Event Pattern Identification in Anonymized System Logs

Siavash Ghiasvand§ and Florina M. Ciorba★
§Technische Universität Dresden, Germany ★University of Basel, Switzerland

1. Motivation and Challenges

Increasing size of computing systems (in terms of components) [1]

Increasing the amount of operational logs, produced by various components of computing systems

There are detectable patterns in the system logs.

Such patterns help system administrators detect irregular activities.

System logs may contain sensitive and confidential data (e.g., user credentials).

Protecting privacy is a major goal.

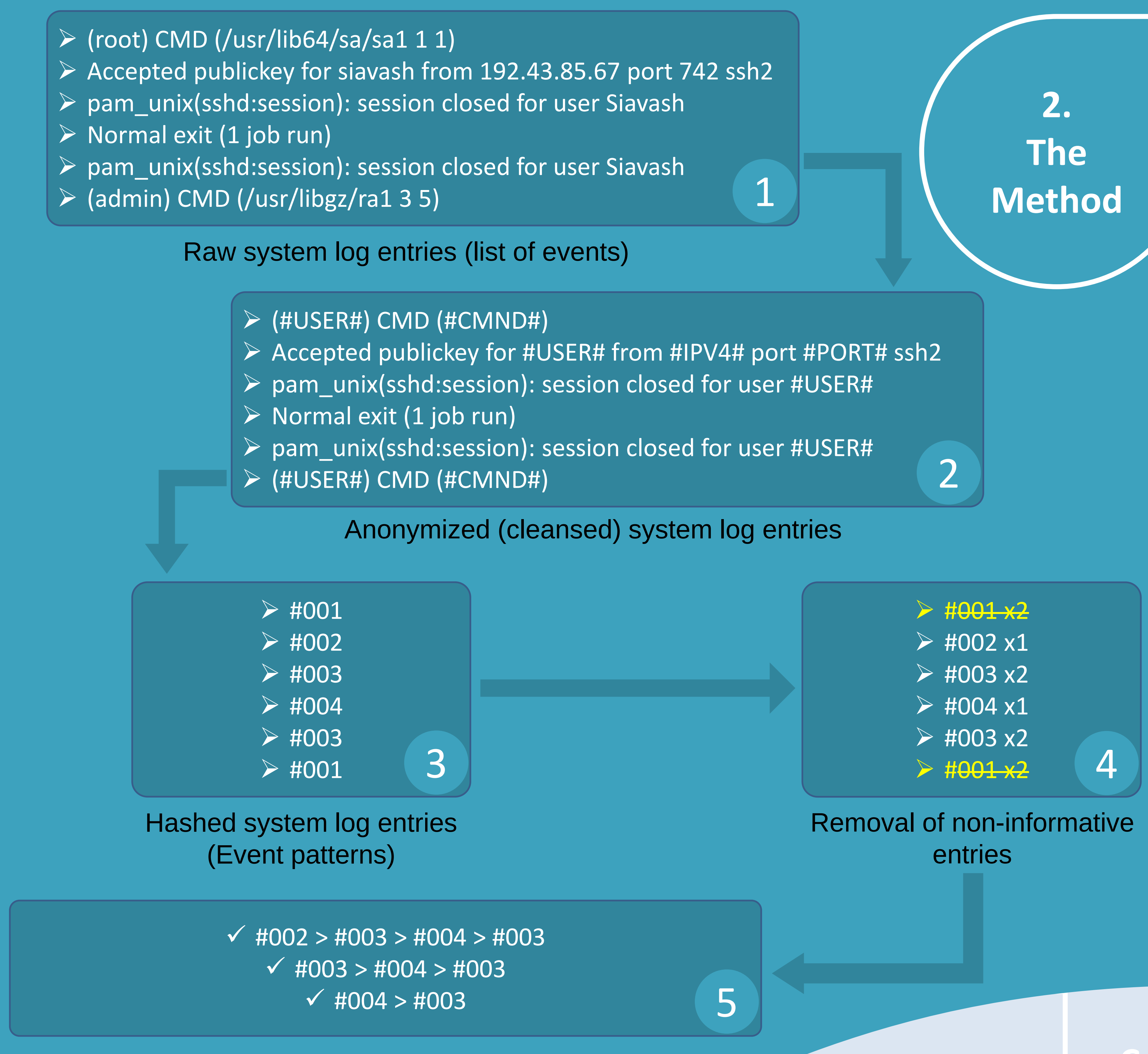
Data mining methods are well known for system log analysis [2].

Most data mining methods employ statistical approaches.

Anonymization can efficiently reducethe size (in Bytes) of system logs.

Data in anonymized logs is still useful for further analysis [3] (e.g., failure early-detection).

2. The Method



```
➢ (root) CMD (/usr/lib64/sa/sa1 1 1)
➢ Accepted publickey for siavash from 192.43.85.67 port 742 ssh2
➢ pam_unix(sshd:session): session closed for user Siavash
➢ Normal exit (1 job run)
➢ pam_unix(sshd:session): session closed for user Siavash
➢ (admin) CMD (/usr/libgz/ra1 3 5)
```

1 Raw system log entries (list of events)

```
➢ (#USER#) CMD (#CMND#)
➢ Accepted publickey for #USER# from #IPV4# port #PORT# ssh2
➢ pam_unix(sshd:session): session closed for user #USER#
➢ Normal exit (1 job run)
➢ pam_unix(sshd:session): session closed for user #USER#
➢ (#USER#) CMD (#CMND#)
```

2 Anonymized (cleansed) system log entries

```
➢ #001
➢ #002
➢ #003
➢ #004
➢ #003
➢ #001
```

3 Hashed system log entries (Event patterns)

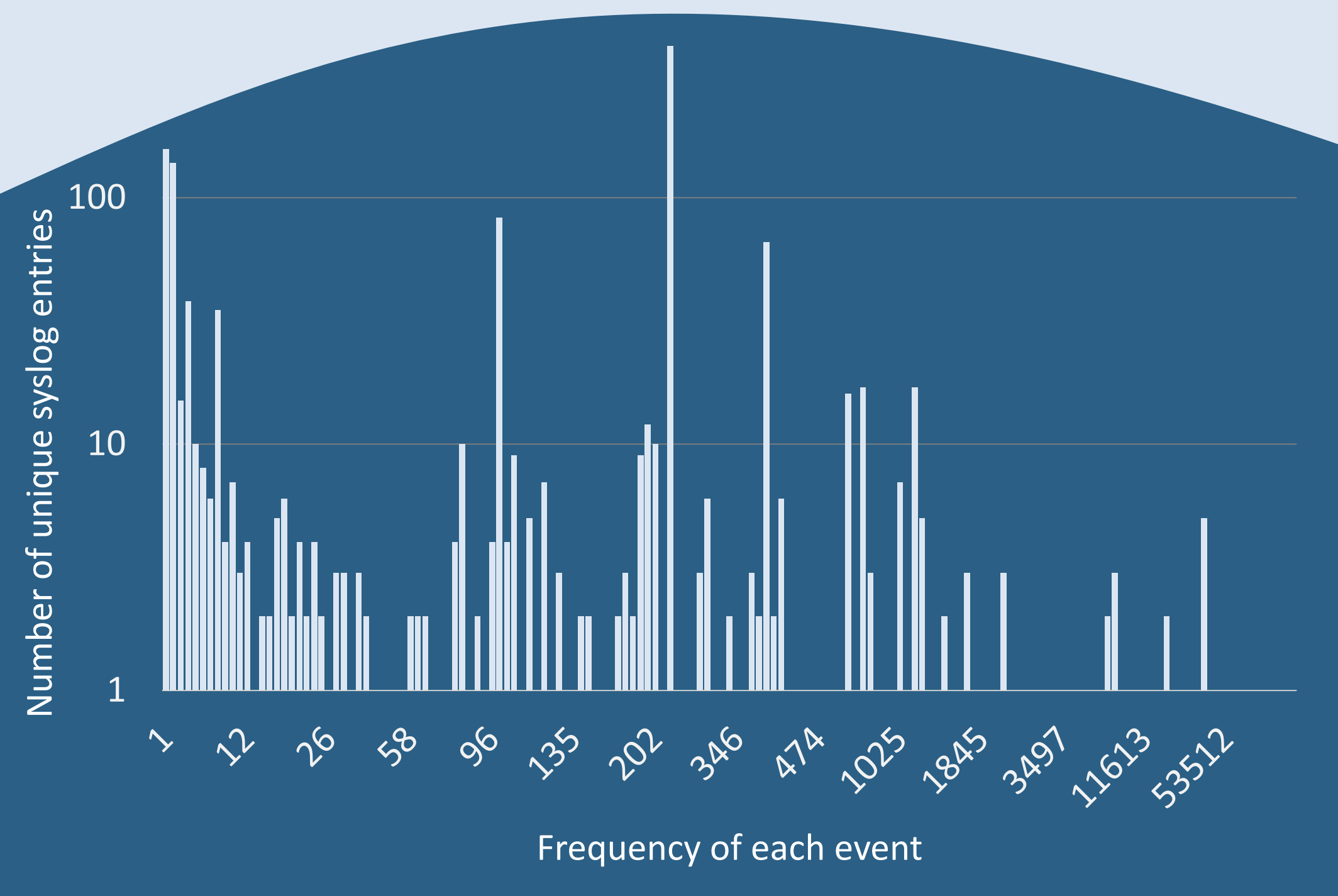
```
➢ ~~#001 x2~~
➢ #002 x1
➢ #003 x2
➢ #004 x1
➢ #003 x2
➢ ~~#001 x2~~
```

4 Removal of non-informative entries

```
✓ #002 > #003 > #004 > #003
✓ #003 > #004 > #003
✓ #004 > #003
```

5 Detection and extraction of chains of event

3. Filtering the Data



Event[1] frequency	Event patterns	Total events	Percentage
1 - 5	358	630	0.04%
6 - 100	233	12,885	0.85%
101 - 200	52	7,493	0.49%
201 - 300	442	91,000	5.98%
301 - 400	12	3,902	0.26%
401 - 500	86	35,694	2.34%
501 - 1000	40	29,414	1.93%
1001 - 4000	55	83,848	5.51%
4002 - 10000	13	73,207	4.81%
10001 - 100000	22	803,452	52.77%
100001 – 150000	1	381,172	25.03%
ALL	1312	1,522,697	

System log form	Data size in Bytes
Raw system log	99,079,741
De-identified	98,006,233
De-identified + Hashed (anonymized)	50,250,651
Double hashing	4,386,137
Smart hashing	150,000 – 400,000

Based on system logs, collected during 10 days on 99 nodes of Taurus[2] HPC system

Observation

More than **77%** of system logs are related to **23** different **events.**

10% of system logs are responsible for more than **90%** of syslog network traffic.

25% of system logs are related to a single event and can safely be ignored.

The **most alarming** events are among the less than **1%** of all system logs.

Events related to several errors, including "file system" failures, are located among the **74%** of all system logs.

4. Pattern Detection

Results

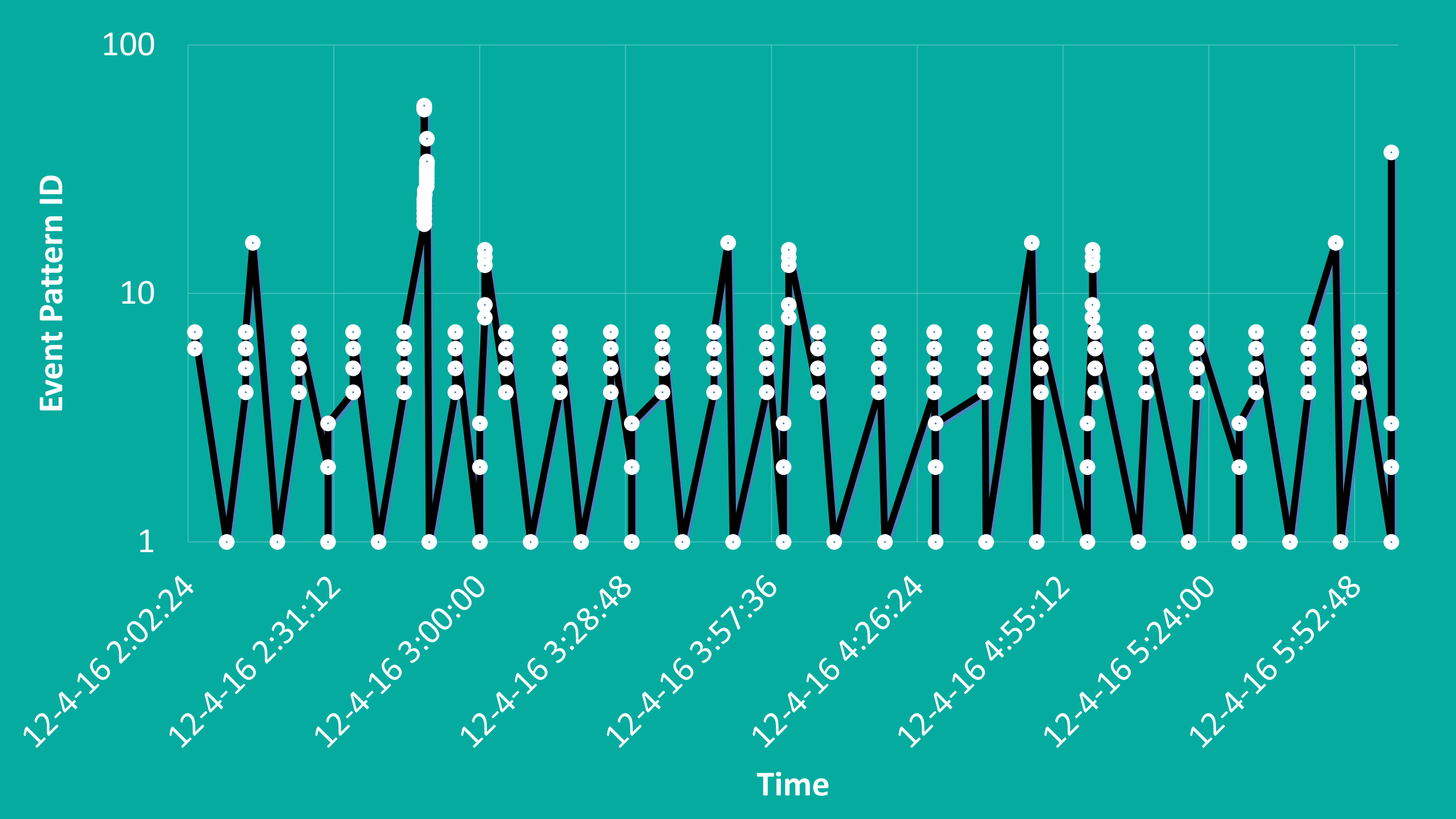
Filtered data requires **~95%** less storage space.

Data filtering, significantly speeds-up the identification process.

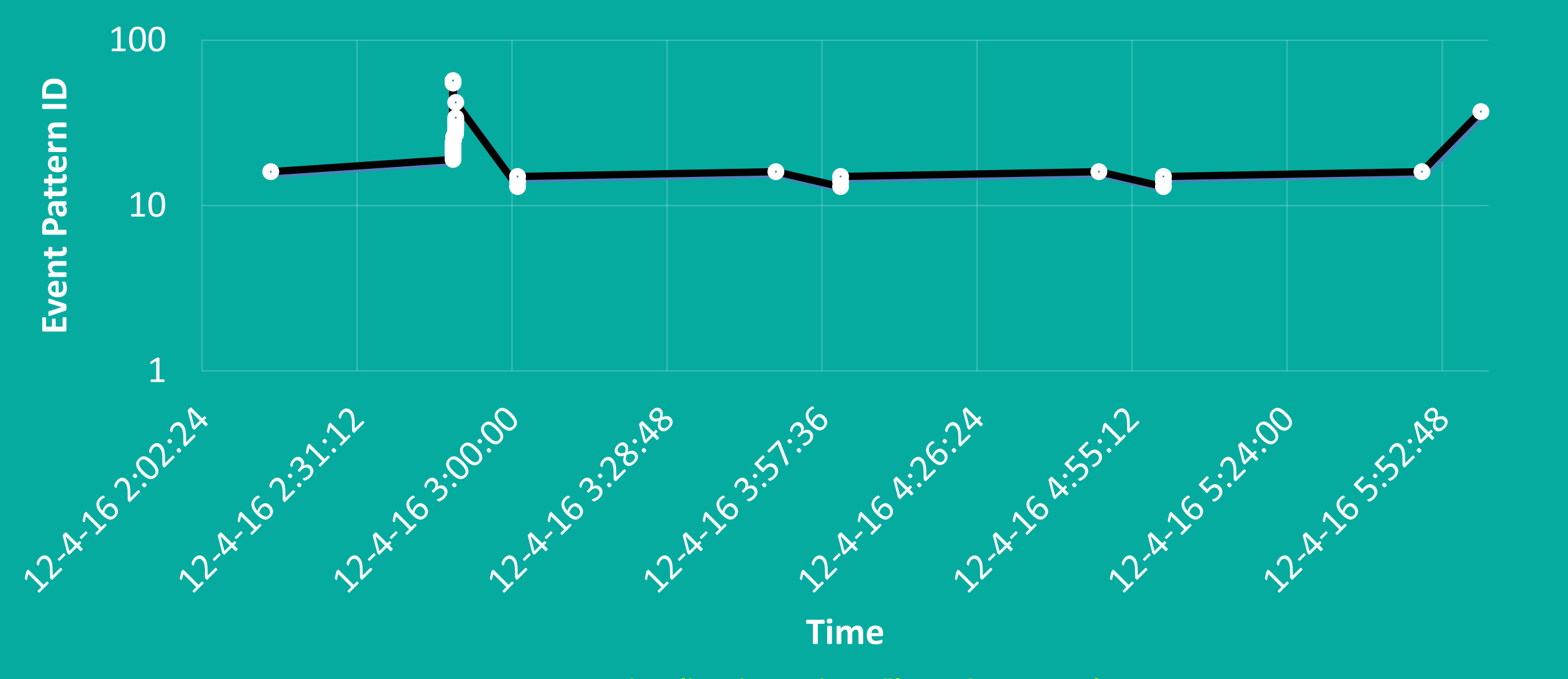
Removing the **25%** of most frequent events, resulted in **~50%** speed-up!

Data is ready to be used by different mining approached, including but not limited to:

- A priori-based
  e.g., GSP[3] and SPADE[4]
- Pattern-growth-based
  e.g., FreeSpan and PrefixSpan



Events timeline based on raw system logs



Events timeline based on filtered system logs

5. Conclusion

1. A small portion of system logs contains the most alarming information.
2. Filtering system logs based on document frequency, can significantly reduce the required storage capacity.
3. The data can be fully anonymized but still useful for some statistical analysis.
4. System log filtering significantly increases the performance of pattern detection algorithms.
5. Reducing the volume of non-informative data and combining other sources of information increase the accuracy of analysis results, without requiring additional computational power.

Footnotes

[1] Each event is a log, in which all variables are replaced by constant sample values.
[2] https://doc.zih.tu-dresden.de/hpc-wiki/bin/view/Compendium/SystemTaurus
[3] Generalized Sequential Pattern mining algorithm.
[4] Sequential PAttern Discovery using Equivalence classes.

Acknowledgment
This work is in part supported by the German Research Foundation (DFG) in the Cluster of Excellence "Center for Advancing Electronics Dresden" (cfaed) .

References
[**1**] W. E. Nagel et al., "Planning for exascale systems: The challenge to be prepared", Dagstuhl Reports, vol. 3, no. 9, pp. 122., 2014
[**2**] R. Vaarandi and M. Pihelgas, "LogCluster - A data clustering and pattern mining algorithm for event logs," 2015 11th International Conference on Network and Service Management (CNSM), Barcelona, 2015, pp. 1-7.
[**3**] S. Ghiasvand, F. M. Ciorba, and W. E. Nagel, "Turning Privacy Constraints into Syslog Analysis Advantage", The International Conference for High Performance Computing, Networking, Storage and Analysis (poster), Saltlake, Utah, USA, Novemeber 2016

TECHNISCHE UNIVERSITÄT DRESDEN

ZIH
Center for Information Services & High Performance Computing

cfaed

Universität Basel