Employing Monitoring System to Analyze Incidents in Computer Network

Lukáš Macura, Jan Rozhon and Jerry Chun-Wei Lin

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.71102

Abstract

Today, network technologies can handle throughputs or up to 100 Gbps, transporting 200 million packets per second on a single link. Such high bandwidths impact network flow analysis and as a result require significantly more powerful hardware. Methods used today concentrate mainly on analyses of data flows and patterns. It is nearly impossible to actively look for anomalies in network packets and flows. A small amount of change of monitoring patterns could result in big increase in potentially false positive incidents. This paper focuses on multi-criteria analyses of systems generated data in order to predict incidents. We prove that system generated monitoring data are an appropriate source to analyze and allow for much more focused and less computationally intensive monitoring operations. By using appropriate mathematical methods to analyze stored data, it is possible to obtain useful information. During our work, some interesting anomalies in networks were found by utilizing simple data correlations using monitoring system Zabbix. Afterwards, we prepared and preprocessed data to classify servers and hosts by their behavior. We concluded that it is possible to say that deeper analysis is possible thanks to Zabbix monitoring system and its features like Open-Source core, documented API and SQL backend for data. The result of this work is a new approach to analysis containing algorithms which allow to identify significant items in monitoring system.

Keywords: monitoring system, computer network, data analysis, neural networks, Monda

1. Introduction

This paper explains new and efficient method to analyze and pre-process monitoring systems data for advanced analysis using neural networks and machine learning methods. Well-trained neural network can predict known and unknown types of incidents with high probability, and warn administrators before these occur. Other approaches exist and they are based on artificial intelligence that can be used for this purpose, for example, a Markovian model-based solution is described in [1, 2]. A swarm intelligence-based solution is presented in [3, 4]. It is also possible to identify the cause of the problem, for example, when indicator (e.g. free disk space) is out of range but the actual cause is elsewhere (the attack on a specific service). The big advantage of using network and system monitoring tools is that the basic correlation rules are already in monitoring systems as these are typically setup to inform administrators about abnormal behavior that could impact system availability.

There are many different monitoring systems. All the principles written here are theoretically applicable to any monitoring tool, however, we selected Zabbix, an Open-Source project. The main selection reason is the proper organization of internal data and history in this system, the possibility of in-depth, focused and automated analyses directly using SQL and open API.

We created an Open-Source tool Monda. Its primary purpose is a selection and pre-processing of Zabbix data allowing use of more sophisticated mathematical methods and procedures. The project is hosted on Github.com server and is accessible to the entire community. The project currently has 6200 lines of source code. It has been designed for team collaboration and allows adding of new analyses.

2. Methods

We can say that if we want to operate a network uninterrupted for a long term, a monitoring system is a crucial part of the successful way to accomplish this. We need to monitor and track most of network equipment and servers to have a good footprint of network. The mere recording of network logs is important, but without monitoring it is ineffective. It can occur very often that some data source (from some security probe) is missing due to failure. If we do not monitor this, network seems to be without problems even if there is a security incident on the background.

There is yet another reason for network monitoring. If an attacker knows where security device is located and he knows its vulnerabilities, he can focus first attack directly there. If this attack is successful, security device is not functioning properly and there is no monitoring enabled, administrator cannot be informed about this and next attacks.

There are software and hardware platforms that are able to detect anomalies in network traffic by inspecting packets or streams [5]. Similarly, there are platforms which are able to analyze the log files [6]. Their disadvantage is mostly narrow focus. Even if information from flows is very important, it is usually not enough for deeper analysis because there is no further information, such as load for each server or network elements. Modern devices are able to classify traffic based on the days of the week and time of day to respect common usage in networks based on work hours and work days. It is even possible to use special probes as source of data for monitoring systems like VoIP attack analysis [7–9].

3. State of the art

There is a lot of tools to identify and classify network incidents but there is no tool based on data from monitoring system. We choose Zabbix and data from Silesian University to do further analysis of data because of their availability and because of Zabbix features.

Generally, the security must be carried as close to the potential problem as possible to ensure the best possible efficiency of the security measures. The local network is required to employ properly set up measures against spoof attacks and enable general ban of unsafe services that are not used. Well-configured network should not allow trivial attacks like faking MAC addresses, IP addresses or ARP. As an opposite, in carrier level network, there has to be only limited amount of security measures. A typical example of the attacks on the carrier level where the attacks to some news sites in Czech Republic. Even though the stream of data flows across most of the big operators, the actual protection against attacks must occur on server itself.

Our goal was to identify interesting data from monitoring system and use them for further analysis.

3.1. Zabbix

Zabbix [12] is very common monitoring system developed as Open-Source. It supports variety of data inputs and can do very flexible operations over these data. In addition, it is very flexible report tool. Its internal design varies from other monitoring tools. It is suitable for further analysis because all data are available in SQL. Changes in configuration are respected automatically and restart of server is not needed to apply them. Most of data are available even over JSON API.

Zabbix key features:

- It is possible to monitor almost everything (due to vast protocols support and external scripting).
- Configuration and data in SQL.
- Web application monitoring.
- It scales from small setup to huge installations.
- Power: it can monitor thousands of new values per seconds.
- Supports user acknowledgments to problems.
- API for automatized analysis.

4. Design of the system

Mathematical model is based on common principles but for further reading and understanding we used glue between mathematical model and monitoring system. Basic rule for any analysis is correct selection of data. There is a lot of data in monitoring system. Our installation of Zabbix on Silesian University has approximately 1 TB of data. We cannot use all data for analysis and it is not even necessary. It is enough to understand the way how data are fetched and stored and how to do basic statistical analysis based these data. After this step, called pre-processing, we can do further analysis.

4.1. Anomaly

To be able to search for an anomaly, we have to define it. This can be tricky. If anomaly is defined too strictly, we can never find it. On the opposite side, too widely defined anomaly will generate a lot of events and computations. It is possible to use trigger priority and trigger acknowledgment for this. For example, we can look into problems with priority higher than warning which were not acknowledged by users.

There are six trigger priorities in Zabbix.

- Not classified—for events which do not affect network but we want to know about it.
- Information—for events which do not affect network but we want to know about it.
- Warning—for events which can affect network.
- Average—for events which effectively affect network.
- High—for events which can harm network.
- Disaster—for events which has to be solved as soon as possible and have big impact on network.

4.1.1. Time window

We have to define Time Window before searching for any anomaly. Some anomalies can last for fraction of second, other can be several days long. In addition, some processes in network are specific for some hour in day or day in week. We have to take this into account. For example, regular backup process can affect entire analysis if we do not consider it as regular process. This work primarily focuses on anomalies which have longer time range, it means several minutes and hours. This is due to the fact that monitoring system is not suitable for very short incidents due to the fact that data are fetched in regular intervals which are at minimum 30 s but in most cases it can be several minutes and data inside these intervals are summarized.

4.1.2. Item

In conjunction with Time Window, it is appropriate to analyze items separately. For example, CPU load on server can be a useful indicator what happens on server. It can have some specific features like recurrence, statistical features and some statistical associations in data [10]. Simple prediction of value or searching anomalies without external data is not efficient. Zabbix can do simple value predictions using linear extrapolation but this will work only for small amount of items due to their features. Zabbix can even evaluate some trigger based on simple mathematical formula and values in history. Again, it is not suitable for further analysis without external data. For example, CPU load can vary depending on server usage and it is not predictable.

4.1.3. Host

After simple analysis of Item History it is needed to find their correlations within host. Typical scenario is dependency between CPU usage, disk load and network interface utilization. This combination will be relatively unique for some kinds of applications or servers.

4.1.4. Correlation with events

Previous analyses were independent on events. Item and Host analysis without events did not respect overall network functionality, but only one Item/Host values in history. Event time and value is very good source of information because we can focus on specific data, time and value which caused event to raise.

4.1.5. Correlation with acknowledgments

Event is based on mathematical formula and data from history of Items. There is no human feedback. There can be some false-negatives or false-positives due to badly configured triggers. Acknowledgment is good for fine tuning analysis and to filter such false states because human (in most situation network administrator) manually tag given event as real problem or false state.

5. Algorithms

5.1. Data selection

To be able to focus on and work with huge amount of data, pre-processing is needed. This part of analysis is crucial. It would not be possible to do complex analysis over all data in monitoring system. And it would not lead to good results. Even for future, pre-processing part will be primary place for any optimization and improvements. Mathematical principles and formulas are strict and their algorithms are known and well optimized. But pre-processing is data specific and has to be driven with focus to data features. There is a lot of data inside monitoring system and there are many kinds of it. It can be number, specifying state of interface which is integer from 0 to 10, it can be float as processor load or integer which saves actual disk free space in bytes. Small change in one item is not important but same change in other item can mean big problem in network.

From this reason we have made our own Open-Source software Monda, hosted on GitHub, which is highly configurable and which does pre-processing part (but even more). Our goal was to create framework and common environment where every user can create his own version of pre-processing strategies based on his setup. After we created and tested Monda, it became possible to do further analysis of data focused to Time Window, Host or network process. One big advantage of this software is that it can be automated.

Primary goal of our work is in innovative approach to selection and pre-processing of data using algorithms above. We used dimensionless quantity *LOI* (Level Of Interest), which is integer. Bigger LOI means more interesting data. Algorithms and formulas used will be explained later. Further mathematical analysis is based on LOI. When doing some complex computation, objects with highest LOI are selected first. If there is enough CPU, RAM and disk size, it is theoretically possible to analyze all data inside or all data for specific Time Window only. But LOI will do preview of data inside Zabbix and selects most interesting data for further analysis.

5.2. Algorithms and data structures

Data in Monda are structured into Time Windows and Item statistics, see **Figure 1**. Basic feature of Monda is that data in Zabbix are untouched. So computation is based in Zabbix database and results from it are saved into Monda database. Monda database only describes data in Zabbix and mark them with adequate LOI.

As shown in **Figure 1**, Item that seems to be important in one Time Window can be uninteresting in another window. Typical example is free disk space. In most Time Windows, there is no big change of it. But in specific window where some kind of attack affected it, change can be bigger and Item can be interesting at this time. Algorithm used in pre-processing will prefer combination of Item and Time Window if there are more changes, see below.

Similar to Time Windows, there can be interesting and uninteresting one. During work hours, there is a lot of changes in network metrics and these windows will be preferred. On the opposite side, night hours can be skipped because there were no interesting processes.

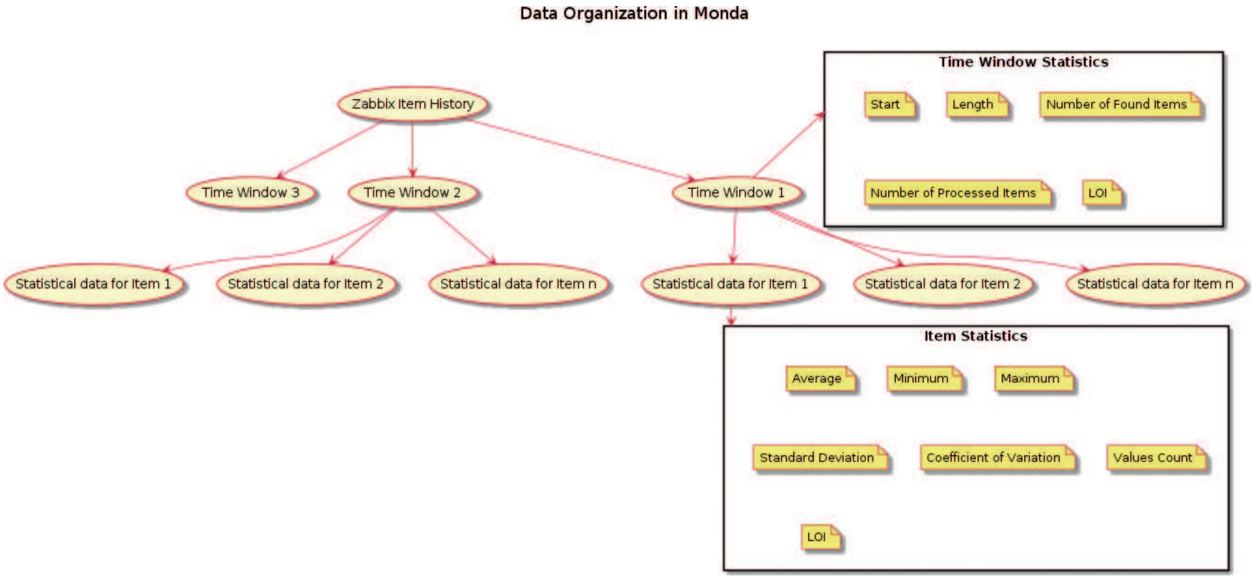
Data Organization in Monda
Zabbix Item History
Time Window 3
Time Window 2
Time Window 1
Time Window Statistics
Start
Length
Number of Found Items
Number of Processed Items
LOI
Statistical data for Item 1
Statistical data for Item 2
Statistical data for Item n
Statistical data for Item 1
Statistical data for Item 2
Statistical data for Item n
Item Statistics
Average
Minimum
Maximum
Standard Deviation
Coefficient of Variation
Values Count
LOI

Figure 1. Organization of data in Monda.

Step-by-step algorithms follows.

1. *Create Time Windows:* Monda creates Time Windows either automatically or based on user preferences. By default, windows with length 1 hour, 1 day and 1 week are created. Time Windows are hierarchical. Week window is parent of 7 single-day windows and each of day window is parent of 24 single-hour windows. But even when there is hierarchical structure, each window is computed separately.

2. *Item Statistics in Time Window:* Item statistics is set of data which describes Item history in given window. There is information about average, minimum, maximum, standard deviation, number of values, coefficient of variation and LOI. Depending on Item source, data can vary very much. For example, processor load can be number from 0 to 100%, size of free disk space can be bytes or terabytes. Situation is even more complex because some Item's absolute change can have big impact on network while same change on other Item has no impact at all. Even more, absolute value can be big but a small change can have impact too. This is typical with counter of errors on interface. Due to some historical reason, there can be thousands of CRC errors on interface but it does not increase in time. But when it increases, even a small change should be taken into account. Next problem which is related to the fact how monitoring system works is that some items can be fetched in 30 s interval but other in 10 min interval and even more. While we need to fetch CPU usage each 30 s, for disk usage it is enough to fetch it each 30 min.

At this time, algorithm works only with numerical data in Zabbix. It ignores everything else than integer or float in history. It is possible that this will change in next versions. Even string processing can be useful task to take into account. Next improvement would be to better describe data and meta information of Items. It could be possible to mark some data as more important directly in Zabbix and instruct Monda to increase LOI of Item even if absolute change of it is small.

Item statistics:

- Minimum *min*(*x*),
- Maximum *max*(*x*),
- Mean *avg*(*x*),

$$\mu = \frac{1}{count} \cdot \sum_{i=1}^{count} x_i \tag{1}$$

- Standard deviation *stddev*(*x*),

$$\sigma = \sqrt{\frac{1}{count} \cdot \sum_{i=1}^{count} (x_i - \mu)^2} \tag{2}$$

- Coefficient of variation *CV*,

$$CV = \frac{\sigma}{\mu} \tag{3}$$

- Number of values *count*,
- Level of interest LOI_{is}

$$LOI_{is} = 100 \cdot \frac{CV}{CV_{max}} \tag{4}$$

CV_{max} is configurable with default equal to 100.

The example of the command for the windows with IDs 47,245 and 46,915 looks like follows:

```
$ monda is:show -w 47,245,46,915 --itemids 1,2.
```

This command yields, for example, the following data (**Table 1**):

Itemid	1	2
min	0	0
max	1,069,274	48
avg	743	0.03
stddev	28,177	1.28
loi	546	484
cnt	1440	1440
hostid	1	2
cv	37	33

Table 1. Item statistics example.

3. *Time Windows*: For all Time Windows, Item statistics are computed. It means that for each Time Window, Zabbix history is searched, analyzed and computed for all Items found inside. Some items are automatically removed at this part of analysis because there is not enough data for them in given window. For example, for item “disk free bytes” which is fetched every 20 min there is not enough data in 1 hour window (three values) to do any usable analysis over it.

There are basic statistics computed for each Time Window, see below. All constants are configurable by Monda. This is first place where data are reduced. Useless data (items with small changes, items without history or items with small standard deviation) are not copied into Monda database (**Table 2**).

Time Window Statistics:

- found: overall number of items found in window
- lowcnt: items with low number of values
- lowavg: items with mean which is near to zero

Length	1 day	1 hour
found	35,863	35,751
processed	4593	1079
ratio	12%	3%
ignored	31,269	34,671
lowstddev	29,281	30,266
lowavg	779	140
lowcnt	105	3794
lowcv	1101	468

Table 2. Time windows statistics example.

- lowstddev: items with small standard deviation
- lowcv: items with small coefficient of variation
- avg.(count): average count of history data per item
- avg.(CV): mean of coefficient of variation
- Level of Interest LOI_{tw} (1)

$$LOI_{tw} = 100\ avg(count)\ avg(CV) \frac{processed}{found} \tag{5}$$

4. *Hosts*: Just after Time Window statistics, Host statistics are computed. Inside each Time Window, all Hosts are found, Items are categorized for them and after this, statistics are computed.

Host in Zabbix can be any network device. Herein statistics makes a glue between Items and Hosts so it is possible to do Host specific computations in Time Windows. Example of Host statistics is in **Table 3**.

5. *Host statistics:*

- Number of Items found for Host in Time Window *items*
- Number of History rows for Host in Time Window *cnt*
- Level of Interest LOI_{hs}

$$LOI_{hs} = 100 \cdot \frac{count}{count_{max}} \tag{6}$$

The example of the command for the windows with IDs 46,965 and 47,241 looks like follows:

```
$ monda hs:show -w 46,965,47,241 --hostids 1,2.
```

This command yields, for example, the following data:

hostid	windowid	count	items	loi	type
1	46,965	8956	54	185	switch
2	47,241	1802	24	100	server

Table 3. Host statistics example.

6. *LOI update for all Time Windows*: As mentioned above, Zabbix has more information inside than only history. There are Events and Triggers. LOI for Time Windows are updated to respect this fact. More Events in Time Window lead to bigger LOI.

- Depending on Event time, LOI is increased for each Time Window affected (parameter *ec_item_increment_loi*)
- Depending on Item(s) which caused Event, LOI is increased for each Item (parameter *ec_window_increment_loi*)
- Depending on Hosts where Event was found, LOI is increased for Host (parameter *ec_host_increment_loi*)

7. *Correlation Statistics*: After marking Items and Time Windows with Loi, correlations are computed. From the principle described above, most interesting correlations are computed. It is not possible to compute all of them because combination of all Items is wide. Example of such statistics can be found in **Table 4**. There are two kinds of correlations to compute. One is for correlation between Items in specific Time Window and correlation of same Item in different Time Windows. First type is to analyze behavior of different values in same time while second is to analyze behavior of Item in different times. For example, to compare disk space usage in same hours of day.

windowid1	47,470	47,443	46,960
windowid2	4747	47,443	46,979
itemid1	server:cpu[user]	Switch:Ifln["125"]	Server:cpu[iowait]
itemid2	Server:cpu[iowait]	Switch:IfOut["77"]	Server:cpu[iowait]
corr	0.97	0.93	0.76
cnt	60	141	60
loi	155	94	398

Table 4. Correlation statistics example.

Correlation does not imply causality. But it is not important at this phase of analysis. Most important to know is if two Items correlated in Time Window and if so, how much significant this correlation was.

- Correlation within same Time Window

More Items are correlated in same Time Window. For example, how network interface load correlated with disk load at given Time Window.

- Correlation within same hour of day

It is common that correlations can occur even between different Time Windows and same Item. For example, there can be significant correlation of disk load on backup server at backup hours each day. Similar correlations can be found for weekly backups in given day of week. Instead of random processes which occurs in Time Windows, these correlations represent in most situations recurrent operations in network.

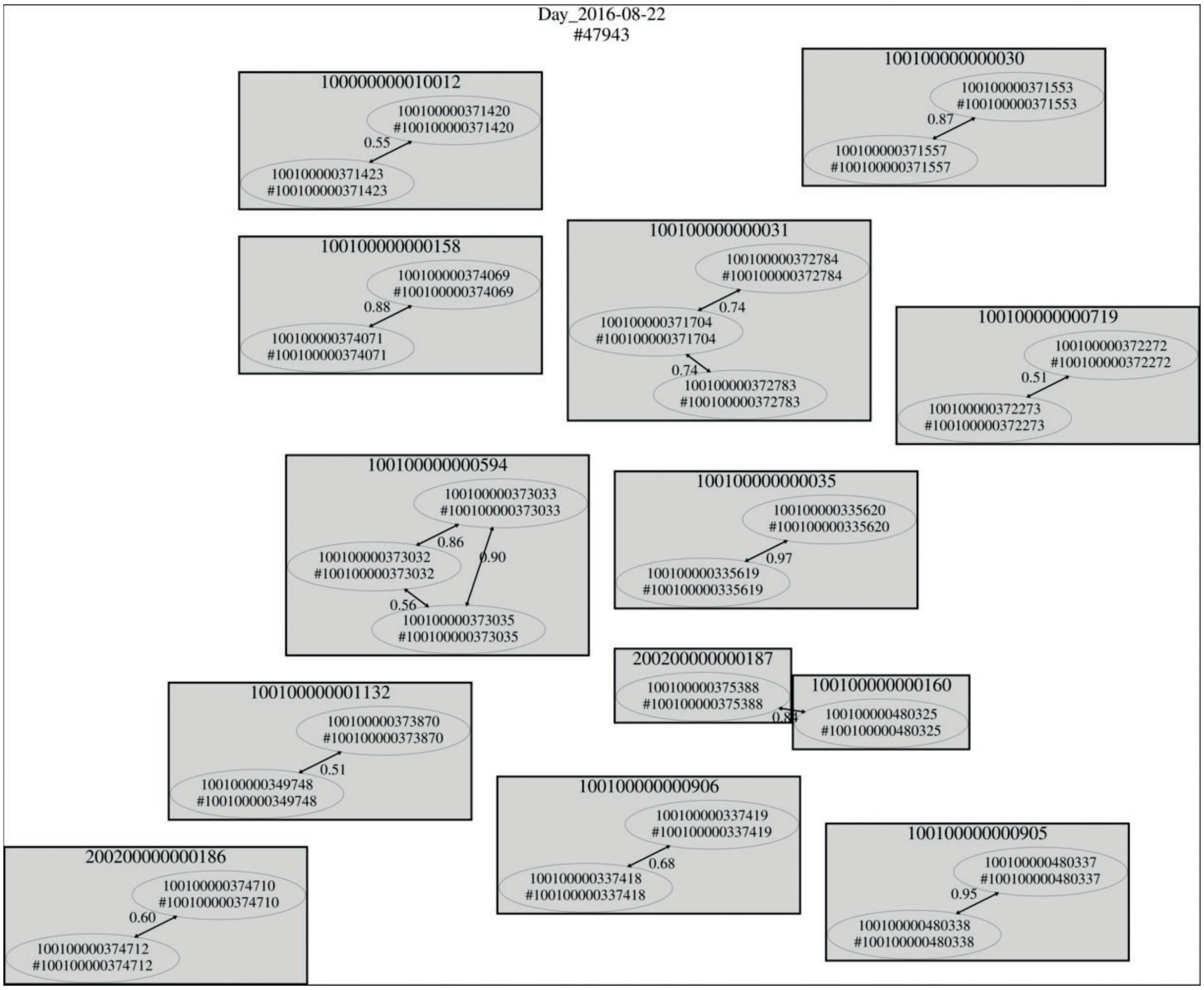

Figure 2. Correlations map: Servers.

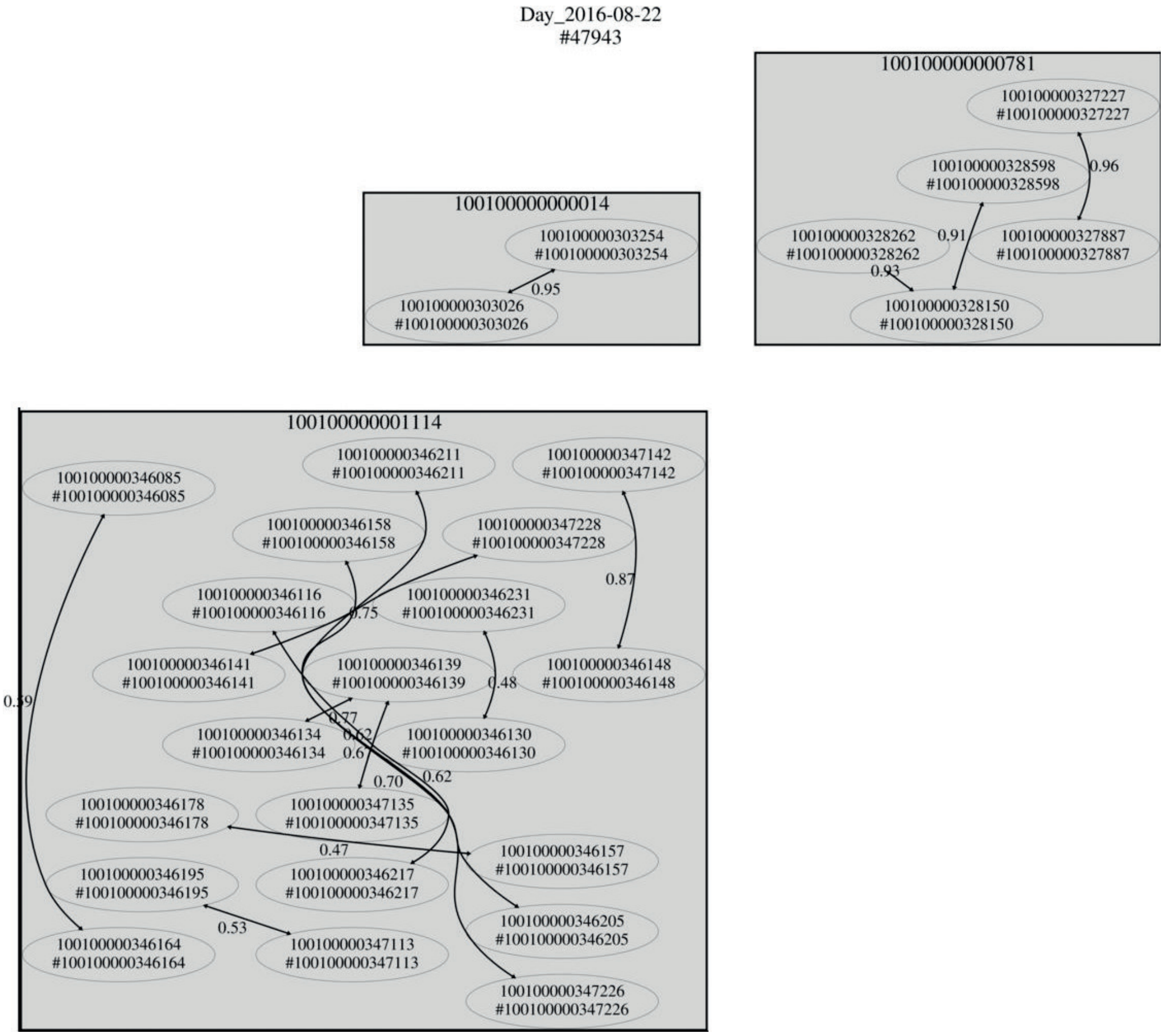

Figure 3. Correlations map: Switches.

- Correlation statistics
  - Number of values found for given correlation *cnt*
  - Pearson correlation coefficient *corr* which is from interval − 1 to 1
  - Level of Interest LOI_{hs}

$$LOI_{ic} = |corr| \cdot count_{w} \tag{7}$$

while count_w is number of next items which correlated in same time.

The example of the command for the windows with IDs 46,965 and 47,241 looks like follows:

```
$ monda ic:show -w 46,965,47,241 --itemids 1,2.
```

This command yields, for example, the following data:

5.3. Results of the algorithm

Example of map, created by applying algorithm on data of Silesian University is depicted in **Figure 2**, which shows correlation between servers and their items. Rectangles represents servers and ellipses their Items. Connectors between them represent correlation coefficient. Data are anonymized for security reasons but correlations are visible. Next fact which is obvious is that more Items correlate together.

Next example is correlation map of network switch and its ports, see **Figure 3**. Different ports correlate between each other according to network throughput. Switches are represented as rectangles, load on their ports are ellipses. Connectors are correlation between loads.

6. Monda

Monda [11] was designed and coded from scratch. It was designed to do most of the computations directly in SQL. This was crucial to speed up analysis. The result of the analysis is stored back to SQL tables, so it is possible to do next quick operation within it. Zabbix server was configured not to delete any data. Instead of deleting history data it creates partitions of SQL tables in regular intervals.

Monda is used as a tool which concentrates to the significant amount of data in Zabbix database and tries to find most interesting values and windows automatically. As mentioned, it is not possible to do a complete analysis with overall data inside in real time. And in fact, it is not needed. A lot of data in monitoring system are not interesting.

Monda never copies data from Zabbix. Instead of it, it uses algorithms and procedures which operate inside data and copy statistical results into Monda database. At this time, Monda has approximately 6200 rows of code.

Overall design rule was not to affect Zabbix server availability or performance. Zabbix uses its tables very often and utilizes SQL server by itself. From this reason, it was crucial to take care of all Monda operations to work in most situations in idle time of Zabbix server. Next, it was needed to set SQL timeout for Monda queries. If Monda analysis takes more than 10 min per query, it stops automatically.

7. Conclusion

Interesting results were found during analysis. A new approach to identify network incidents was invented. We created software Monda which is Open-Source, and it can be used by anybody for subsequent kinds of analysis in Zabbix. Verification of methods was done on Silesian University data stored in the monitoring database.

7.1. General results

Data in monitoring system are interesting for subsequent analysis. Even if it is relatively complicated to choose right data and right intervals, data are suitable for prediction of some incidents. Monda can do pre-processing part very quickly and an effective way directly within SQL server. Anybody can write its analysis module to focus on specific incident or time. Algorithms used here are mainly based on logical assumptions which are derived from knowledge of monitoring system and its data.

Next assumption is that to do better analysis and prediction of incidents, the monitoring system must have more inputs about incidents on the network. In other words, more data related to security and statistics of systems, better analysis and prediction of incidents.

7.2. Future improvements

More information about stored data and their source means better pre-processing of data. One of the improvements could be a manual description of Items inside Zabbix so pre-processor could know right ranges for given Items.

Next, it would be nice if Zabbix could do data approximation on historical data. Zabbix deletes data from history after configured amount of days and computes trends from it. So we can see minimum, maximum and average in hour intervals. If Zabbix uses approximation function, it is possible to describe data at summarized intervals better.

It is possible to use SOM in future for better fingerprinting of Hosts. But it needs more investigations and more data of separated Zabbix servers to do so.

Acknowledgements

The research received a financial support from the SGS grant no. SP2017/174, VSB: Technical University of Ostrava, Czech Republic. Authors would like to thank to Silesian University for making Zabbix server data available.

Author details

Lukáš Macura[1]*, Jan Rozhon[2] and Jerry Chun-Wei Lin[3]

*Address all correspondence to: macura@opf.slu.cz

1 School of Business Administration in Karvina, Silesian University in Opava, Karvina, Czech Republic

2 Faculty of Electrical Engineering and Computer Science, VSB—Technical University of Ostrava, Ostrava, Czech Republic

3 School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, Guangdong, China

References

[1] Fazio P, Tropea M. A new Markovian prediction scheme for resource reservation in wireless networks with mobile hosts. Advances in Electrical and Electronic Engineering. 2012;**10**(4):204-210

[2] Fazio P, Tropea M, Marano S. A distributed hand-over management and pattern prediction algorithm for wireless networks with mobile hosts. In: Proc. 9th International Wireless Communications and Mobile Computing Conference (IWCMC), Sardinia, pp. 294-298, 2013

[3] De Rango, Tropea M, Provato A, Santamaria AF, Marano S. Multi-constraints routing algorithm based on swarm intelligence over high altitude platforms. Studies in Computational Intelligence. 2007;**129**:409-418

[4] De Rango F, Tropea M, Provato A, Santamaria AF, Marano S. Minimum hop count and load balancing metrics based on ant behavior over HAP mesh. In: Proc. IEEE GLOBECOM 2008, New Orleans, pp. 1-6, 2008

[5] Celeda P, Kovacik M, Konicek T, et al. FlowMon Probe. Networking Studies. 2006

[6] Singh N, Jain A, Raw RS, Raman R. Detection of web-based attacks by analyzing web server log files. In: Networking, and Informatics. Advances in Intelligent Systems and Computing, Vol. 243. Springer; 2014

[7] Safarik J, Voznak M, Rezac F, Macura L. IP telephony server emulation for monitoring and analysis of malicious activity in VOIP network. Komunikacie. 2013;**15**(2A):191-196

[8] Safarik J, Partila P, Rezac F, Macura L, Voznak M. Automatic classification of attacks on IP telephony. Advances in Electrical and Electronic Engineering. 2013;**11**(6):481-486

[9] Safarik J, Voznak M, Rezac F, Macura L. Malicious traffic monitoring and its evaluation in VoIP infrastructure. In Proc. 35th Int. Conference on Telecommunications and Signal Processing, TSP 2012, Art. No. 6256294, pp. 259-262, 2012

[10] David N, Reshef N, Yakir A, et al. Detecting novel associations in large data sets. Science. 2011;**334**(6062):1518-1524

[11] Open-Source tool MONDA, data analyzing in monitoring system Zabbix. URL https://github.com/limosek/monda/

[12] Open-Source tool ZABBIX, the network monitoring SW. URL http://www.zabbix.com/