Detection of Undesired Events on Real-World SCADA Power System through Process Monitoring

Md. Turab Hossain [1], Md. Shohrab Hossain[1], Husnu S. Narman[2]
[1]Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Bangladesh
[2]Weisberg Division of Computer Science, Marshall University, Huntington, WV, USA
Email: turab.cse59@gmail.com , mshohrabhossain@cse.buet.ac.bd, narman@marshall.edu

***Abstract*—A Supervisory Control and Data Acquisition (SCADA) system is a common industrial process automation system which is used to collect data from instruments and sensors located at remote sites and to transmit data at a central site for either monitoring or controlling purpose. Most of the existing works on SCADA system focused on simulation-based study which cannot always mimic the real world situations. We propose a novel methodology that analyzes SCADA logs on offline basis and helps to detect process-related threats. Process related threat takes place when an attacker gains user access and performs malicious actions. We conduct our experiments on a real-life SCADA system of a Power transmission utility. Our proposed methodology will automate the analysis of SCADA logs and systemically identify undesired events. Moreover, it will help to analyse process-related threats caused by user activity. Several test case study suggest that our approach is effective in detecting undesired events that might caused by possible malicious occurrence.**

Keywords: SCADA, monitoring, malicious actions, undesired events, logs, process-related threats

I. Introduction

SCADA is a control system architecture for high-level process supervisory management in different critical infrastructures. This system comprises of computers, networked data communications and graphical user interfaces. It is the core of electric power system. SCADA systems have historically been isolated from other computing resources.

SCADA systems monitor and control mission-critical equipment and infrastructure. Failures in the security or safety of critical infrastructures can impact mass people and cause massive damages to industrial facilities. On May 2002, hacking into the Queensland computerised waste management system an attacker caused millions of litres of raw sewage to spill out into local parks, rivers and even ground of a hotel. A recent survey [1] states that current critical infrastructures are not sufficiently protected against Cyber threats. Nearly 60% of executives at critical infrastructure operators stated that they lack appropriate controls to protect their environments from security threats [2].

To detect anomalous behaviour in SCADA systems, there have been several works that are based on network traffic inspection [3], analyzing data readings [4] and validating protocol specifications [5]. However, process-related attacks typically cannot be detected by observing network traffic or protocol specifications in the system. Besides, having clear understanding about the user action, one needs to analyze route of the data. Bigham et al. [6] proposed a way of anomaly detection in SCADA system through taking periodic snapshots of power load reading in a grid system and compared it to check whether a snapshot varies significantly from expected proportions. However, data readings give a low-level view of the process and do not always provide user tractability. On the other hand, SCADA log gives a high-level view of industrial process and provide traceability. Again there are some notable works regarding network anomaly detection [7], [8]. However, they did not conduct real experiments on the SCADA system, rather conducted experiments on the log events generated from the testbed environment. Thus, most of the existing works focused on simulation-based study on SCADA systems, which sometimes cannot mimic the real world situations. Therefore, it is essential to study in real SCADA system so that the real world situations get reflected. There exists one work [9] that tried to find undesirable events in water management SCADA system that deals with a dataset different from the power system SCADA dataset. To the best of our knowledge, there exists no previous work that conducts experiments on real-world power system SCADA. Such experiments on real world SCADA is very essential to extract undesired events for the detection of possible malicious activities. This work is *first such work* that deals with real world power system SCADA to detect undesired events.

The main *contributions* of this work are: i) providing a semi-automated approach of log processing on a real-life power system SCADA, ii) analyze large amount of data and automatically categorize the less frequent patterns (serious anxiety, moderate, low anxiety and no anxiety), thereby avoiding manual interventions.

We have used quantitative approach for process monitoring. The available dataset contains one month log history of SCADA EMS application. Data preprocessing removes unwanted data and extract remaining data into a new file in a structured way. Then appropriate attributes are chosen to construct pattern. Two different algorithms, namely Apriori (with candidate generation) and FP-growth (without candidate generation) are used to find less frequent patterns for analysis.

The proposed tool based on our mining approach can be applied in power system SCADA system. This tool can help

978-1-7281-9656-5/20/$31.00 ©2020 IEEE

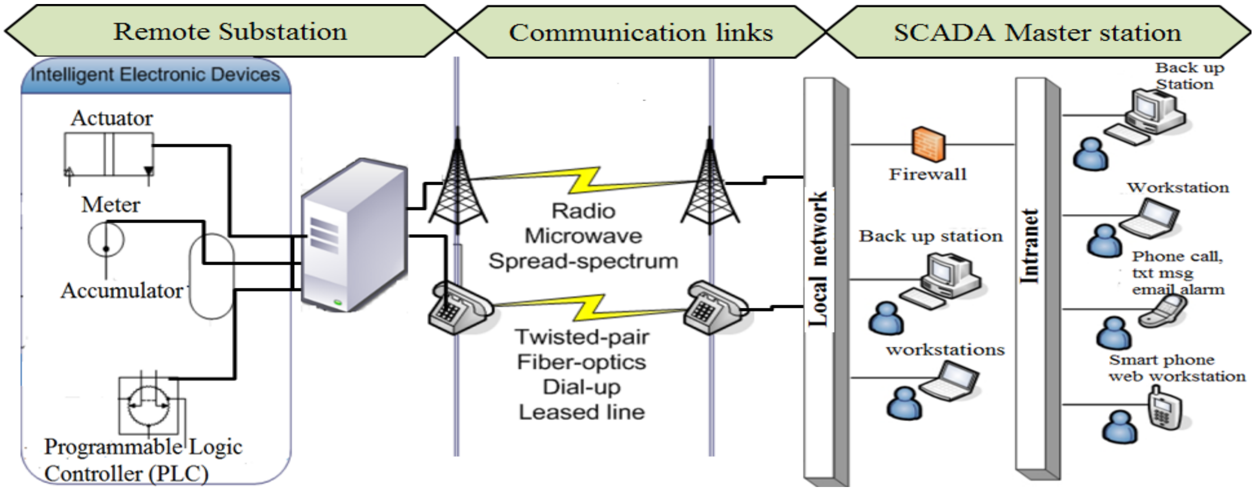


Fig. 1: SCADA System.

engineers extract less frequent patterns as well as undesired events with categorization according to their severity level. Power system engineers will then run the analysis offline and it will help them to decide which events need to be analyzed for detection of possible malicious occurrence.

The rest of the paper is organized as follows. SCADA system components and its architecture are explained in Section II. Section III describes the proposed mining approach. Implementation details are explained in Section IV. Section V describes the results of our approach. Finally, Section VI has the concluding remarks.

II. SCADA System

SCADA is one of the solutions available for data acquisition, monitor and control systems covering large geographical areas. Plant in several industries, such as power plants, oil and gas refining, water and waste control, telecommunications, etc use SCADA system for their monitoring and control system.

A. SCADA components

Fig. 1 shows the components that are available in a typical power system SCADA which includes SCADA master/control center, operator workstations, Communication links and remote stations.

- Remote Terminal unit (RTU): An RTU or Remote Terminal Unit is a standalone data acquisition and control unit, which monitors and controls equipment at some remote location from the central station. it is generally microprocessor based.
- Master Terminal Units (MTUs): A central host servers or server is called Master Terminal Unit. The central base station can be connected to a local area network with Internet access which permits other computers to be added to the system as backup base stations or mobile workstations.
- Communications System: The communication network transfers data among central host computer servers and the field data interface devices and control units.

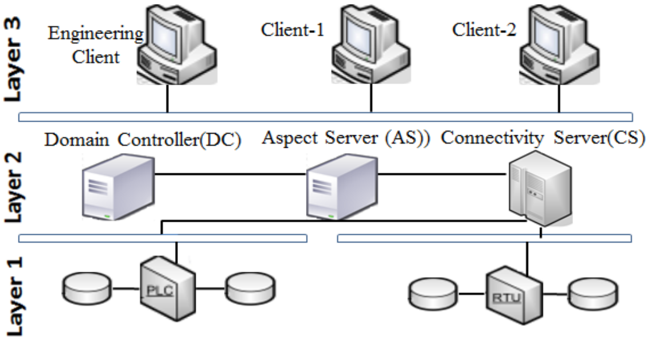


Fig. 2: Typical SCADA layered architecture.

- Operator Workstations: these are the computer terminals consisting of standard HMI (Human Machine Interface), networked with a central host computer and different software.

B. SCADA System Architecture

Fig. 2 shows a typical SCADA layered architecture. It consists of three layers:

- Layer 1: Layer 1 consists of field devices which include remote terminal units(RTUs) and programmable logic controller(PLCs). Layer 1 devices convert analog data to digital and transmits the digital data through communication channel.
- Layer 2: Layer 2 consists of different types of server like Aspect sever(AS), Connectivity server(CS), Domain controller(DC) etc. Servers collect and analyze values sent from the field devices.
- Layer 3: Layer 3 consists of client machines that interact with the server through terminals. Client runs different applications like alarms, real time networking, state estimators, contingency analysis, etc.

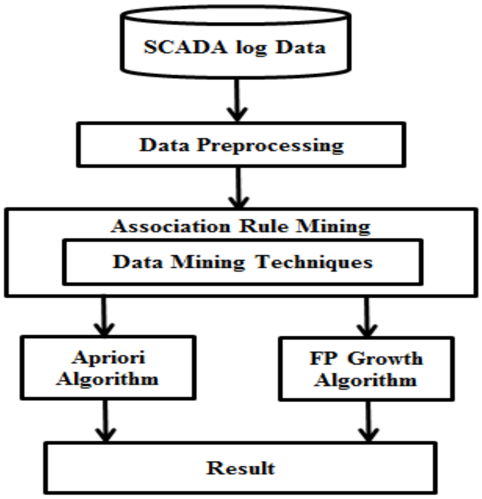


Fig. 3: System flow diagram.

III. Proposed Approach

The flow diagram of our proposed approach is shown in Fig. 3. It consists of several steps including raw data collection, data pre-processing, association rule mining techniques (Apriori and FP-growth) and finally, the result. After raw data collection, pre-processing steps removes unwanted data and combines them into a structured file format. Then, two popular pattern mining algorithms (Apriori and FP-growth) are used on the structured data to find out undesired events. Details about these steps are discussed in the following subsections.

A. System logs

System logs capture information about the events like status update, configuration changes, condition changes, user actions etc. A lot of system logs are generated per day. system logs are of two kinds 1) logs that are generated from the direct actions of the user and 2) logs that are generated as a consequence of the previous events. The first type of log includes time, location, user, event type of the event while the second type of log is generated as a consequence of future event it does not contain user information. The available dataset contains one month logs of SCADA EMS application. The log consists of ten attributes which are EventID, EventTimeStamp, SCADA category, TOC, AOR, Priority code, Substation, Device Type, Device and event Message. The detailed about these attributes will be discussed in Section III-D.

B. Log Mining

Success of any log mining depends on its context [10]. We can determine a set of patterns that are regular in their presence and frequency. If a pattern suddenly changes its regularity, this implies that a possible attack is taking place. On the other hand, if a regular pattern becomes less frequent, this can imply that a device is malfunctioning or has been reconfigured. So the objective is to apply mining on the SCADA logs to find the regularity of the patterns. Over a large amount of time, frequent behavior is likely to be normal as logs for usual system activity are normally frequent. [6], [11], [12].

C. Algorithms for frequent pattern mining

Two popular frequent pattern mining algorithms is used. The first one is Apriori that uses candidate generation and other is FP-growth that does not use candidate generation. While storage structure in Apriori is array based, storage structure in FP-growth is tree based. Search type in Apriori is BFS while search type in FP-growth is divide and conquer: 'join and prune' technique is used in Apriori while FP-growth constructs conditional frequency pattern tree which satisfy minimum support count. Fp-growth requires less memory while Apriori requires a large amount of memory. Finally as FP-growth requires only 2 scans, runing time of FP-growth is found much faster.

D. Data Collection

The dataset is collected from the SCADA system of a power utility. Table I shows the characteristics of the collected dataset. It contains one month log of the month May 2018. The dataset contains ten attributes.

TABLE I: Collected dataset.

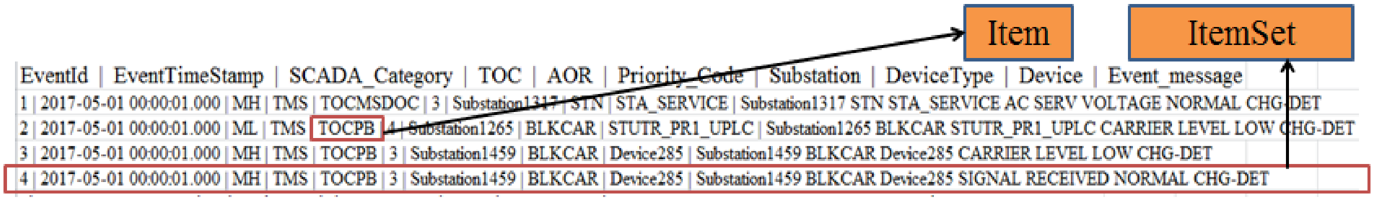
Dataset name	Number of instances	Number of attributes	Time duration
Power system event log	57,58,500	10	1 month

A snapshot of the dataset is shown in the Fig. 4. In pattern mining each cell value of an attribute is called an item, a set of items is called an itemset. Item and itemset are shown in Fig. 4. The dataset contains ten attributes in the form EventID | EventTimeStamp | SCADA category | TOC | AOR | Priority code | Substation | Device Type | Device | event Message.

1) EventId: Numerical value, Count of the event.
2) EventTimeStamp: date and time of the event.
3) SCADA Category: like Analog, SheadLoad, Bkr-Fail, D-switch, Fdr-brkr, Station etc.
4) TOC: indicates source system (ignored).
5) AOR: Area of Responsibility.(Which operators)
6) Priority code : Priority of the event.
7) Substation: Event in which Substation.
8) Device Type: Device type of the event originator.
9) Device : Event generator device.
10) Event-message : event message in the form Substation + Device Type + Device + Message.

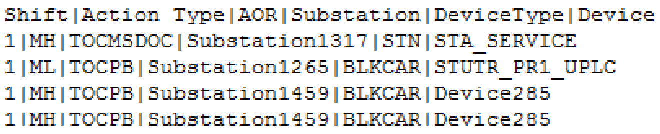
E. Preprocessing

This technique involves removing of the unwanted data and splitting of data into a structured file format. As server log usually do not have right format, there is a necessary for pre-processing technique [13]. After preprocessing six attributes are extracted which is shown in Fig. 5. We transform the Timestamp attribute to represent usual working shifts in the company. In this way we aggregate a time series attribute into



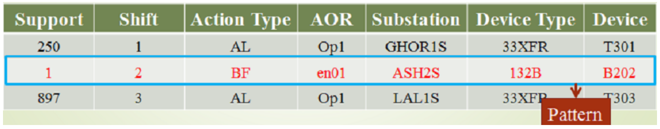
```
EventId | EventTimeStamp | SCADA_Category | TOC | AOR | Priority_Code | Substation | DeviceType | Device | Event_message
1 | 2017-05-01 00:00:01.000 | MH | TMS | TOCMSDOC | 3 | Substation1317 | STN | STA_SERVICE | Substation1317 STN STA_SERVICE AC SERV VOLTAGE NORMAL CHG-DET
2 | 2017-05-01 00:00:01.000 | ML | TMS | TOCPB | 4 | Substation1265 | BLKCAR | STUTR_PR1_UPLC | Substation1265 BLKCAR STUTR_PR1_UPLC CARRIER LEVEL LOW CHG-DET
3 | 2017-05-01 00:00:01.000 | MH | TMS | TOCPB | 3 | Substation1459 | BLKCAR | Device285 | Substation1459 BLKCAR Device285 CARRIER LEVEL LOW CHG-DET
4 | 2017-05-01 00:00:01.000 | MH | TMS | TOCPB | 3 | Substation1459 | BLKCAR | Device285 | Substation1459 BLKCAR Device285 SIGNAL RECEIVED NORMAL CHG-DET
```

Fig. 4: Dataset.



```
Shift|Action Type|AOR|Substation|DeviceType|Device
1|MH|TOCMSDOC|Substation1317|STN|STA_SERVICE
1|ML|TOCPB|Substation1265|BLKCAR|STUTR_PR1_UPLC
1|MH|TOCPB|Substation1459|BLKCAR|Device285
1|MH|TOCPB|Substation1459|BLKCAR|Device285
```

Fig. 5: Preprocessed dataset.



Support	Shift	Action Type	AOR	Substation	Device Type	Device
250	1	AL	Op1	GHOR1S	33XFR	T301
1	2	BF	en01	ASH2S	132B	B202
897	3	AL	Op1	LAL1S	33XFR	T303

Fig. 6: Desired pattern selection.

a 3-value discrete format that is more suitable for mining workload patterns. In this case, working shift 1 covers all events occurring between 00:00 and 08:59hrs. Working shift 2 includes events occurring between 09:00 and 16:59hrs. Working shift 3 includes events occurring between 17:00 and 23:59hrs.

F. Pattern Discovery

If the occurrence of an itemset I exceeds a predefined minimum support count threshold, then I is a pattern [14]. Adding support count(that defines the number of time a pattern appears) with the six attributes got after preprocessing steps we construct the desired pattern. Fig. 6 shows the desired pattern selection.

G. Output pattern

Two algorithm Apriori and FP-Growth are used on the extracted dataset to find less frequent pattern. Two types of pattern are found. Some are regular and some are irregular [15]. Since SCADA system polls data from remote substation after some certain intervals, same patterns repeat again and again. So the number of irregular patterns are very few. By analyzing the regularity of the pattern we try to find the minimum threshold of the support count. After this minimum value, pattern counts moves to a larger value.

IV. Implementation Details

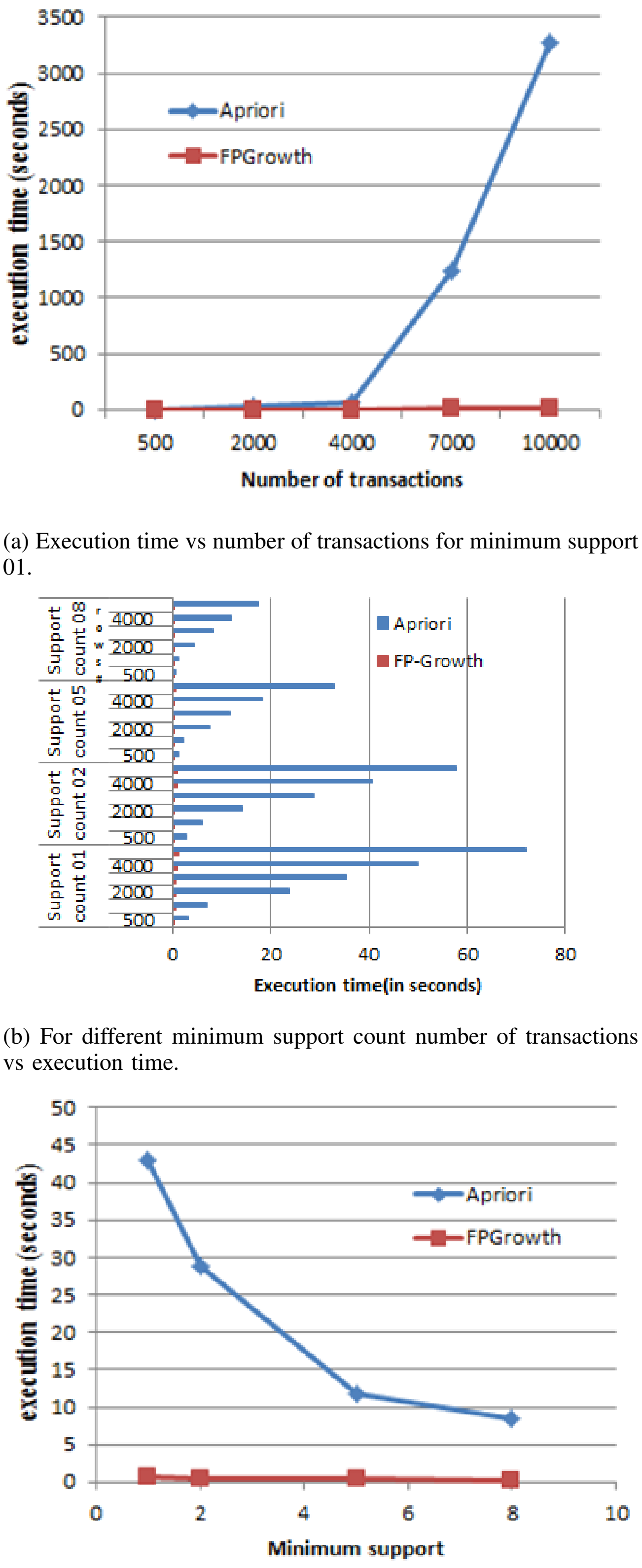
We have used the dataset of real-world power system SCADA. It includes one month of events as logged by an Energy Management System application owned by a power system utility. The Energy Management system (also called SCADA/EMS or EMS/SCADA) is a computer aided system tool used by operators of electric utility grids to optimize, monitor, and control the performance of the transmission and generation system. The total dataset contains 57,58,500 number of rows of 31 days (May 2018). Events in the data are arranged in rows where each row is a unique event, except the first row which gives names of the columns. Data of each date is extracted to a separate file. Each day-wise file contains about 21,5000 entry each.
We use pattern mining algorithms to extract the least frequent event patterns from SCADA log. Hundreds of algorithms have been proposed for sparse/dense data, many rows/columns, data fits/does not fit in memory etc. Among these we can filter out most useful methods which we can categorize them as scalable methods for mining frequent patterns. Apriori and FP-growth are two of the major approaches. Apriori uses candidate generation [16]. FP-growth doesn't use candidate generation [14]. For mining a k-size itemset, an algorithm that uses candidate generation may need up to 2^k scans of the data set while an algorithm that does not use candidate generation typically requires only two scans of the data set.

V. Results

A. Performance evaluation and methodology selection

We have applied two popular data mining algorithm Apriori and FP-Growth. Apriori algorithmic program takes longer time in compare to FP-Growth algorithm. Fig. 7a shows the execution time vs number of transactions graph for the two algorithms. With number of transactions increasing, execution time of Apriori become exponential. Fig. 7b shows number of transactions vs execution time graph for different minimum support count. Here it is observed that for 500, 1000, 2000, 3000, 4000 and 5000 number of transactions, Fp-Growth algorithm runs much faster than Apirori. Again Fig. 7c shows execution time vs minimum support count graph for fixed number of transactions. Here it is observed that with the increasing of minimum support count, execution time of Apriori reduces a lot. It is because with the increase in minimum support count, the size of candidate generation reduces. All the three figure reveals that the time taken to execute the FP-growth algorithm is extremely less compared to Apriori algorithm for any Support level. The reason is as Apriori uses candidate generation, it requires to scan database again and again [17].

For processing 10,000 rows Apriori takes more than 50 minutes which is unacceptable. So FP-Grpwth is used for pattern regularity analysis.

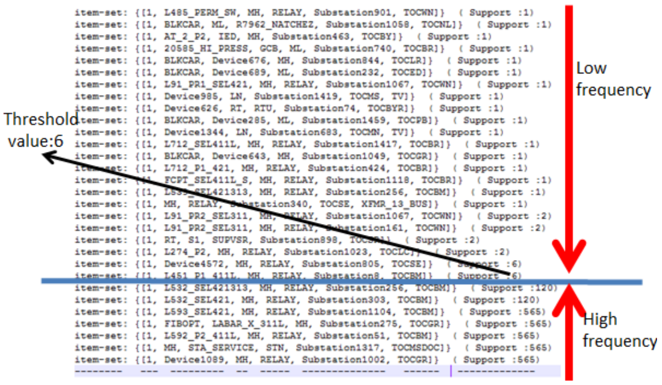


(a) Execution time vs number of transactions for minimum support 01.

(b) For different minimum support count number of transactions vs execution time.

(c) For fixed number of transaction(3000) execution time vs minimum support count.

Fig. 7: performance comparison between Apriori and FP-Growth.



```
item-set: {[1, L485_PERM_SW, MH, RELAY, Substation901, TOCWN]}  ( Support :1)
item-set: {[1, BLKCAR, ML, R7962_NATCHEZ, Substation1058, TOCNL]}  ( Support :1)
item-set: {[1, AT_2_P2, IED, MH, Substation463, TOCBY]}  ( Support :1)
item-set: {[1, 20585_HI_PRESS, GCB, ML, Substation740, TOCBR]}  ( Support :1)
item-set: {[1, BLKCAR, Device676, MH, Substation844, TOCLR]}  ( Support :1)
item-set: {[1, BLKCAR, Device689, ML, Substation232, TOCED]}  ( Support :1)
item-set: {[1, L91_PR1_SEL421, MH, RELAY, Substation1067, TOCWN]}  ( Support :1)
item-set: {[1, Device985, LN, Substation1419, TOCMS, TV]}  ( Support :1)
item-set: {[1, Device626, RT, RTU, Substation74, TOCBYR]}  ( Support :1)
item-set: {[1, BLKCAR, Device285, ML, Substation1459, TOCPB]}  ( Support :1)
item-set: {[1, Device1344, LN, Substation683, TOCMN, TV]}  ( Support :1)
item-set: {[1, L712_SEL411L, MH, RELAY, Substation1417, TOCBR]}  ( Support :1)
item-set: {[1, BLKCAR, Device643, MH, Substation1049, TOCGR]}  ( Support :1)
item-set: {[1, L712_P1_421, MH, RELAY, Substation424, TOCBR]}  ( Support :1)
item-set: {[1, FCPT_SEL411L_S, MH, RELAY, Substation1118, TOCBR]}  ( Support :1)
item-set: {[1, L533_SEL421313, MH, RELAY, Substation256, TOCBM]}  ( Support :1)
item-set: {[1, MH, RELAY, Substation340, TOCSE, XFMR_13_BUS]}  ( Support :1)
item-set: {[1, L91_PR2_SEL311, MH, RELAY, Substation1067, TOCWN]}  ( Support :2)
item-set: {[1, L91_PR2_SEL311, MH, RELAY, Substation161, TOCWN]}  ( Support :2)
item-set: {[1, RT, S1, SUPVSR, Substation898, TOCBR]}  ( Support :2)
item-set: {[1, L274_P2, MH, RELAY, Substation1023, TOCLC]}  ( Support :2)
item-set: {[1, Device4572, MH, RELAY, Substation805, TOCSE]}  ( Support :6)
item-set: {[1, L451_P1_411L, MH, RELAY, Substation8, TOCBM]}  ( Support :6)
item-set: {[1, L532_SEL421313, MH, RELAY, Substation256, TOCBM]}  ( Support :120)
item-set: {[1, L532_SEL421, MH, RELAY, Substation303, TOCBM]}  ( Support :120)
item-set: {[1, L593_SEL421, MH, RELAY, Substation1104, TOCBM]}  ( Support :565)
item-set: {[1, FIBOPT, LABAR_X_311L, MH, Substation275, TOCGR]}  ( Support :565)
item-set: {[1, L592_P2_411L, MH, RELAY, Substation51, TOCBM]}  ( Support :565)
item-set: {[1, MH, STA_SERVICE, STN, Substation1317, TOCMSDOC]}  ( Support :565)
item-set: {[1, Device1089, MH, RELAY, Substation1002, TOCGR]}  ( Support :565)
--------   ---  ---------  --  -----  --------------   ------ | -------------
```

Fig. 8: Sample output after pattern mining.

B. *Defining threshold in less frequent pattern mining*

Since the objective is to find less frequent pattern for unwanted events recognition, we set minimum support count value to 1 for the algorithm. After analyzing logs per day, among the less frequent patterns, about 30-40 patterns are identified that can be analyzed by the engineers for possible malicious events. Thus, it is essential to define ’less frequent’. According to the power transmission utility engineers, it is observed that the threshold of minimum support should be determined dynamically. As polling from different remote substations occurs after certain interval, almost all the pattern appears with a large number of support count. After a particular value, the support count value of the remaining patterns changes to a higher value. Let us call this particular value as natural threshold value. All the pattern having support count less than or equal to this value are less frequent patterns while patterns having higher support count than this value are high frequent patterns. For example, Fig. 8 shows a natural threshold count value 06. Support count increases a lot for the patterns having support greater than 6. Table II shows how the gap between patterns of low and high frequency changes over a week. The natural threshold value of those 7 days can be determined as 9, 10, 10, 7, 10, 8 & 5, respectively.

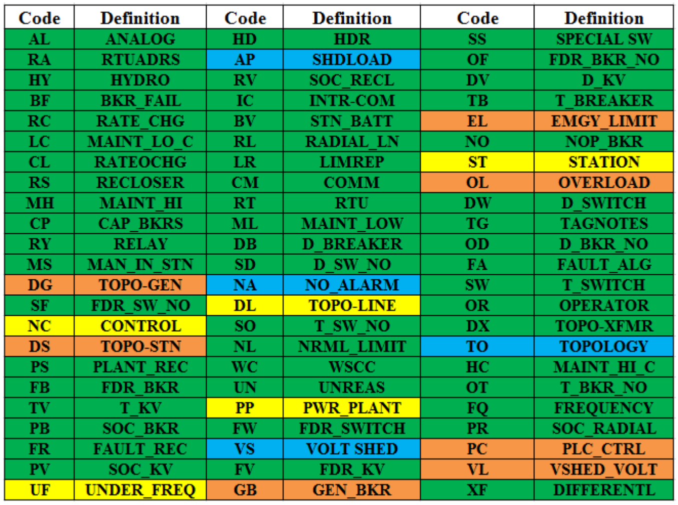
C. *Detection of anomalous occurrence*

In order to detect probable malicious events from less frequent patterns, we have consulted with electrical engineers form the power transmission utility and categorized SCADA events into four categories: serious anxiety, moderate anxiety, low anxiety and no anxiety events. These categorization is shown in Fig. 9, Where ’blue’ events are serious anxiety events, ’orange’ are moderate, ’yellow’ are low anxiety and ’green’ are no anxiety events.

Fig. 10a shows the severity level tested on 10,000 rows. 481 patterns are found as less frequent of which no serious anxiety event is found, 4 patterns are found as moderate and 15 as low anxiety patterns, remaining 462 patterns are no anxiety patterns. Similarly, Fig. 10b shows the severity level tested on 20,000 rows. 755 patterns are found as less frequent of which

SUPPORT COUNT						
day1	day2	day3	day4	day5	day6	day7
1	1	1	1	1	1	1
2	2	2	2	2	2	2
4	3	3	7	6	8	5
6	6	4	83	10	71	26
9	10	10	522	36	169	98
91	70	81	874	71	272	150
196	105	205	1024	250	333	180
202	242	237	2023	265	456	271
248	412	357	2209	278	870	337
343	819	371	2956	411	976	462
473	912	552	3096	613	1120	502
547	957	554	4005	631	2394	615
635	1009	679	7607	1050	3479	1344
913	1056	970	10903	5050	3997	2029
959	5358	1093	11504	10021	5247	6348
—-	—-	—-	—-	—-	—-	—-

TABLE II: Frequency of pattern occurrences over one week of SCADA log.



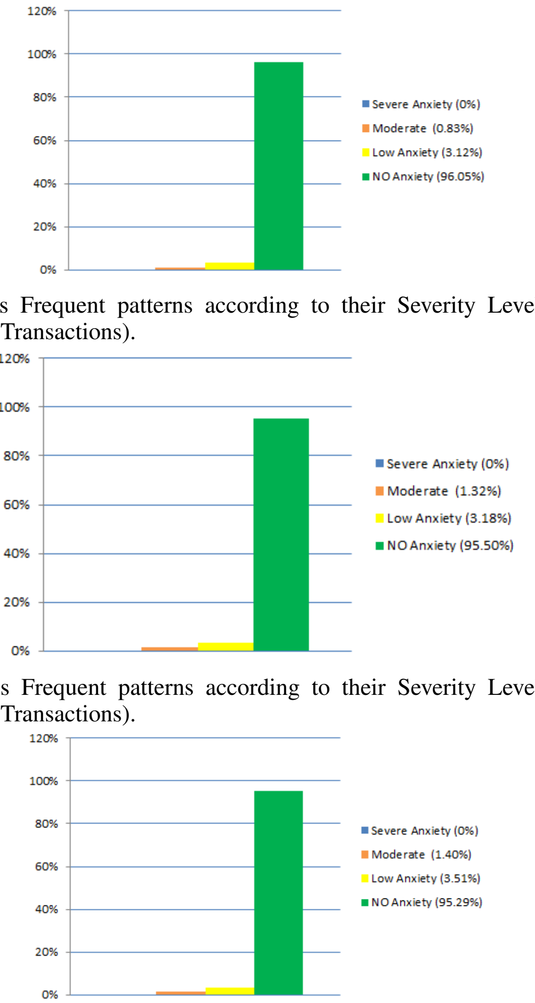
Code	Definition	Code	Definition	Code	Definition
AL	ANALOG	HD	HDR	SS	SPECIAL SW
RA	RTUADRS	AP	SHDLOAD	OF	FDR_BKR_NO
HY	HYDRO	RV	SOC_RECL	DV	D_KV
BF	BKR_FAIL	IC	INTR-COM	TB	T_BREAKER
RC	RATE_CHG	BV	STN_BATT	EL	EMGY_LIMIT
LC	MAINT_LO_C	RL	RADIAL_LN	NO	NOP_BKR
CL	RATEOCHG	LR	LIMREP	ST	STATION
RS	RECLOSER	CM	COMM	OL	OVERLOAD
MH	MAINT_HI	RT	RTU	DW	D_SWITCH
CP	CAP_BKRS	ML	MAINT_LOW	TG	TAGNOTES
RY	RELAY	DB	D_BREAKER	OD	D_BKR_NO
MS	MAN_IN_STN	SD	D_SW_NO	FA	FAULT_ALG
DG	TOPO-GEN	NA	NO_ALARM	SW	T_SWITCH
SF	FDR_SW_NO	DL	TOPO-LINE	OR	OPERATOR
NC	CONTROL	SO	T_SW_NO	DX	TOPO-XFMR
DS	TOPO-STN	NL	NRML_LIMIT	TO	TOPOLOGY
PS	PLANT_REC	WC	WSCC	HC	MAINT_HI_C
FB	FDR_BKR	UN	UNREAS	OT	T_BKR_NO
TV	T_KV	PP	PWR_PLANT	FQ	FREQUENCY
PB	SOC_BKR	FW	FDR_SWITCH	PR	SOC_RADIAL
FR	FAULT_REC	VS	VOLT SHED	PC	PLC_CTRL
PV	SOC_KV	FV	FDR_KV	VL	VSHED_VOLT
UF	UNDER_FREQ	GB	GEN_BKR	XF	DIFFERENTL

Fig. 9: Severity Level of different SCADA Event Category.

no serious anxiety event is found, 10 patterns are found as moderate and 24 as low anxiety patterns. Finally, Fig. 10c shows the severity level tested on 1 day log data. Here, again no serious anxiety event is detected. 782 patterns are found as less frequent of which 11 are detected as moderate and 24 patterns are detected as low anxiety patterns, remaining are no anxiety pattern. So at the end of the day stakeholders can analyze about 11 (moderate)+ 24 (low) = 35 patterns for possible malicious occurrence.

D. Baseline parameters

The baseline parameters used in the simulation are shown in Table III. We increased the number of transactions by 500 to 1000 and observed the results and system performance. ’222169’ was the maximum number of log entry found in a day. We varied the minimum support count value from 1 to 8. The execution time reduces with the increase in minimum support count value. Since, we are trying to find less frequent patterns, we argue that the minimum support count value should be set to 1. We found the support count threshold value varies from 4 to 9. It was observed that after this threshold value, pattern gets significantly higher frequency of occurrence. We argue that threshold value need not be set to more than 10. Since we are considering less frequent patterns, patterns having support count greater than this threshold need not be considered.



(a) Less Frequent patterns according to their Severity Level (on 10,000 Transactions).

(b) Less Frequent patterns according to their Severity Level (on 20,000 Transactions).

(c) Less Frequent patterns according to their Severity Level (on day1 Transactions).

Fig. 10: Less Frequent patterns according to their Severity Level.

TABLE III: Baseline parameters.

No of transactions	500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 5000, 10000, 20000, 222169
Minimum support count	1, 2, 5, 8
Support count threshold	9, 8, 6, 5, 4

E. Results summary

Finally, we propose to run the mining approach analysis offline. At the end of a day, stakeholders can run the analysis to detect potential threats. Testing has been performed on a machine with an Intel Core i5-5200U CPU at 2.2GHz and 8Gb of memory. The average running system performance shown in Table IV is achieved after applying the mining approach on different separate dated log file. The table contains nine columns. The first column shows the dataset information. The second column shows the number of less frequent logs and patterns found in a day. Among the less frequent events, the number of events and patterns require to be inspected is shown in the next column. The next four columns respectively shows the number of serious, moderate, low and no anxiety events as well as patterns found within these less frequent events. The next column shows the total number of unique items and the final column shows the total execution time.

TABLE IV: average system performance result(per day).

	less freq log(daily)	for inspection	serious	moderate	low	no anxiety	distinct items	total execution time(S)
number of events	3442	154	0	28	126	3288	2345	80.996
number of patterns	782	35	0	11	24	747		

VI. Conclusion and future work

SCADA system controls vital resources in every critical infrastructure sector. Therefore, the security of SCADA systems has been the subject of research, industrial practices and standardization for several years. However, currently there exists to monitoring tools to mitigate process-related threats that occur in power system SCADA. We have proposed a semi-automated approach of log processing for the detection of undesirable events that relate to user actions in power system SCADA. In our approach, we have proposed an analysis tool that extracts non-frequent patterns, which may be the result of an anomalous event. We conduct our experiments on real logs from the SCADA system of a Power transmission utility. We propose to run the mining approach analysis offline. Our results show that at the end of a day, stakeholders can run the analysis on the logs generated on that day and get 20-30 patterns on average to analyze for possible malicious occurrence. Although no serious anxiety events occurred in the log (one month log entry) we analyzed, some moderate anxiety events were detected which was found as the result of system mis-configurations done by the stakeholders. Again, our result shows that FP-Growth algorithm performs better than Apriori for any number of transactions. So for the data mining tool, FP-Growth will be used.

A large number of entries are generated on the log file per day. These huge logs are usually not analyzed by the engineers. As there is no tool currently available for analyzing purpose, manual checking is the only solution. But due to large amount of data, manual checking is not feasible. Our proposed tool will help the power system operation engineers to analyze SCADA log easily and detect possible process-related threats. Finally, we argue that SCADA logs represent interesting behaviour of SCADA system. We believe log analysis will be an indispensable part in our network defense strategy in future.

In future, we aim at experiencing the mining approach on bigger dataset and search for potential threats. Again in our approach, we address only single event or operation. Sequence of actions are not considered here. In future, we aim at addressing anomalous sequence of actions for power system SCADA in our proposed tool.

References

[1] W. Hurst, M. Merabti, and P. Fergus, "A survey of critical infrastructure security," 8th International Conference on Critical Infrastructure Protection (ICCIP), Arlington, TX, March 2014.

[2] E. Tara, "Majority of critical infrastructure orgs unprepared for attacks," https://www.infosecurity-magazine.com/news/majority-of-critical/, 03 April 2018.

[3] C. Balducelli, L. Lavalle, and G. Vicoli, "Novelty detection and management to safeguard information-intensive critical infrastructures," *Int. J. Emergency Management*, vol. 4, no. 1, pp. 88–103, 09 Feb 2007.

[4] Y. Liu, P. Ning, and M. Reiter, "False data injection attacks against state estimation in electric power grids," 16th ACM conference on Computer and communications security(CCS), New York, NY, USA, November 2009.

[5] C. Bellettini and J. Rrushi, "Vulnerability analysis of scada protocol binaries through detection of memory access taintedness," 8th IEEE SMC Information Assurance Workshop, IEEE Press, pp. 341–348, 2007.

[6] J. Bigham, D. Gamez, and N. Lu, "Safeguarding scada systems with anomaly detection," *2nd International Workshop on Mathematical Methods, Models and Architectures for Computer Network Security, LNCS 2776*, pp. 171–182, Springer Verlag, 2003.

[7] M. K. Islam, P. Hridi, M. S. Hossain, and H. S. Narman, "Network anomaly detection using lightgbm: A gradient boosting classifier," 30th International Telecommunication Networks and Applications Conference (ITNAC), 25-27 November 2020.

[8] T. Dipon, M. S. Hossain, and H. S. Narman, "Detecting network intrusion through anomalous packet identification," 30th International Telecommunication Networks and Applications Conference (ITNAC), 25-27 November 2020.

[9] D. Hadziosmanovic, D. Bolzoni, and P. Hartel, "A log mining approach for process monitoring in scada," *Journal of Information Security*, vol. 11, pp. 231–251, August 2012.

[10] A. Oliner and J. Stearley, "What supercomputers say: A study of five system logs," 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Edinburgh, UK, pp. 575–584, June 2007.

[11] K. Begnum and M. Burgess, "Principle components and importance ranking of distributed anomalies," *Machine Learning*, vol. 58, pp. 217–230, Feb 2005.

[12] R. Vaarandi, "Tools and techniques for event log analysis," PhD thesis, Tallinn University of Technology, 2005.

[13] Ristoski, Petar, C. Bizer, and H. Paulheim, "Mining the web of linked data with rapidminer," *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 35, pp. 142–151, December 2015.

[14] J. Han and M. Kamber, *Data mining : concepts and techniques*. San Francisco [u.a.]: Kaufmann, 2005.

[15] K. Dharmarajan and M. A. Dorairanjaswamy, "Analysis of fp-growth and apriori algorithms on pattern discovery from weblog data," in *International Conference on Advances in Computer Applications*, Coimbatore, India, 24-24 Oct 2016.

[16] "Fast algorithms for mining association rules in large databases," in *20th International Conference on Very Large Data Bases*, Morgan Kaufmann, 1994, pp. 487–489.

[17] M. Mythili and M. Shanavas, "Performance evaluation of apriori and FP-Growth algorithms," *Journal of Computer Application*, vol. 79, pp. 34–37, October 2013.