Anomaly Detection and Monitoring in Internet of Things Communication

Deris Stiawan
Department of Computer Engineering, Faculty of Computer Science,
Universitas Sriwijaya
deris@unsri.ac.id

Mohd. Yazid Idris
Dept. Soft Engineering
Faculty of Computing,
Universiti Teknologi Malaysia
yazid@utm.my

Reza Firsandaya Malik
Department of Computer Engineering, Faculty of Computer Science,
Universitas Sriwijaya
reza.fm@unsri.ac.id

Siti Nurmaini
Department of Computer Engineering, Faculty of Computer Science,
Universitas Sriwijaya
siti@unsri.ac.id

Rahmat Budiarto
College of Computer Science & Inf. Technology, Albaha University, Saudi Arabia
rahmat@bu.edu.sa

Abstract**—** The Internet of Things (IoT) presents unique challenges in detecting anomaly and monitoring all connected devices in a network. Moreover, one of the objectives of anonymity in communication is to protect the data traffic of devices. The summary status visualization is indispensable to depict all devices/sensors that are most indicative of a pending failure and a predictive power/energy. Thus, this paper proposes a multi-platform monitoring and anomaly detection system that supports heterogeneous devices. The proposed system addresses the problems of: (i) how to monitor the network to prevent device failures and (ii) how to design a comprehensive feature for the early anomaly detection of IoT communication.

Keywords—Monitoring, Detection Anomaly, Internet of Things, Heterogeneous

I. Introduction

The Internet of Things (IoT) is currently going through a phase of rapid growth. Analysts have predicted that the IoT will become the "next big thing" in upcoming years. It has also been predicted that, within the next decade, the Internet will exist as a seamless fabric of classic networks and networked objects. The increasing number of IoT users also raises new issues, where various multi-platform devices converge into one centralized, interconnected, shared, multi-user, multi-device and flexible network. Studies done by [1] and [2] state the main issues with the IoT. The growing internetwork has established a heterogeneous network that is more complex than before. With various devices attached in the network, technical problems will increase in monitoring, managing, surveying and early detection of the network itself.

On the other hand, because of the number of network devices, there will be a need for a monitoring mechanism of network information, which is designed to visualize the status of IoT infrastructure and ensure the attached devices are in normal and active conditions. Thus, the abilities to visually manage traffic statistics in the form of charts, check the condition of devices and predict potential problems are needed.

Even the standard of services provided by the service providers to the end user are normally already arranged in a Service Level Agreement (SLA). However, problems involving the fulfillment of network information services still appear. In fact, this remains a hot topic of discussion because it involves a great number of intertwined attributes and factors.

This research proposes an integrated IoT traffic monitoring system with heterogeneous devices with the aim of addressing the problems indicated by [3], such as: (i) how to build an integrated prototype system for multi-platform/protocol on heterogeneous networks information, (ii) how to integrate multiple devices into a dashboard view to provide warning information of early anomaly detection before a failure occurs and (iii) how to develop a proxy as a middleware multi-protocol.

The rest of the paper is organized as follows: Section II discusses and reviews related works on the monitoring and anomaly detection issues of IoT; Section III describes the proposed system; Section IV discusses the experimental set up; Section V presents the evaluation results; and Section VI provides concluding remarks.

II. Related Work

There are some solutions for IoT vendors involving various standards that should be integrated. Unfortunately, due to incompatibility, not all platforms are able to adapt because they use their own proprietary technologies, even though these technologies are claimed to offer multi-platform support. These various technologies have also encouraged the appearance of heterogeneous network information. Studies conducted by [4], [5] and [6] mentioned that the heterogeneous IoT network must have services with the following characteristics: (i) network transparency, (ii) transparency on the location of the service, (iii) transparency of data formats and (iv) transparency of control protocols.

Problems involving IoT devices with different Simple Network Management Protocol (SNMP) versions have been discussed in [7] and [8]. The SNMP Protocol is used for capturing the inbound-outbound packet load to a monitoring application. However, the existing monitoring applications only support monitoring in a single version of the SNMP protocol. Thus, an IoT network with a monitoring and early anomaly detection system can prevent system failure, which in turn, will increase the reliability of the IoT network itself. Authors in [8] and [9] proposed a network monitoring application with an SNMP trap. The application is already

978-1-5090-4139-8/16/$31.00 ©2016 IEEE

informative but not yet able to perform the monitoring task if the SNMP protocol used is a different version; also, it merely focuses on network traffic. Besides, studies by [10] and [11] confirmed that the profiles of the system network activity (user, host, server and last mile connections) can also be indicators for conditions that occur in the network, such as: (i) utilization reaches 95% of the total traffic in a long period of time, which is typically only 40% in the peak time; (ii) continuously increased use of memory on the main server; (iii) data access on a server outside of the normal hours; and (iv) some devices attempt to connect and sync.

III. System Design

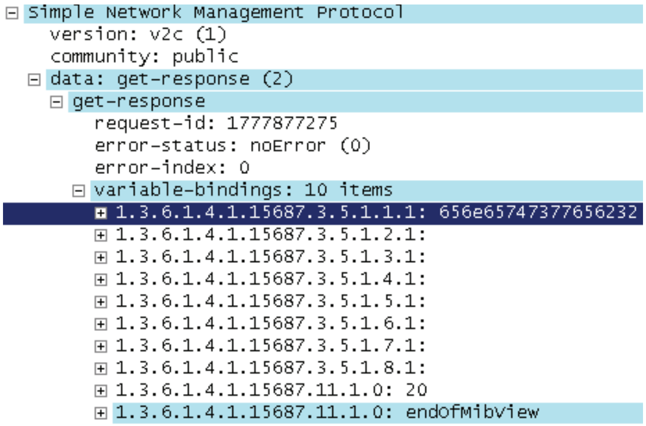
In the development of an IoT monitoring system for detecting failures, SNMP protocol is the main issue, where several heterogeneous network devices use different versions of SNMP that have different characteristics and features (SNMPv1, SNMPv2 and SNMPv3). Extracting raw data from the SNMP must be done to get the "Object Identifier" of the device.

The design consists of two phases:

(i) deploying a network monitoring system for observing the IoT network with heterogeneous devices; and

(ii) deploying an early system for detecting errors based on device profiles and activities. In fact, there are several stages to a design monitoring and detection system, including data agent input, trapping the agent, storing the data in a database, trapping the agent again (in case of data error) and displaying data.

The first step involves paying attention to the monitoring of the "Get-Request" process between managers and agents, which acquires messages that occur in the process. The format of these messages is: (i) version, (ii) community name, (iii) command, (iv) request ID, (v) error status, (vi) error index and (vii) the value of the variable from the object.



```
⊟ Simple Network Management Protocol
    version: v2c (1)
    community: public
  ⊟ data: get-response (2)
    ⊟ get-response
        request-id: 1777877275
        error-status: noError (0)
        error-index: 0
      ⊟ variable-bindings: 10 items
        ⊞ 1.3.6.1.4.1.15687.3.5.1.1.1: 656e65747377656232
        ⊞ 1.3.6.1.4.1.15687.3.5.1.2.1:
        ⊞ 1.3.6.1.4.1.15687.3.5.1.3.1:
        ⊞ 1.3.6.1.4.1.15687.3.5.1.4.1:
        ⊞ 1.3.6.1.4.1.15687.3.5.1.5.1:
        ⊞ 1.3.6.1.4.1.15687.3.5.1.6.1:
        ⊞ 1.3.6.1.4.1.15687.3.5.1.7.1:
        ⊞ 1.3.6.1.4.1.15687.3.5.1.8.1:
        ⊞ 1.3.6.1.4.1.15687.11.1.0: 20
        ⊞ 1.3.6.1.4.1.15687.11.1.0: endofMibview
```

Fig. 1. SNMP raw data

This work refers to previous studies done by [7] and [9] that focused on trapping traffic in SNMP and used the approach method of the SNMP message format. SNMP has a Protocol Data Unit (PDU) as part of the message and five types of PDU: GetRequest PDU, GetNextRequest PDU, SetRequest PDU, GetResponse PDU and Trap PDU. Fig. 1 shows the raw data packets that are successfully extracted from the traffic on the experimental network. SNMP is able to be installed in Raspberry "sysORDescr = STRING: The MIB module for SNMPv2 entities."

The system assumes user activities and network profiles based on topology, as shown in Fig. 2. Components involved in this research are able to be visualized. Request, response and trap notifications interact with the MIB server manager. Request and response traffic are set for 60 seconds, which means that every 60 seconds, an SNMP agent will trap a notification to the server manager. To get the real data, a time interval must be set appropriately, and traffic from the agent to the manager will use broadcast traffic. Thus, the issues to be addressed are the three main components in the system: (1) MIB SNMP Manager, SNMP Agent (2) and (3) User interface.

This system is designed with the following features: (i) a dashboard for viewing all inbound and outbound traffic, (ii) a display menu of all devices, (iii) visual management graphics, (iv) a device status and (v) matric mapping.

IV. Experimental Scenario

A network environment is set up in our computer network laboratory, as illustrated in Fig. 2. The network consists of two components—network peripherals and devices—in order to represent an IoT network.

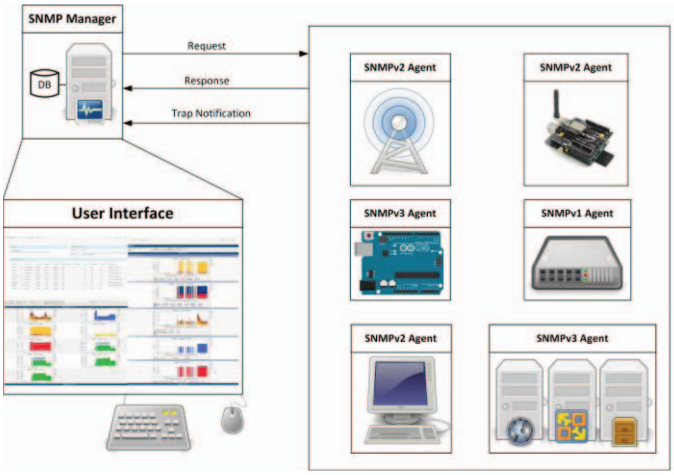


Fig. 2. Topology

The topology of the experimental network is set up in such a way that we can deploy an application system to monitor and detect anomaly on multi-platform devices attached to an IoT network.

A. Setup and Configuration

The network pheriperals are: (i) two Cisco routers as a data packet forwarder at layer 3 (Network Layer), (ii) two switches and (iii) one wireless access point router (WiFi 2.4 Ghz with AP-OpenWRT 802.11g) to provide WiFi service access from mobile devices.

The devices attached to the experimental network include:

- Two servers to run multiple applications, including web server, database server and other application servers with the following specifications: Intel Core 2 duo, 4048MB RAM, 320 GB Storage and Operating system: Ubuntu Server 14.04.3 LTS 64bit;
- Three PC Workstation users which do the user profile with Windows 7 Operating System (OS): Intel Core 2 Quad Processor Q9500, 2048MB RAM DDR3, 500GB HD;
- One server as a network monitoring MIB server: Intel Core 2 Quad Processor Q9500, 8048MB RAM DDR3, 500GB HD;
- One cloud computing server for running virtual hosts with platform Debian OS (Proxmox): Intel Xeon, 12048 MB, 1TB HD; and
- Sensors for sensing room condition: (i) two Raspberry Pi: ARM Processor, Storage MMCard 8GB, 1 port Ethernet, Raspbian OS; (ii) three Xbee S1 modules: 3.3V @ 50mA, 250kbps Max data rate, 1mW output (+0dBm); (iii) three M2303 sensors and smoke sensor.

B. Experiment Scenario

- Perform ping and tracer command and gain access to multiple servers;
- Perform server testing by running its daemon and enable it by restarting after every test;
- Measure traffic load to determine performance threshold values either in time or number of data packets;
- Gain three PC users' access to web and application servers as well as cloud servers—we duplicated treatment activities, enabling separate access to three servers, simultaneous access and random access with a specified time interval;
- Use Memory and CPU usage as measurements to enable a trap in the traffic by the agent;
- Run applications of the monitoring system to receive data packages from any agent;
- Perform filtering to distinguish normal traffic from a failure by separating and dividing the data traffic into several stages based on time, target machine and used tools;
- Pump data packet into the experimental network using the Packet Generator (packgen) to enable active users to achieve real world traffic;
- Capture raw traffic data using TCPdump to produce pcap files;
- Enable all services/daemons in target machines and restart them before each test to ensure the same starting conditions;
- Configure sensors' option configurations LINUX-RASPI: Downed Device Detection SNMP Uptime, Timeout Value 400, Retry 1, SNMP Ver 3, Community public, Port 161 and Timeout 10; and
- IPtraf, TCPdump and Collasoft Capsa are used to packet sniffing.

V. Experiment Results and Discussions

A network mapping matrix of interconnection traffic and summary data information is shown in Figure 3. This traffic was obtained during the experimental observation. Fig. 4 shows a graph that describes the traffic profiles of a user and the CPU processing usage percentage of the experimental IoT network

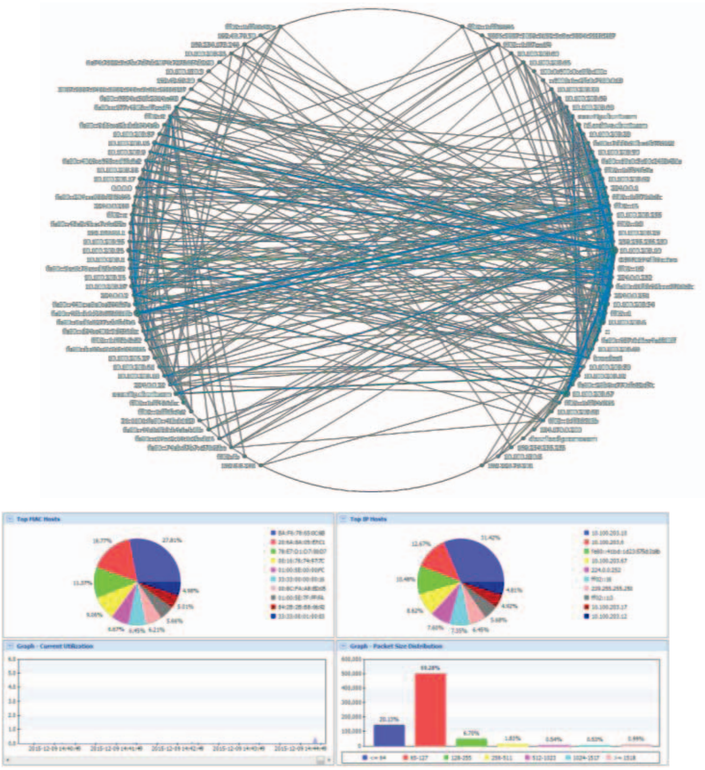


Fig. 3. Overall traffic and protocol used

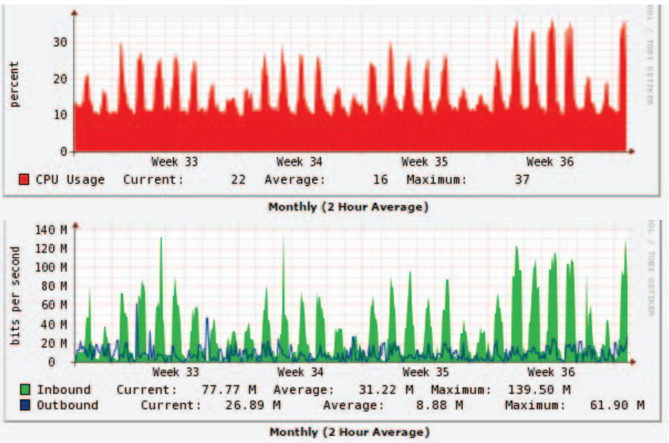
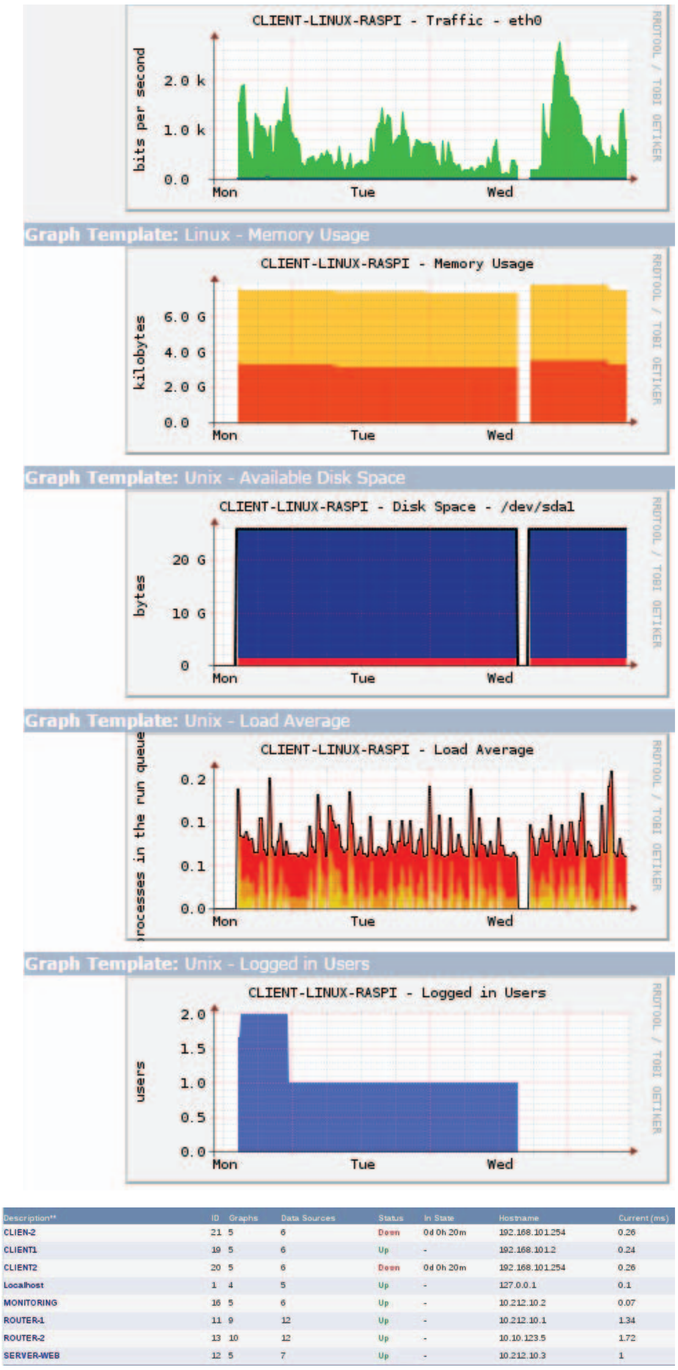


Fig. 4. Snapshot of traffic profile in a period of observation time

We analyzed the same graphic flow between the incoming and outgoing traffic of CPU/memory usage on the monitored server. Thus, it is possible to set a basic threshold value to trigger an alert in the notification system. For example, we may set the value of the threshold value network traffic as if "the traffic load < 500 Kbps or > 150 Mbps = usage processor > 45 percent." Then, the system will trigger an alert as an anomaly notification. Fig. 5 shows the Raspberry device status from the SNMP agent, displayed with a graph. The system will inform in real-time with an elapsed time of five seconds (adjustable) after the agent cannot respond to the request from the SNMP agent. Meanwhile, if the status is down, the system will be updated automatically.


CLIENT-LINUX-RASPI - Traffic - eth0
bits per second
2.0 k
1.0 k
0.0
Mon
Tue
Wed
RRDTOOL / TOBI OETIKER
Graph Template: Linux - Memory Usage
CLIENT-LINUX-RASPI - Memory Usage
kilobytes
6.0 G
4.0 G
2.0 G
0.0
Mon
Tue
Wed
RRDTOOL / TOBI OETIKER
Graph Template: Unix - Available Disk Space
CLIENT-LINUX-RASPI - Disk Space - /dev/sda1
bytes
20 G
10 G
0
Mon
Tue
Wed
RRDTOOL / TOBI OETIKER
Graph Template: Unix - Load Average
CLIENT-LINUX-RASPI - Load Average
processes in the run queue
0.2
0.1
0.1
0.0
Mon
Tue
Wed
RRDTOOL / TOBI OETIKER
Graph Template: Unix - Logged in Users
CLIENT-LINUX-RASPI - Logged in Users
users
2.0
1.5
1.0
0.5
0.0
Mon
Tue
Wed
RRDTOOL / TOBI OETIKER

Description**	ID	Graphs	Data Sources	Status	In State	Hostname	Current (ms)
CLIEN-2	21	5	6	Down	0d 0h 20m	192.168.101.254	0.26
CLIENT1	19	5	6	Up	-	192.168.101.2	0.24
CLIENT2	20	5	6	Down	0d 0h 20m	192.168.101.254	0.26
Localhost	1	4	5	Up	-	127.0.0.1	0.1
MONITORING	16	5	6	Up	-	10.212.10.2	0.07
ROUTER-1	11	9	12	Up	-	10.212.10.1	1.34
ROUTER-2	13	10	12	Up	-	10.10.123.5	1.72
SERVER-WEB	12	5	7	Up	-	10.212.10.3	1

Fig. 5. Device status visualization

Similarly, the graph will show an up and down graphic to depict traffic activity from those devices. The summary status visualization depicts not only the device sensors that are most indicative of a pending failure but also those indicative of a predictive strength. Network mapping matrix of interconnection traffic and summary data information is shown in Figure 5. This traffic obtained during the experimental observation.

VI. Concluding Remark

In this paper, we introduce a system for early anomaly and device failure detection and monitoring of the IoT network. This system is able to recognize anomalous communication devices based on their profiles by comparing normal and anomalous traffic. The results of our work can be described in two steps—that is, the system is able to: (i) determine the circumstances of anomaly in real-time and (ii) determine which of these anomalies are from network device profiles. There is significant scope for future work in these areas, such as security and privacy, IoT attack visualization, real-time Intrusion detection and reporting.

Acknowledgment

The research leading to these results has received funding from Sriwijaya University under Grant Agreement no. 042.01.2.400953/2016.

References

[1] L. Malina, J. Hajny, R. Fujdiak, and J. Hosek, "On perspective of security and privacy-preserving solutions in the internet of things," *Computer Networks*, vol. 102, pp. 83-95, 6/19/ 2016.

[2] C. Tankard, "The security issues of the Internet of Things," *Computer Fraud & Security*, vol. 2015, pp. 11-14, 9// 2015.

[3] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for Internet of Things: A Survey," *IEEE Internet of Things Journal*, vol. 3, pp. 70-95, 2016.

[4] Y. Zhenhui, J. Keeney, S. Van Der Meer, G. Hogan, and G. M. Muntean, "Context-aware heterogeneous network performance analysis: Test-bed development," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on*, 2014, pp. 472-477.

[5] P. Hyunho, L. Hyeong Ho, and L. Seung-Hwan, "IEEE 802 standardization on heterogeneous network interworking," in *Advanced Communication Technology (ICACT), 2014 16th International Conference on*, 2014, pp. 1140-1145.

[6] H. Sakakibara, J. Nakazawa, and H. Tokuda, "PBN: A seamless network infrastructure of heterogeneous network nodes," in *Networked Sensing Systems (INSS), 2009 Sixth International Conference on*, 2009, pp. 1-1.

[7] H. Yongqi, Z. Yun, L. Taihao, and C. Liying, "Research of Network Monitoring Based on SNMP," in *Instrumentation, Measurement, Computer, Communication and Control (IMCCC), 2013 Third International Conference on*, 2013, pp. 411-414.

[8] R. Sánchez, Á. Herrero, and E. Corchado, "Visualization and Clustering for SNMP Intrusion Detection," *Cybernetics and Systems*, vol. 44, pp. 505-532, 2013/10/03 2013.

[9] V. Tavares Guimaraes, G. Lessa dos Santos, G. da Cunha Rodrigues, L. Zambenedetti Granville, and L. M. Rockenbach Tarouco, "A collaborative solution for SNMP traces visualization," in *Information Networking (ICOIN), 2014 International Conference on*, 2014, pp. 458-463.

[10] M. A. Aydın, A. H. Zaim, and K. G. Ceylan, "A hybrid intrusion detection system design for computer network security," *Computers and Electrical Engineering*, vol. 35, pp. 517-526, 2009.

[11] C. Y. Wang, S.-c. T. Chou, and H.-c. Chang, "Emotion and Motivation : Understanding User Behavior of Web 2 . 0 Application," *IEEE Computer Society Seventh Annual Commnuication Networks and Services Research Conference*, pp. 1341-1346, 2009.