed over such Kafka topics. The connectivity between Kafka and the minions is bi-directional. The in-house developed NMS consumer model manages the flow and generates the relevant alerts. The system logs are passed to the NMS backend system, which then allows the users to see all the alerts, events, information from the devices, and perform analyses. Also, the data is segregated into SQL data and NoSQL time series data, and stored in databases for future analyses. After NMS consumer processing, the data is passed to ElasticSearch module that provides advanced searching capabilities. In addition, we use Redis [4] as an *event enricher module* that appends events with supplementary information about devices and alerts. We chose Redis over other similar services because of its fast processing power. Subsequently, based on the business logic, the alerts are sent to the users. There are couple of other downstream applications too, which are under development, and are discussed in the future work section. Since the entire NMS solution is hosted in containerized mode with Docker [9] and Kuberenetes [3], it ensures high availability and high scalability. Note that, owing to unforeseen scenarios, there may be an outage on the hosting platform, which may hamper our monitoring solution. To tackle this problem, we have a separate module that monitors the monitoring system itself (i.e., the main server); in case any issue in the container is detected, the system administrator in our team is notified, and we take the necessary steps to resolve the issue and make the monitoring system stable.

2.3 Alerts

The health of each device being monitored by our NMS is examined at short intervals and logged; we call all such records as *events*. Note that every event does not require the attention of an engineer or call for an intervention; however, when it does, for example, in the case of an outage when a device is no longer accessible or the utilization of a device has reached some critical level, our system needs to send out notifications, which are more commonly known as *alerts*. Currently, we send out 101 types of different alerts. The priority

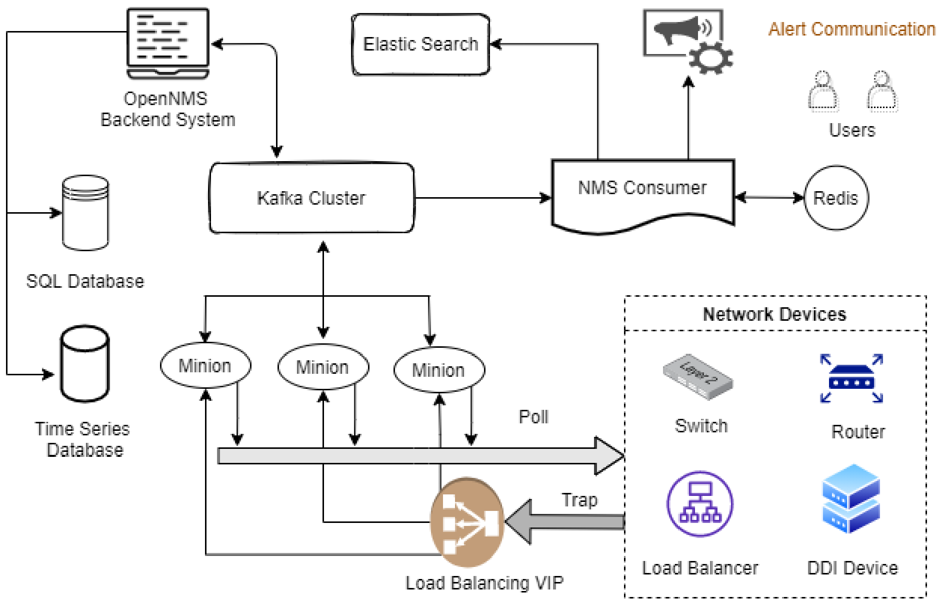
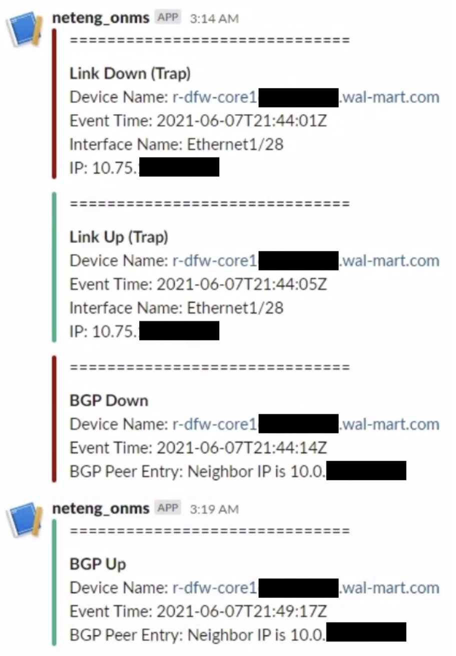


Figure 2: High-level architecture of Walmart's network monitoring system.

level of an alert, however, varies based on its business impact and the urgency to resolve the issue. We have consulted with our client teams and have come up with four priority levels: P1 (highest), P2, P3 and P4 (lowest), and have assigned each of the 101 alerts to one of these priority levels. Presently, the number of alerts belonging to each level are as follows: P1 – 19, P2 – 30, P3 – 31, P4 – 21. It may be worth noting that different subsets of these alerts are relevant to each client team. We give an example of an alert belonging to each priority level below.
P1: Node Down – if the device is offline/non-reachable.
P2: Interface Error – if there is insufficient bandwidth to support the traffic volume.
P3: Disk Partition-Low Free Space – if the free space available in the device is below a pre-specified threshold.
P4: Power Status Change – if the power of a non-critical device is down.

2.4 User Interfaces

Our NMS solution communicates with the users over multiple channels. Firstly, notifications or alerts are sent over either Microsoft Teams or Slack or both, as per the choice of the client team. An example of such notifications sent over Slack is shown in Figure 3; positive and negative notifications are marked in green and red, respectively, for easy identification. As can be seen from the time stamps in this example, the alerts were addressed and fixed within brief time periods. Sometimes, more critical alerts that need immediate attention may also be sent as email over Micorosft Outlook. These communications over different business channels not only help the engineers but also the managers for auditing purpose. Secondly, engineers may be interested in investigating the health of a particular device, for such purposes we use the user interface provided by OpenNMS; an example of the same is given in Figure 4.


neteng_onms APP 3:14 AM
==============================
Link Down (Trap)
Device Name: r-dfw-core1[redacted].wal-mart.com
Event Time: 2021-06-07T21:44:01Z
Interface Name: Ethernet1/28
IP: 10.75.[redacted]

==============================
Link Up (Trap)
Device Name: r-dfw-core1[redacted].wal-mart.com
Event Time: 2021-06-07T21:44:05Z
Interface Name: Ethernet1/28
IP: 10.75.[redacted]

==============================
BGP Down
Device Name: r-dfw-core1[redacted].wal-mart.com
Event Time: 2021-06-07T21:44:14Z
BGP Peer Entry: Neighbor IP is 10.0.[redacted]

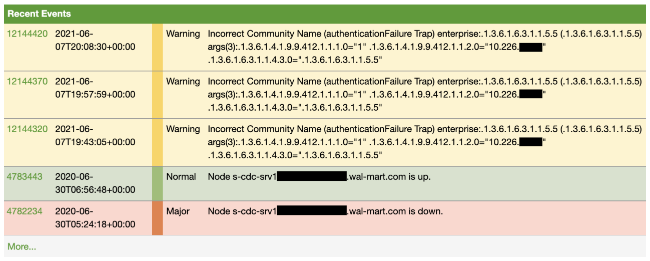
neteng_onms APP 3:19 AM
==============================
BGP Up
Device Name: r-dfw-core1[redacted].wal-mart.com
Event Time: 2021-06-07T21:49:17Z
BGP Peer Entry: Neighbor IP is 10.0.[redacted]

Figure 3: An example of notifications sent over a business communication platform (note that to protect sensitive information, some portions have been masked out).

3 EXPERIMENTAL RESULTS

3.1 Usage Patterns

In this section, we share our findings from more than 50, 000 alerts collected over a span of 17 days, with an average of ~3200 alerts



Recent Events			
12144420	2021-06-07T20:08:30+00:00	Warning	Incorrect Community Name (authenticationFailure Trap) enterprise:.1.3.6.1.6.3.1.1.5.5 (.1.3.6.1.6.3.1.1.5.5) args(3):.1.3.6.1.4.1.9.9.412.1.1.1.0="1" .1.3.6.1.4.1.9.9.412.1.1.2.0="10.226.[redacted]" .1.3.6.1.6.3.1.1.4.3.0=".1.3.6.1.6.3.1.1.5.5"
12144370	2021-06-07T19:57:59+00:00	Warning	Incorrect Community Name (authenticationFailure Trap) enterprise:.1.3.6.1.6.3.1.1.5.5 (.1.3.6.1.6.3.1.1.5.5) args(3):.1.3.6.1.4.1.9.9.412.1.1.1.0="1" .1.3.6.1.4.1.9.9.412.1.1.2.0="10.226.[redacted]" .1.3.6.1.6.3.1.1.4.3.0=".1.3.6.1.6.3.1.1.5.5"
12144320	2021-06-07T19:43:05+00:00	Warning	Incorrect Community Name (authenticationFailure Trap) enterprise:.1.3.6.1.6.3.1.1.5.5 (.1.3.6.1.6.3.1.1.5.5) args(3):.1.3.6.1.4.1.9.9.412.1.1.1.0="1" .1.3.6.1.4.1.9.9.412.1.1.2.0="10.226.[redacted]" .1.3.6.1.6.3.1.1.4.3.0=".1.3.6.1.6.3.1.1.5.5"
4783443	2020-06-30T06:56:48+00:00	Normal	Node s-cdc-srv1[redacted].wal-mart.com is up.
4782234	2020-06-30T05:24:18+00:00	Major	Node s-cdc-srv1[redacted].wal-mart.com is down.
More...			

Figure 4: An example of the user interface used for monitoring the status of a particular device (note that to protect sensitive information, some portions have been masked out).

Table 1: Usage patterns for 17 days by some clients of our NMS system

Team	#Alerts	Daily				Hourly			
		Min	Max	Avg	Std dev	Min	Max	Avg	Std dev
Team1	35067	411	3079	2062.77	916.84	56	222	85.95	35.25
Team2	16808	373	2000	988.71	367.77	21	420	41.19	93.70
Team3	2178	56	254	128.12	48.83	5	19	5.34	3.37
Team4	264	2	48	15.53	12.66	0	19	0.65	3.87

registered each day. These alerts correspond to four teams whom we designate as Team1, Team2, Team3 and Team4, for anonymity. Note that our solution has been in production for two years now and has been adopted by other teams; however, for privacy reasons, we focus on the data collected for only four teams for a small period. We believe that the data chosen for this study provides good indications about the overall usage of our developed system, while highlighting its various idiosyncrasies. Team1 is responsible for managing the networking hardware and telecommunications equipment for a specific vendor. Team2 manages the load balancers supplied by two different vendors. Team3 is accountable for maintaining the office networks stable for a given geography. Lastly, Team4 is responsible for various networking related services such as, domain name system (DNS) and dynamic host configuration protocol (DHCP). Team1 and Team2 are examples of high usage clients for our system, while Team3 and Team4 are examples of mid and low usage clients, respectively. Table 1 captures the usage patterns of our NMS solution by these teams. Specifically, we show the number of alerts recorded by our NMS across the numerous devices which are monitored by us for the respective teams. We then look into the chronological distribution of these alerts at different granularity levels – daily and hourly; for each, we provide the minimum, the maximum and the average number of alerts along with their standard deviations. Figure 5 shows the daily and the hourly distributions of alerts for Team2. Apparently, the distribution of alerts occurring for Team2 seems to be more uniform than Team1 (as indicated by the lower standard deviation value under "Daily" column); however, on zooming in at an hourly basis, it is revealed that the number of alerts occurring for Team2 follows a more irregular pattern than Team1 (as indicated by the higher standard deviation value under "Hourly" column). Another interesting observation is that the maximum number of alerts occurring at an hourly basis for Team3 and Team4 are identical although Team3, in general, deals with much more number of alerts; this happens because when a service that Team4 provides is interrupted, it typically generates alerts on multiple devices. It may be noted that we have analyzed the distributions at granularity levels of minutes and seconds also but we omit these here for brevity.

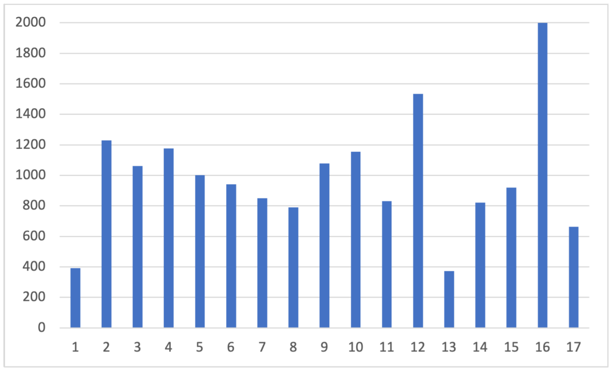
We further look into the priority-wise break up of the alerts that are encountered in the specified period for each of the teams. This is shown in Figure 6. From the figure we can see that for Team1, the alerts were either categorized as high priority P1 (61%) or low priority P4 (35%), with a very small number of these categorized as P2 or P3 (2% each). Team2, on the other hand, seems to follow a bottom-heavy approach with a large number of alerts categorized as P3 (52%) or P4 (32%), and a smaller percentage allocated as P1 (14%), with the remaining as P2 (2%). Surprisingly, no alert categorized as P2 was encountered by Team3 in this period, and an insignificant number of alerts was categorized as P3 (1%). Majority of the alerts were given a low priority P4 (75%), whereas almost a quarter of the alerts were considered as high priority P1 (24%). Finally, Team4, in contrast to some of the earlier teams, follows a top-heavy approach; a high percentage of the alerts belonged to either P1 (36%) or P2 (57%), and smaller percentages were assigned to P3 (5%) and P4 (2%).

3.2 Survey Results

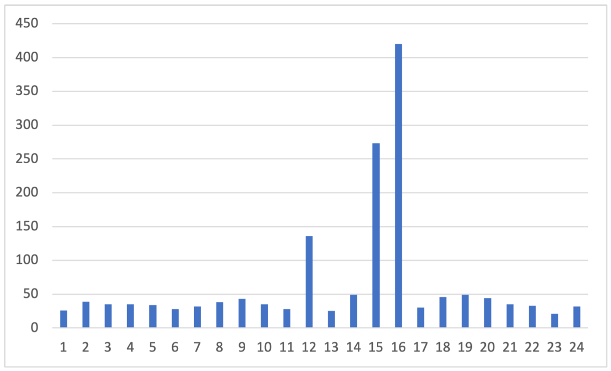
Table 2: Survey results from Walmart engineers in the client teams

Question	Score
Current NMS solution is helping to keep network infrastructure stable (1–10)	8
Based on the types of alerts supported by the system, are we fully covered (Yes/No)	Y: 78%, N: 22%
Is the NMS system putting any extra load on the infrastructure (Yes/No)	Y: 0%, N: 100%
Do you get timely alerts (i.e low latency between device and monitoring server) (Yes/No)	Y: 89%, N: 11%
Are we able to proactively monitor and take corrective actions before any major outage (Yes/No)	Y: 89%, N: 11%
The current NMS solution is better than before (1–5)	4.67

We further conducted a survey to get an idea about how satisfied the engineers in our client teams feel about our NMS solution. This has been given in Table 2. The majority of the questions were Yes/No type, while a couple of others required scores between 1 to 5, and 1 to 10, with the lower numbers representing lower satisfaction level. As can be seen from this table, the response has been largely positive. Note that we have also asked for comparison between

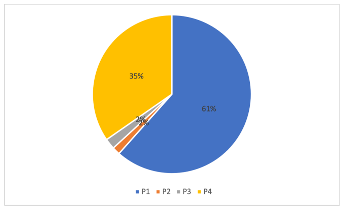


(a) Daily distribution of alerts.

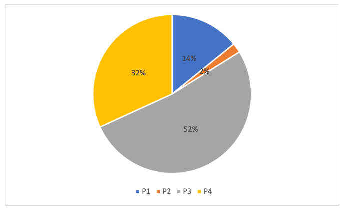


(b) Hourly distribution of alerts.

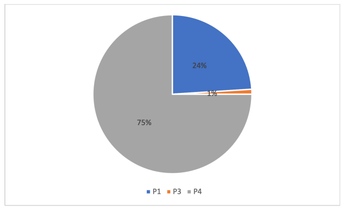
Figure 5: Chronological distribution of alerts for Team2.



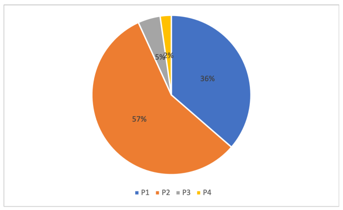
(a) Team1



(b) Team2



(c) Team3



(d) Team4

Figure 6: Priority-wise break up of the alerts in percentages for each team.

our current solution with another external commercial solution[1] that was there before; based on the response, it seems that our solution is much more appreciated over the other. We also asked for feedback on what new features may be added; some responses involved more automated resolution support, while some of the others requested for better dashboard.

4 CONCLUSION AND FUTURE WORK

Network monitoring is an essential requirement for an industry to provide continuous and seamless user experience. It becomes more and more challenging with the growth of the industry, and for Walmart, which operates across multiple continents with myriad of services entailing different business disciplines, the task becomes colossal. Consequently, we adopted OpenNMS which is an open source network monitoring system that is built for scale, and then suitably enhanced it with advanced search capabilities with ElasticSearch, fitted the backend with separate SQL and time series databases for storing the massive data generated by our company along with providing efficient and fast retrieval of relevant data with minimal latency. Moreover, we augmented the system with different business communication platforms for sending alerts to our users. These alerts often help in averting device downtimes and/or malfunctions in a pro-active or a re-active manner, and thus save revenues for the company. Additionally, monitoring the health of these devices also prolong their longevity, and therefore require less frequent replacements and repairs.

In future, we plan to enhance root cause analyses for device failures using state-of-the-art machine learning algorithms. We are also working on a new user interface that will allow advanced filters for query search, and will also have better visualizations. Some work on auto-assigning jobs on the Jira project management software [1] has been done by our team for several Walmart teams [2]. Hence, on encountering or predicting a device failure, we may assign the corresponding job to an engineer in an automated fashion for swift resolution.

[1]The name of the competitor has been purposefully withheld for legal reasons.

REFERENCES

[1] atlassian.com. [n.d.]. Jira Software. https://www.atlassian.com/software/jira. Accessed: 2021-06-08.
[2] Arkadip Basu and Kunal Banerjee. 2021. Designing a Bot for Efficient Distribution of Service Requests. In *BotSE@ICSE*. 16–20.
[3] Kelsey Hightower, Brendan Burns, and Joe Beda. 2017. *Kubernetes: Up and Running Dive into the Future of Infrastructure* (1st ed.). O'Reilly Media, Inc.
[4] Tiago Macedo and F. Oliveira. 2011. Redis Cookbook. O'Reilly Media, Inc.
[5] Bernard Marr. 2017. Really Big Data At Walmart: Real-Time Insights From Their 40+ Petabyte Data Cloud. https://www.forbes.com/sites/bernardmarr/2017/01/23/really-big-data-at-walmart-real-time-insights-from-their-40-petabyte-data-cloud/?sh=25137c2a6c10. Accessed: 2021-05-14.
[6] Neha Narkhede, Gwen Shapira, and Todd Palino. 2017. *Kafka: The Definitive Guide Real-Time Data and Stream Processing at Scale* (1st ed.). O'Reilly Media, Inc.
[7] opennms.com. [n.d.]. OpenNMS. https://github.com/opennms/. Accessed: 2021-05-14.
[8] Walmart Staff. 2017. 5 Ways Walmart Uses Big Data to Help Customers. https://corporate.walmart.com/newsroom/innovation/20170807/5-ways-walmart-uses-big-data-to-help-customers. Accessed: 2021-05-14.
[9] John Willis. 2015. *Docker and the Three Ways of DevOps*. Technical Report. https://goto.docker.com/rs/929-FJL-178/images/20150731-wp_docker-3-ways-devops.pdf Accessed: 2021-06-08.