Detecting New Patterns of Attacks – Results and Applications of Large Scale Sensoring Networks

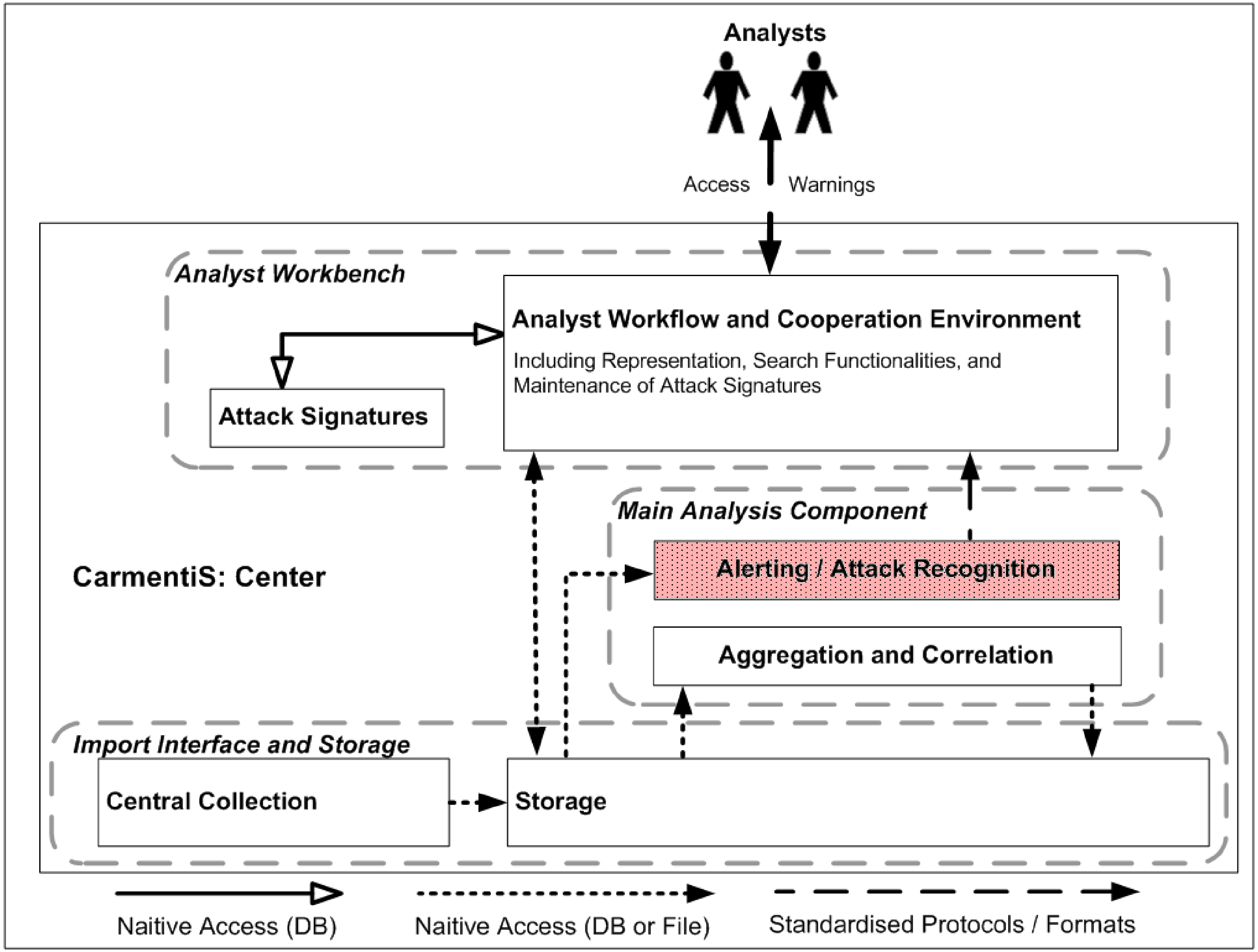
IMF 2006, October 18 – 19, 2006
Torsten Voss & Klaus-Peter Kossakowski

Contents

Background and Motivation

- Projects within the context of CERTs to improve early warning a.k.a. CarmentiS
  - Detection of new attacks from viruses and worms
  - Trend analysis in regard to attack pattern and sources
  - Correlation of diverse sensor data
- Support for the human analyst
  - to deal with large data sets
  - to allow easy classification and priorization

Detailes of the Structure of CarmentiS-Center



Algorithm

- Data-Mining Algorithmus

  Apriori of R. Agrawal, et al. 1993

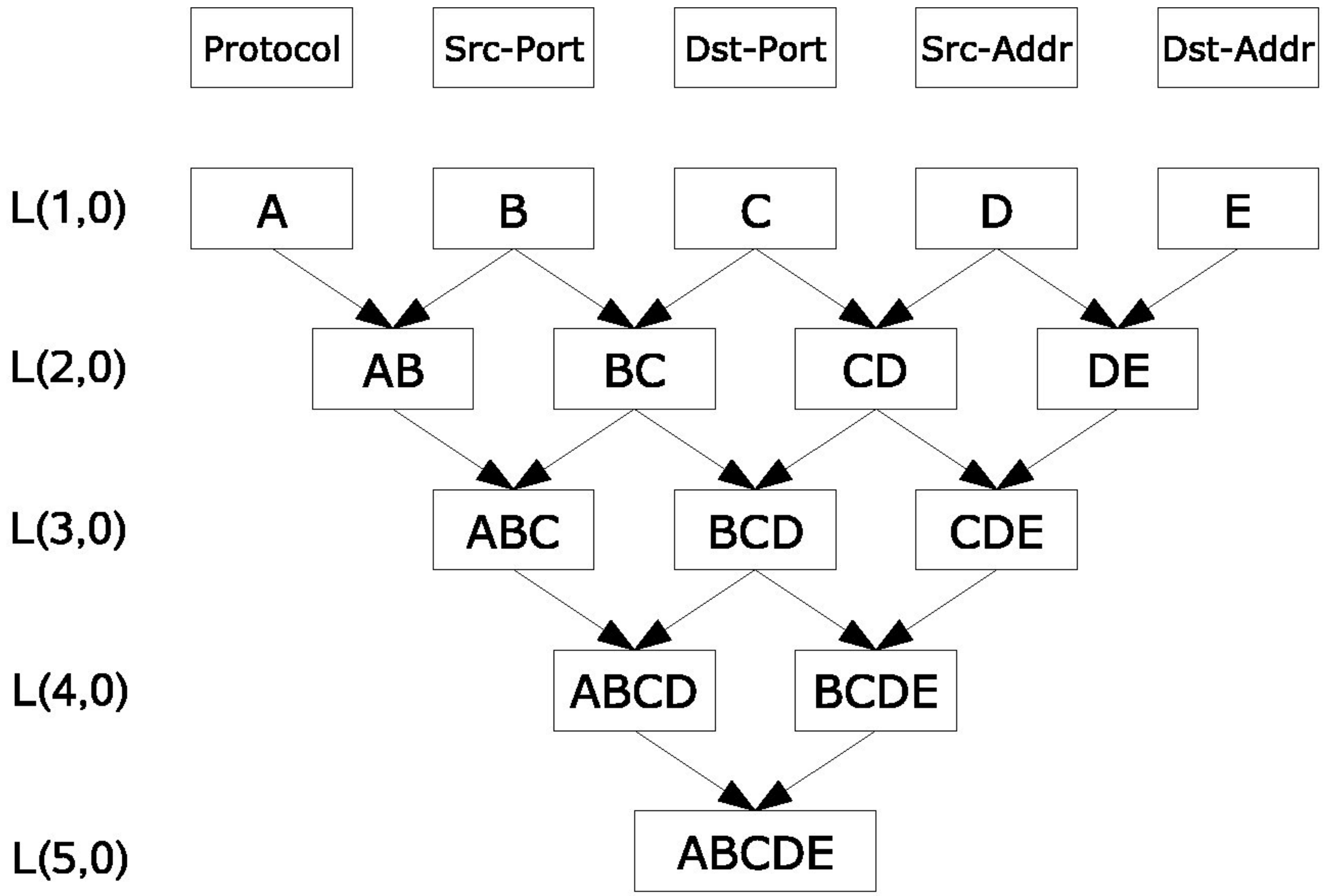
  Data-Mining Framework of L. Wenke, Phd-Thesis 1999

  Discontinuous Pattern of Y.-L. Chen, et al. 2002

  A. Alharby, 2005, combined approches of Wenke and Chen

  - Finding Frequent Items in a Dataset
  - Counting Item Freuquencies
- Data-Mining at Database
  - we use Postgres

Continuous Pattern Tree



(Figure from Alharby, 2005)

Example

- Dataset:

Protocol	Src-Port	Dst-Port	Src-Addr	Dst-Addr
TCP	1025	80	192.168.0.1	10.0.0.1
ICMP	0	0	192.168.0.4	10.0.0.2
TCP	1029	22	192.168.0.4	10.0.0.1
UDP	1027	21	192.168.0.2	10.0.0.3
TCP	1026	80	192.168.0.2	10.0.0.2
TCP	1027	80	192.168.0.3	10.0.0.1
ICMP	0	0	192.168.0.2	10.0.0.2
TCP	1027	22	192.168.0.4	10.0.0.1
TCP	1028	22	192.168.0.4	10.0.0.1
ICMP	0	0	192.168.0.5	10.0.0.3

Example 1: Protocol

- Dataset:

Protocol
TCP
ICMP
TCP
UDP
TCP
TCP
ICMP
TCP
TCP
TCP

- Select all different elements from one element-type:

Protocol
TCP
UDP
ICMP

Example 1: Protocol

- Dataset:

Protocol
TCP
ICMP
TCP
UDP
TCP
TCP
ICMP
TCP
TCP
TCP

- Select all different elements from one element-type
- Counting the frequencies:

Protocol	Counter
TCP	7
UDP	1
ICMP	3

Example 1: Protocol

- Dataset:

Protocol
TCP
ICMP
TCP
UDP
TCP
TCP
ICMP
TCP
TCP
TCP

- Select all different elements from one element-type
- Counting the frequencies
- Threshold of 3:

Protocol	Counter
TCP	7
UDP	1
ICMP	3

Example 2: Src- & Dst-Port

- Pattern:

Src-Port
0
1027

Dst-Port
0
22
80

- Combine the pattern:

Src-Port	Dst-Port
0	0
0	22
0	80
1027	0
1027	22
1027	80

Example 2: Src- & Dst-Port

- Dataset:

Src-Port	Dst-Port
...	...
0	0
...	...
1027	21
...	...
1027	80
0	0
1027	22
...	...
0	0

- Combine the pattern
- Counting the frequencies:

Src-Port	Dst-Port	Counter
0	0	3
0	22	0
0	80	0
1027	0	0
1027	22	1
1027	80	1

Example 2: Src- & Dst-Port

- Dataset:

Src-Port	Dst-Port
...	...
0	0
...	...
1027	21
...	...
1027	80
0	0
1027	22
...	...
0	0

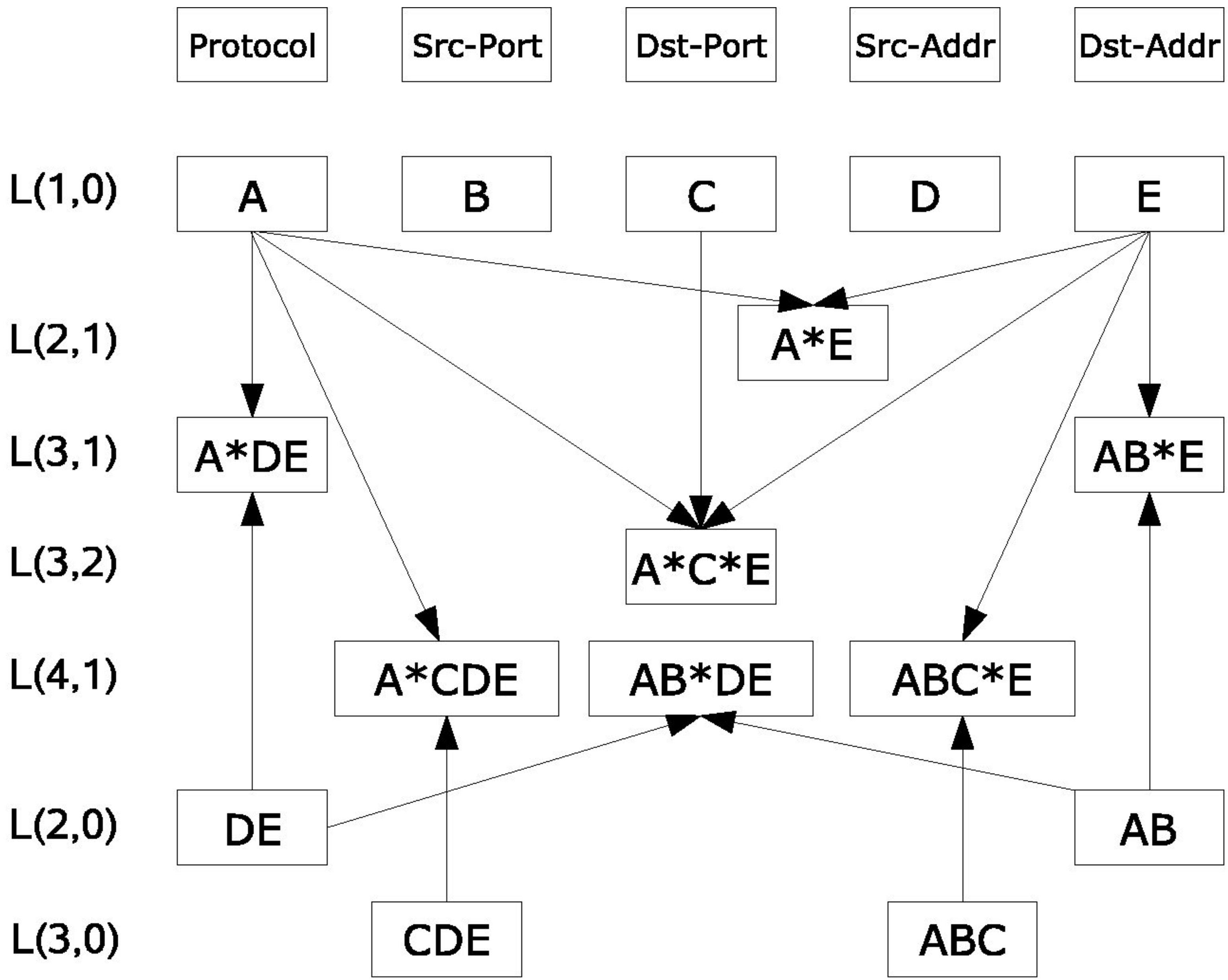
- Combine the pattern
- Counting the frequencies
- Threshold of 3:

Src-Port	Dst-Port	Counter
0	0	3
0	22	0
0	80	0
1027	0	0
1027	22	1
1027	80	1

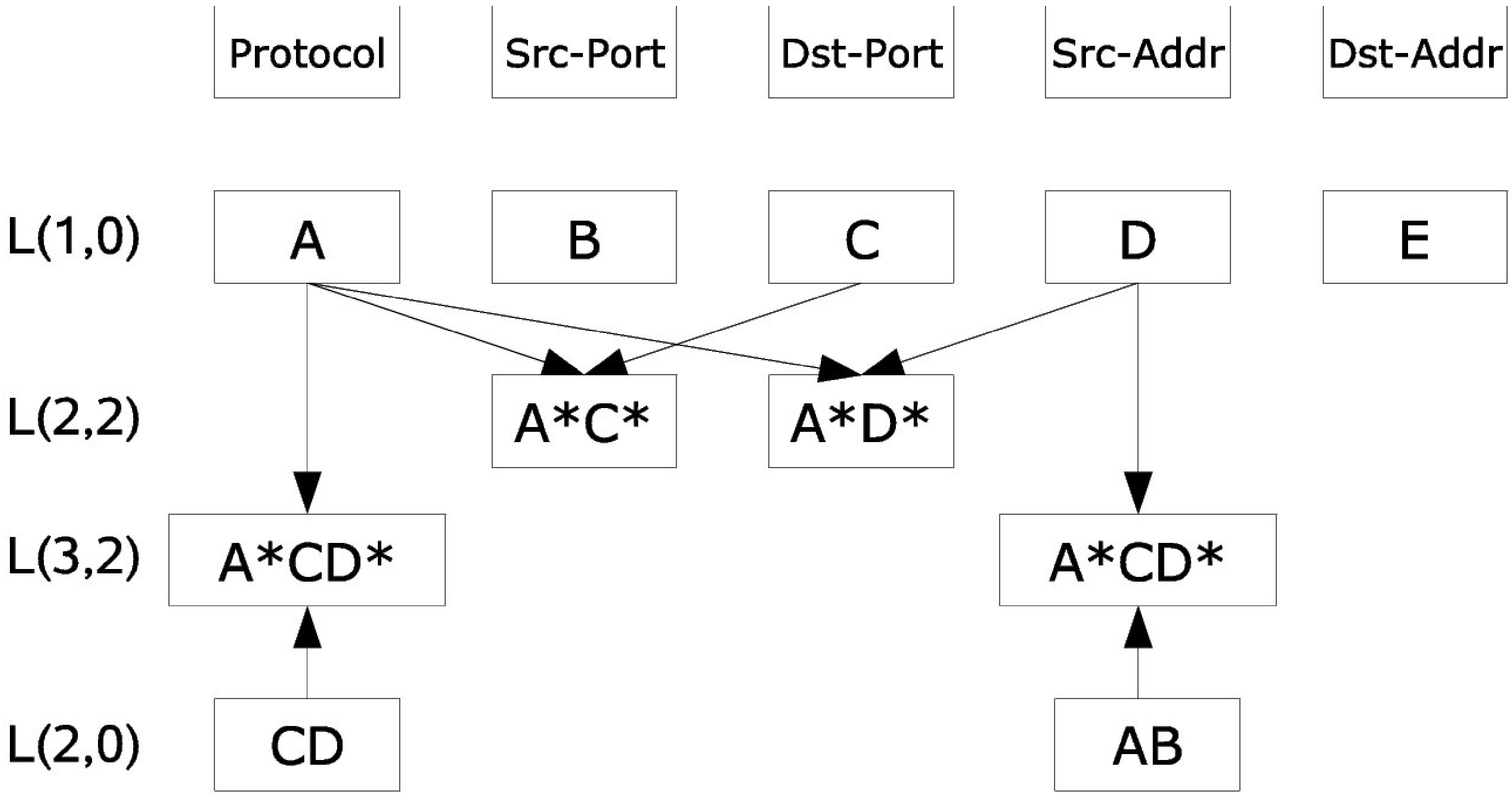
Discontinuous Pattern

- A 'gap' (defined as star) in the Continuous Pattern
- From history the definition of discontinuous pattern:
  - Start and end with a value (string)
  - That means:
    - Start with Protocol
    - End with Dst-Addr
- Our improvement:
  - Start with Protocol
  - May end with a star

Discontinuous Pattern Tree



New Discontinuous Pattern



Practical Problem

- Example:
  - To combine 15.000 source-ports and 16.000 destination-ports it results over 240 million combinations

- Algorithm
  - too much combinations
  - combining patterns costs a lot of resources
  - counting the frequencies spent
  - Approximate 90 percent of combinations are not in the database

1. Improvement

- none combining
- search for combinations that are really in the dataset
- This can be efficiently done with a Single SQL-Statement:

```
SELECT DISTINCT b.srcport, b.dstport
    FROM l10_b a
    INNER JOIN flows b
    ON (a.srcport = b.srcport);
```

1. Improvement

DFN CERT

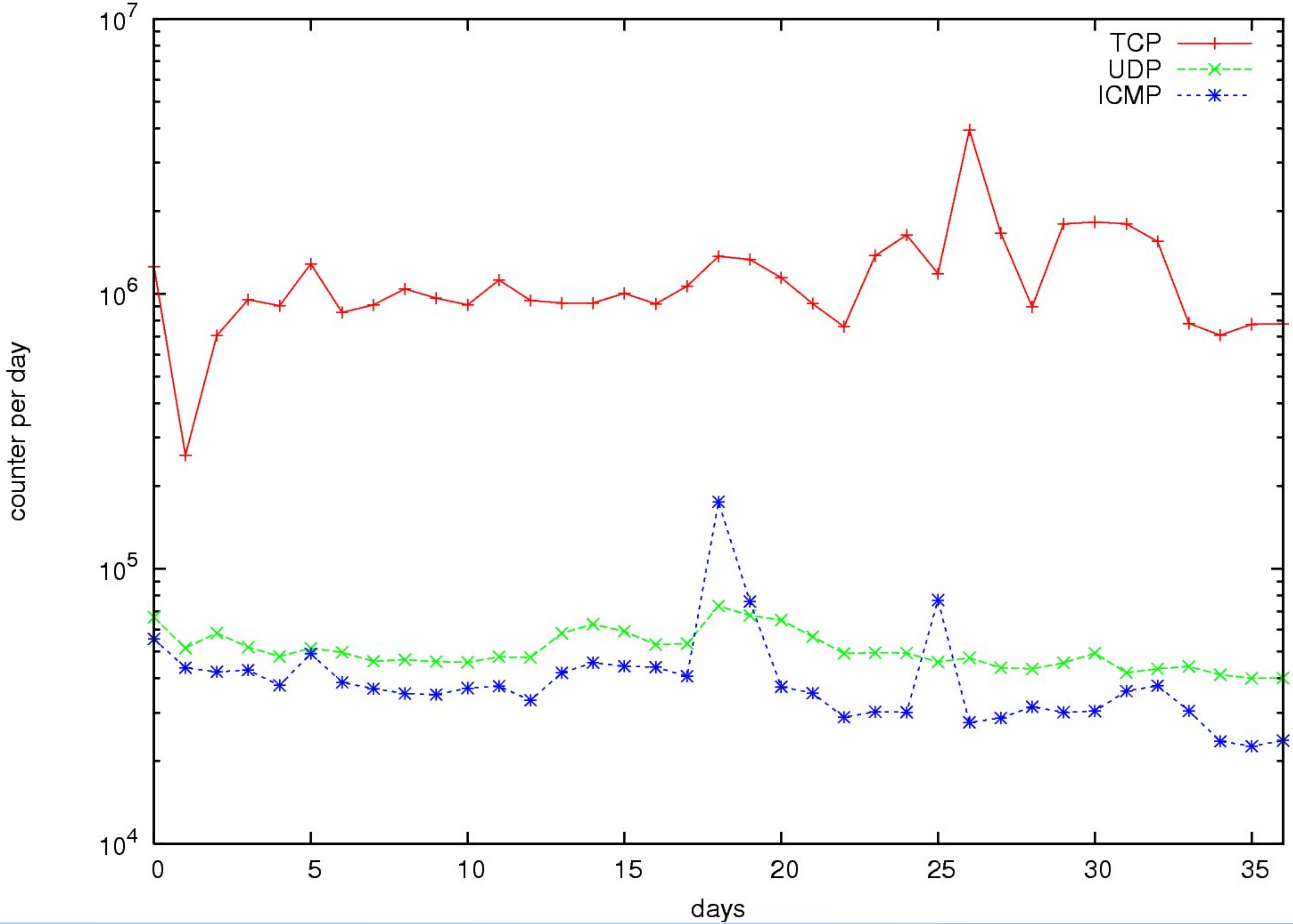
- In our example only 270.000 combinations have been identified applying our approuch.
- Compared to the original number of 240 million combinations the improvement is significant
- Speeds up the processing in this example from around 4 hours to 10 minutes

Additional Improvements (1)

- Store the counter of every pattern
  - useful to estimate the patterns of the analyst
- Save the difference of counter of actual and last timestamp
  - for every pattern
  - saved in a seperate database-schema
  - to see tendencies (used in the following figures)
  - Costs more process-time
    - approx. 20 minutes for over 2.2 million pattern

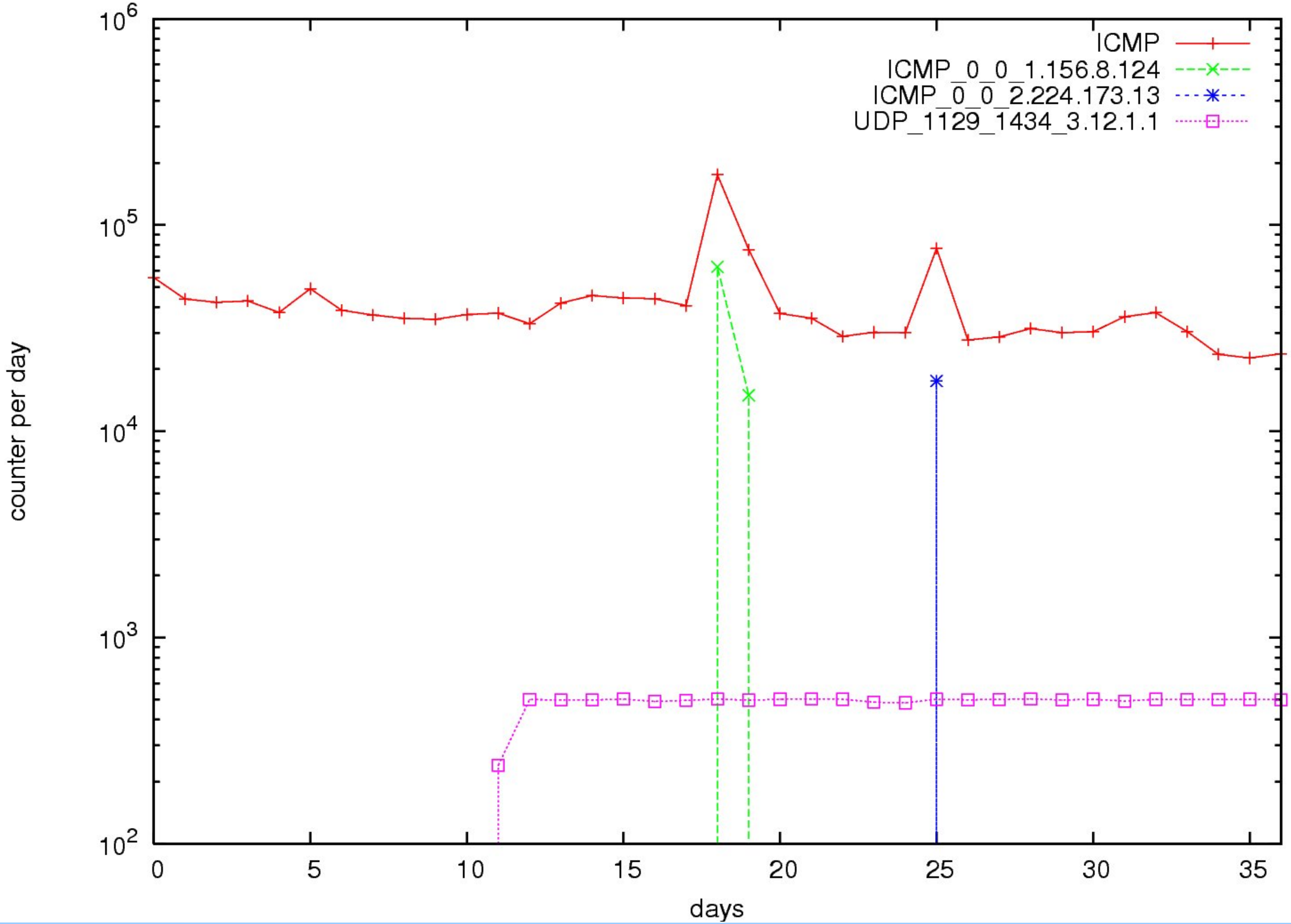
Result (1)

Tendency of pattern: Protocoll (L10_a)



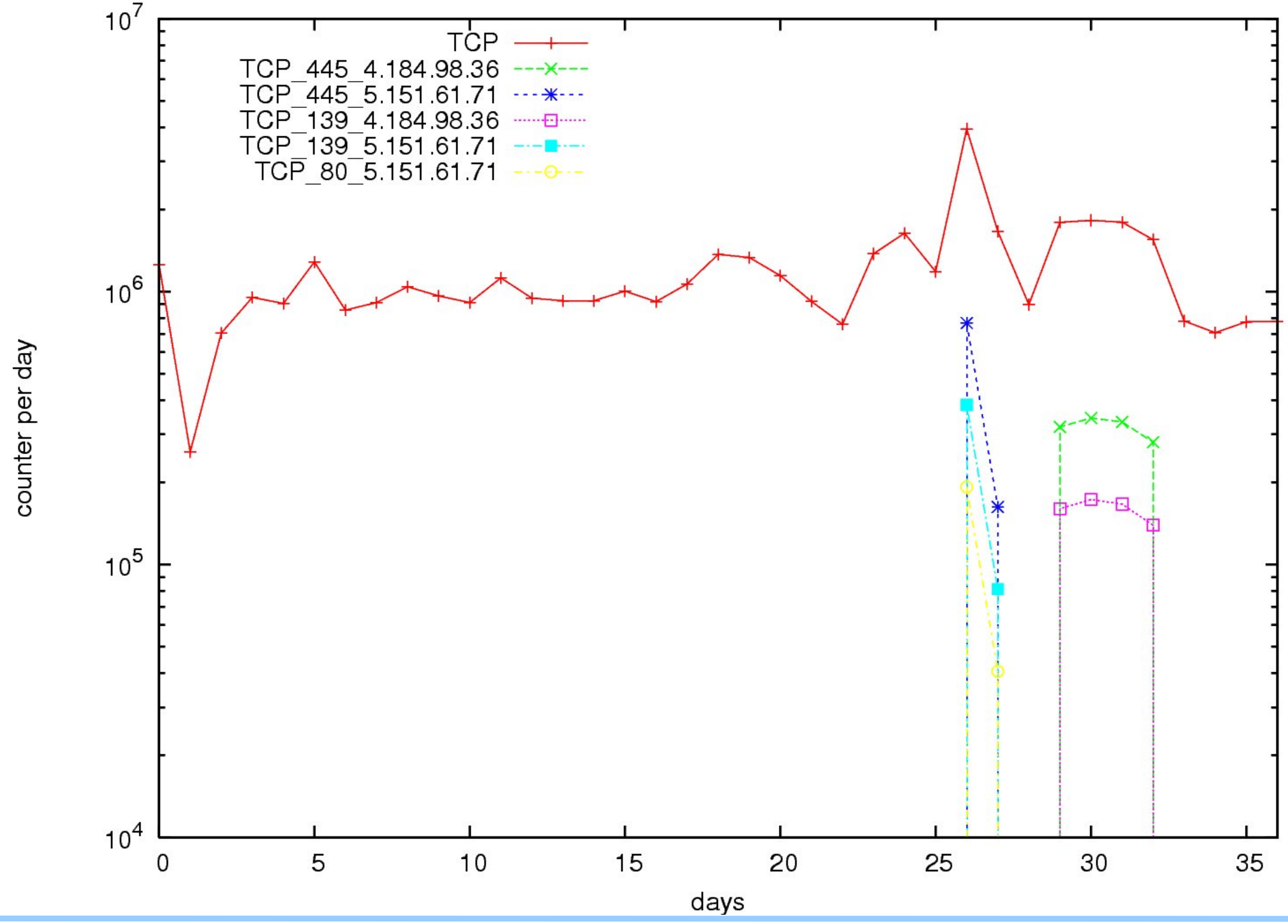
Result (2)

Tendency of pattern: Prot., Src.- and Dst-Port, Src-IP (L40_abcd)



Result (3)

Tendency of pattern: Protocoll, Dst.-Port, Src.-IP (L32_a_cd)



Result of processing

- Results from the algorithm:
  - nealy 47 million netflows
  - sliced in 36 days and processed per day
  - over 2.2 million pattern are created (with a threshold of 15)
  - processed in round about 8 hours (*)

(*) with Dual Xeon 3.2 Ghz Processor

Additional Improvements (2)

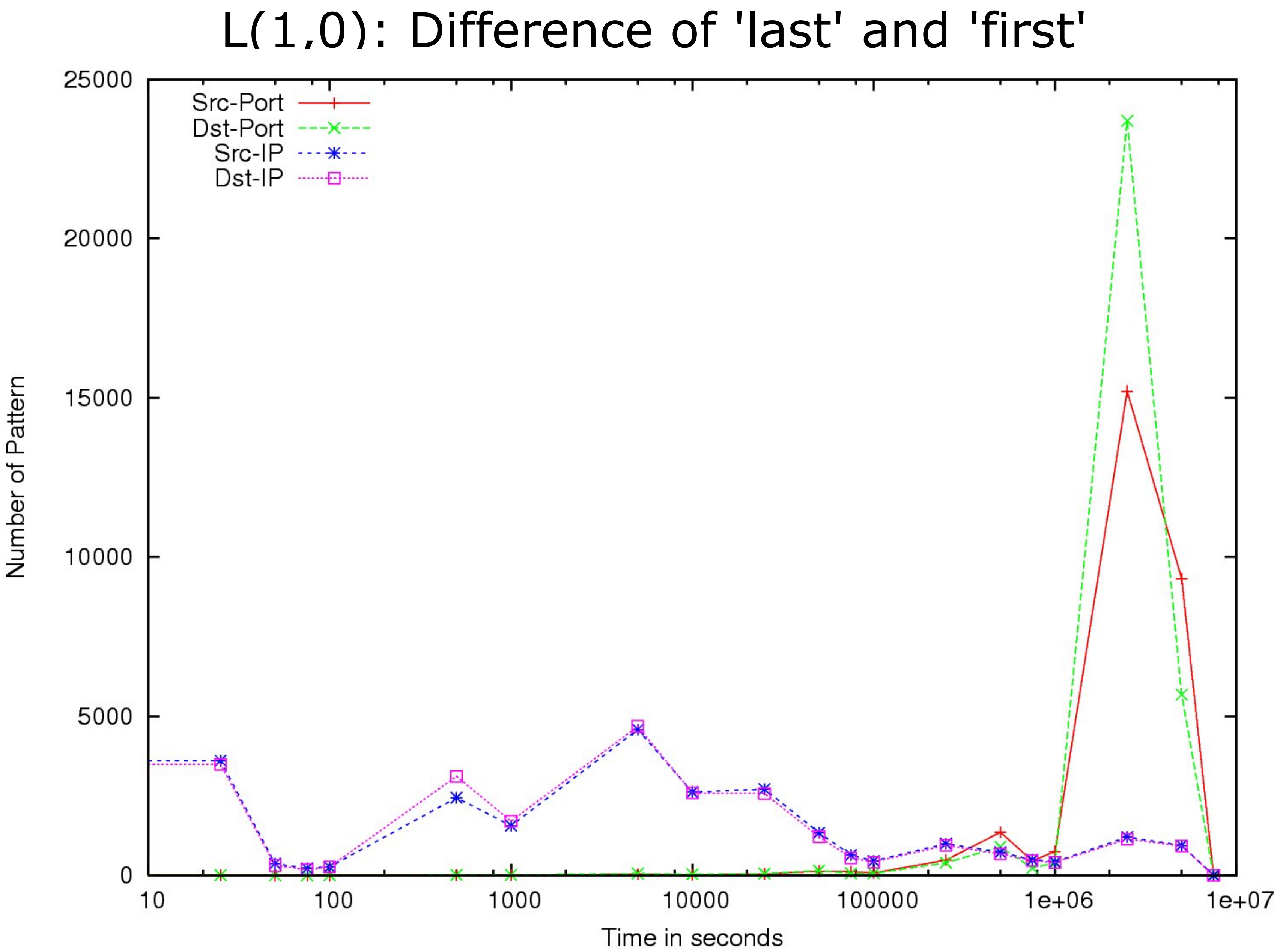
- Is every pattern actual and used?
- Which pattern are obsolete and not any more used?

- Save two timestamps of every pattern
  - 'first' – seen: timestamp of the datarecord that creates the pattern
  - 'last' – seen: timestamp of the datarecord that matched this pattern at last
  - used to analyse the lifetime of a pattern

Lifetime of Pattern

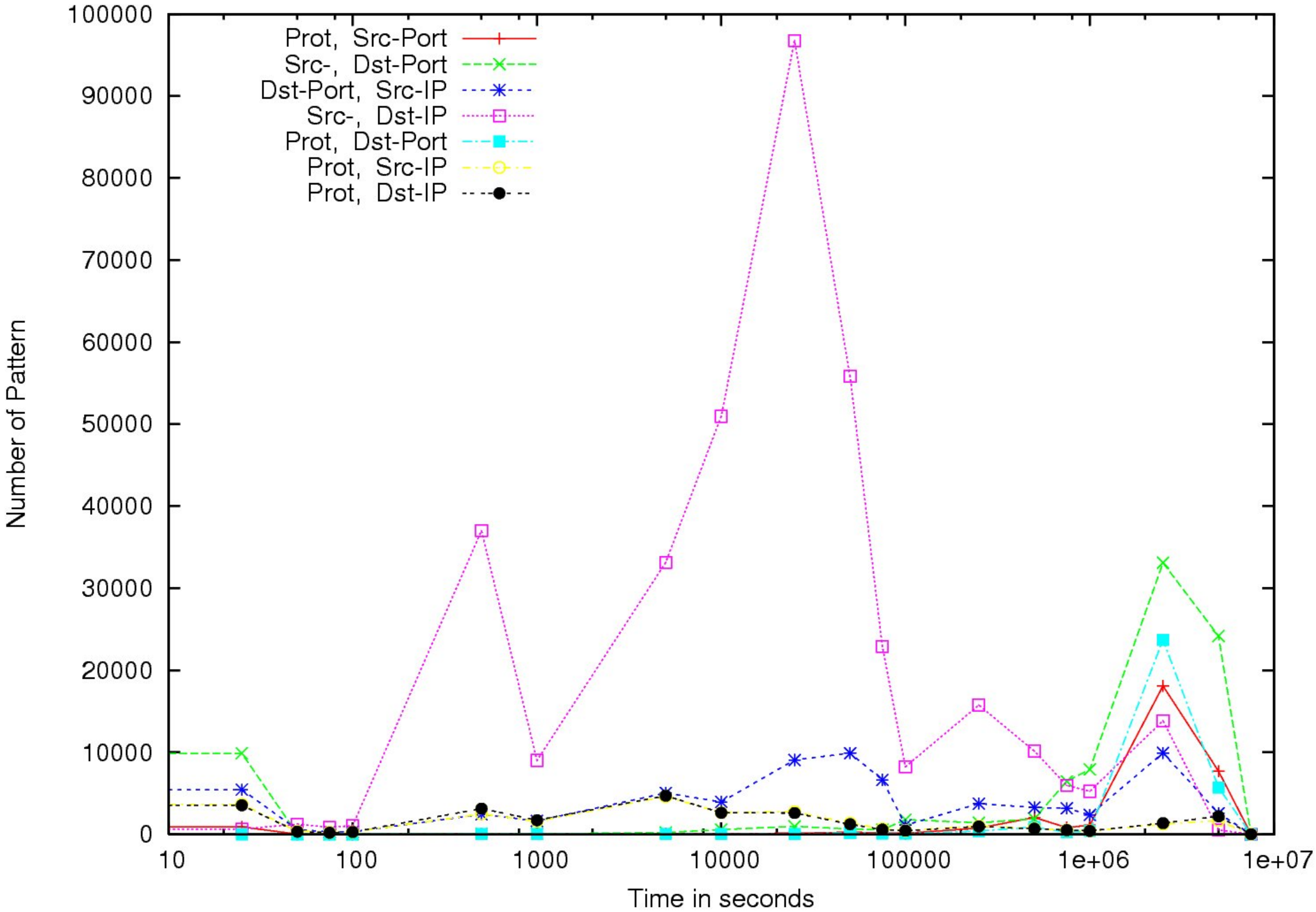
- Select and count the pattern by difference of 'last'- and 'first'-seen (differ per pattern type)
  - that means we count pattern differ of lifetime
  - we collect figure 1 and 2

Lifetime of Pattern (1)



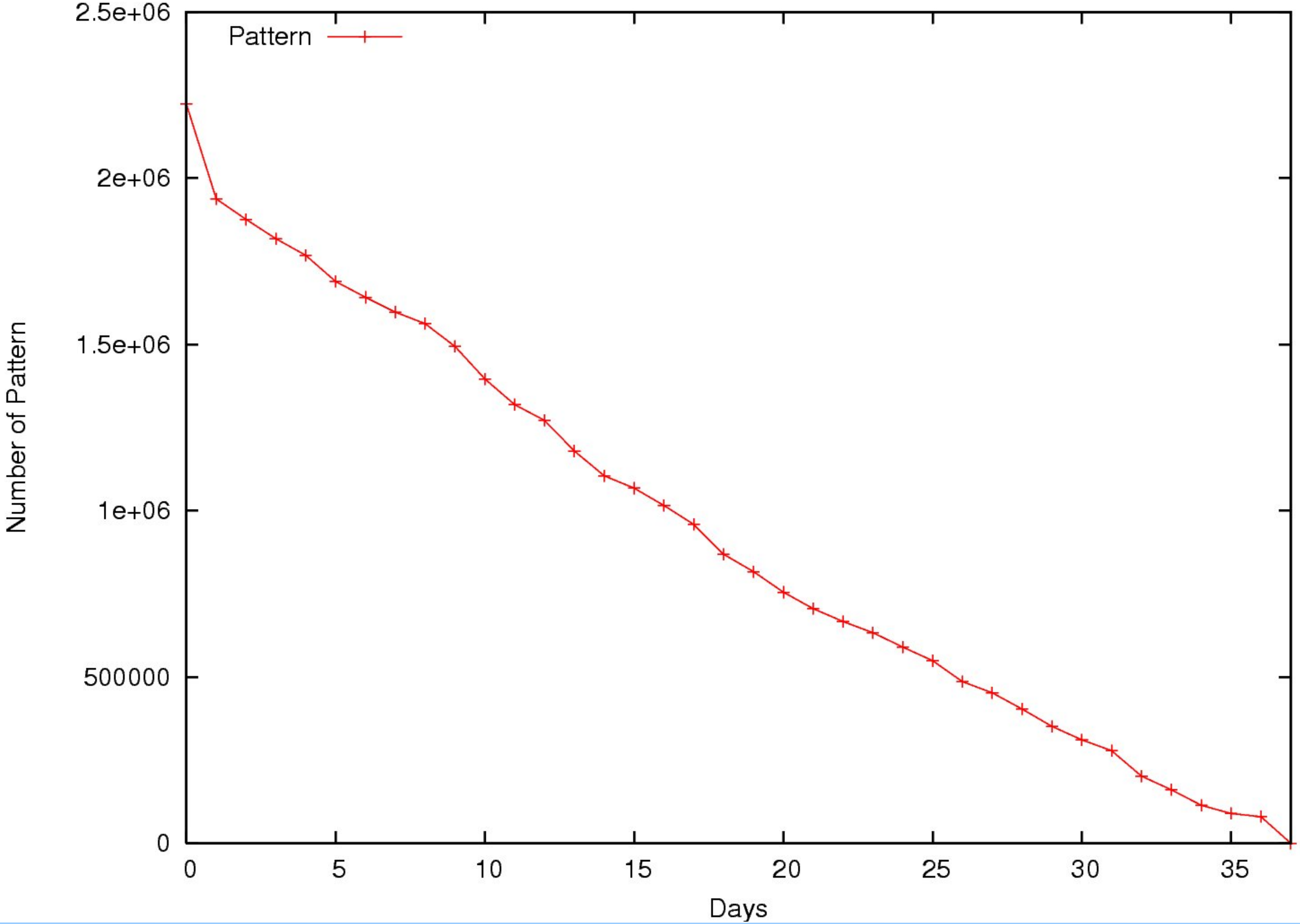
Lifetime of Pattern (2)

L(2,x): Difference of 'last' and 'first'



Lifetime of Pattern (5)

Number of P. where 'last' < timestamp of day (1=yesterday,...)

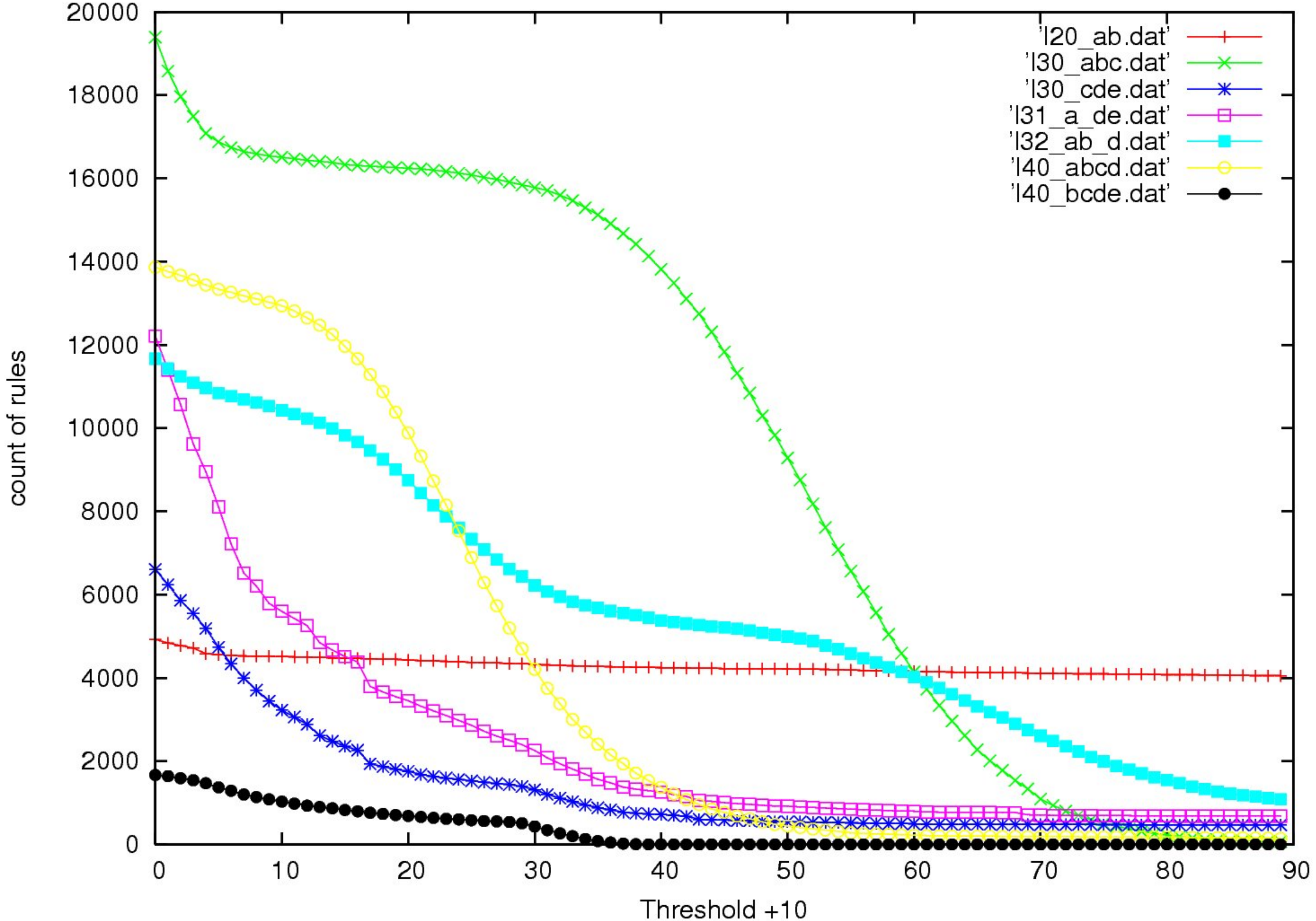


Result of Lifetime

- fast aging of great many pattern
- after 10 days over 1.4 million pattern are obsolete
- can be used to save space because older pattern can be deleted
- Post-Processing has to be done very fast

Threshold

Behavior of pattern-types by changing the threshold



Result of Threshold

- The behavior of pattern-types are different
- if only one threshold for all
  - see not every attack or
  - have too much uninteresting pattern
- diff. value of threshold for pattern-types
- the analyst need to be able to set these thresholds

Conclusion

- no combination-building
- significant faster than the orig. algorithm
- Trend analysis in regard to attack pattern
- obsolte pattern can be filtered with the lifetime

Thank you for your attentions!

Questions?