How to tackle Interconnect Bypass Fraud ?

Bruno Veloso[1],

[1]LIAAD-INESCTEC; U. Portucalense and FEP - UP

bruno.m.veloso@inesctec.pt

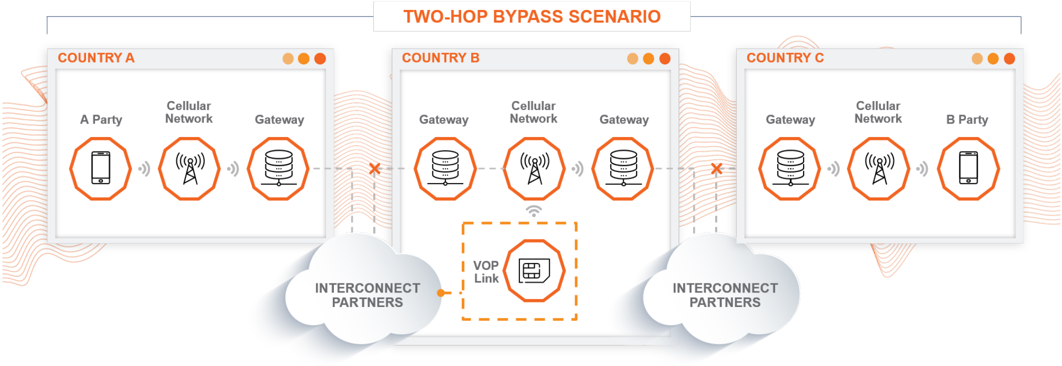
Contents

Problem definition

- In the telecommunication world, fraud is defined as the abusive usage of network and services without the intention of paying.
- According to the Communications Fraud Control Association (CFCA) 2021 Global Fraud Loss Survey, around 39 Billion USD is lost annually in the Telecom sector as a result of Telecom Fraud.
- In the survey, interconnect bypass fraud is one of the largest sources of lost revenues and costs network operators.

Problem definition

In Interconnect Bypass Fraud, one of several intermediaries responsible for delivering phone calls forwards the traffic over a low cost IP connection.



Problem definition

- This type of fraud is detected by analysing the call patterns of the gateways.
- But, the behaviour of gateways evolve over time, resembling some of them, true SIM Farms, capable of manipulating identifiers, simulating standard call patterns similar to the ones of normal users

Proposed Approach

How to detect interconnect bypass fraud on telecommunications?

Current approaches are based on `blacklists`:

- Inefficient in detecting new frauds
- Inefficient in detecting changes in the patterns

Proposed Approach

How to detect interconnect bypass fraud on telecommunications?
Our approach is data driven and works online. We are looking for:

- High asymmetry of international termination rates.
- High activity with abnormal behaviours.
  - Bursts of calls – a huge amount of calls;
  - Calling large set of numbers
  - Repetition – same pattern of calls during a period of time;
  - Mirror – the huge amount of calls are divided by multiple numbers.

Video

Used Techniques

Detect in real time and as soon as possible:
One pass streaming algorithms!

- Frequent Items
  - Heavy Hitters – provide approximate counts of the frequent items [1].
  - Hierarchical Heavy Hitters – provide a rank of the most frequent items in a specific hierarchy [2].
- We signal alarms, when calling numbers, exhibit activity profile:
  - Large number of phone calls - HH
  - Bursts in activity - HH
  - Calling too many numbers - HHH

[1] G. S. Manku and R. Motwani, "Approximate frequency counts over data streams," inVLDB'02

[2] G. Cormode,S. Muthukrishnan, and D. Srivastava, "Finding hierarchical heavy hitters in streaming data", TKDD

Experimental Work

- Two data sets: different periods
- Each record (one phone-call) contains information about:
  - Origin numbers (A-Numbers).
  - Destination number (B-Numbers).
  - Timestamp.
  - Blacklist Code: if the A-number is in the blacklist or not.

Data set 1

- Collected during three months between 24/07/2018 to 21/10/2018
- 89 days which includes 83.366.367 examples.
- Unique ANumber: 9.006.011
- Unique BNumber: 2.387.932

Data set 2

- Collected during one month between 01/06/2019 to 30/06/2019
- 29 days which includes 32.879.670 examples.
- Unique ANumbers: 3.217.069
- Unique BNumbers: 1.380.235

Experimental Work – HH

- The sequence of A numbers are used as a stream:
  Frauds are originated from A numbers,
- Use the lossy counting algorithm to provide approximate counts of the frequent items

Lossy Counting Algorithm

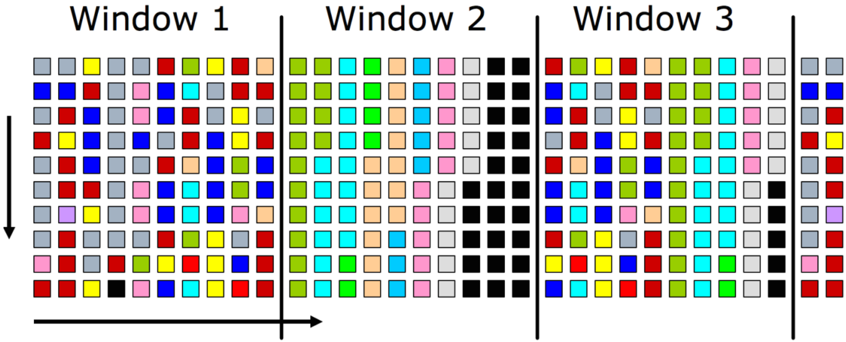


Figure: Stream of calls [fig source:micvog.com]

Lossy Counting Algorithm

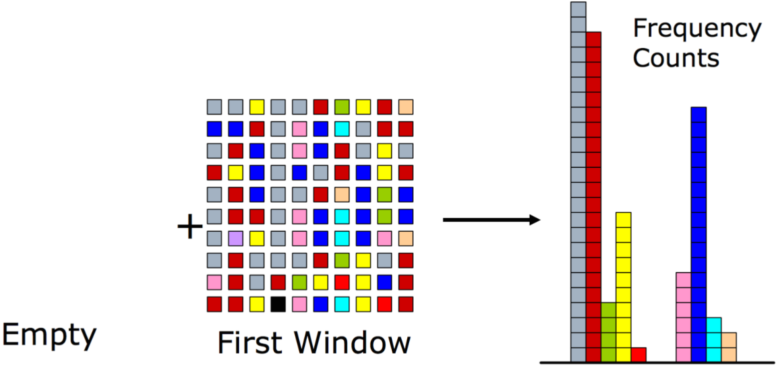


Figure: Stream of calls [fig source:micvog.com]

Lossy Counting Algorithm

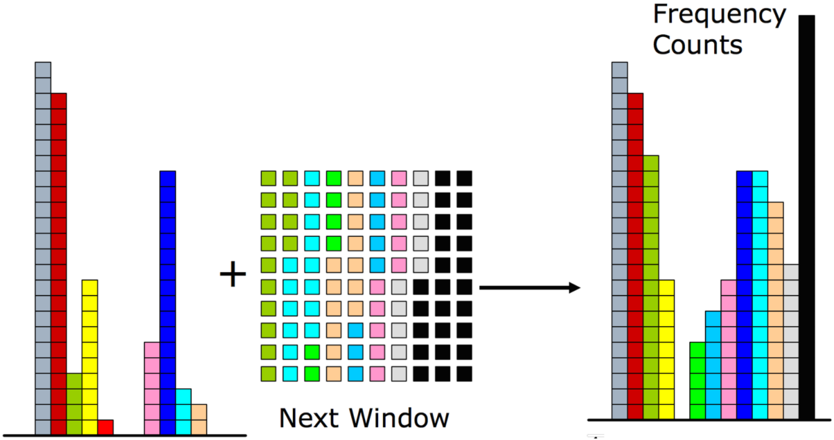


Figure: Stream of calls [fig source:micvog.com]

Experimental Work – Contribution

Lossy Count

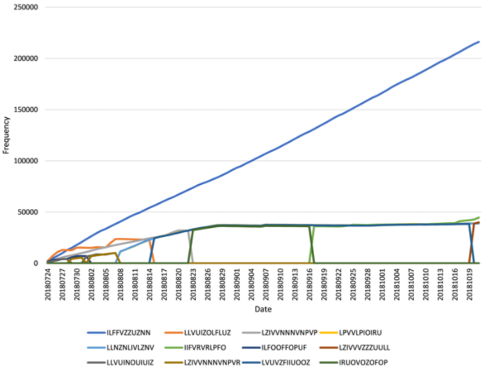
input: S: A Sequence of Examples; ϵ: Error margin;
begin
    $n \leftarrow 0$; $\Delta \leftarrow 0$; $T \leftarrow 0$;
    foreach *example* $e \in S$ **do**
        $n \leftarrow n + 1$
        if *e is monitored* **then**
            Increment $Count_e$
        else
            $T \leftarrow T \cup \{e, 1 + \Delta\}$
        end
        if $\left\lceil \frac{n}{\epsilon} \right\rceil \neq \Delta$ **then**
            $\Delta \leftarrow \frac{n}{\epsilon}$
            foreach *all* $j \in T$ **do**
                if $Count_j < \Delta$ **then**
                    $T \leftarrow T \backslash \{j\}$
                end
            end
        end
    end
end

Lossy Count with Forgetting

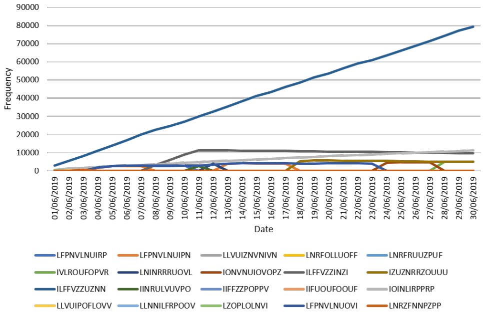
input: S: A Sequence of Examples; ϵ: Error margin; α: fast forgetting parameter
begin
    $n \leftarrow 0$; $\Delta \leftarrow 0$; $T \leftarrow 0$;
    foreach *example* $e \in S$ **do**
        $n \leftarrow n + 1$
        if *e is monitored* **then**
            Increment $Count_e$
        else
            $T \leftarrow T \cup \{e, 1 + \Delta\}$
        end
        if $\left\lceil \frac{n}{\epsilon} \right\rceil \neq \Delta$ **then**
            $\Delta \leftarrow \frac{n}{\epsilon}$
            foreach *all* $j \in T$ **do**
                $Count_j \leftarrow (1 - \alpha) * Count_j$
                if $Count_j < \Delta$ **then**
                    $T \leftarrow T \backslash \{j\}$
                end
            end
        end
    end
end

Experimental Work: Lossy Counting: Top-k A numbers

Data set 1

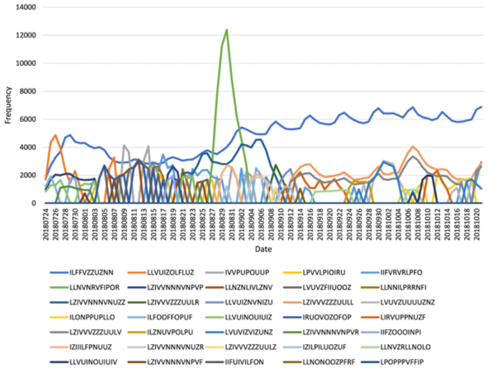


Data set 2

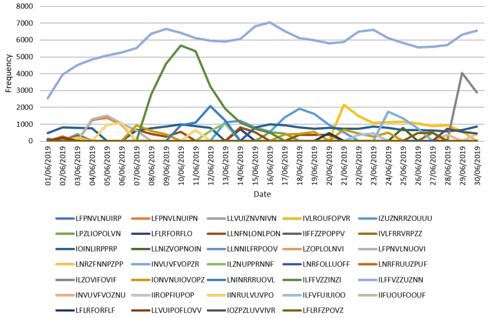


Experimental Work: Lossy Counting w/ Forgetting Top-k A numbers

Data set 1



Data set 2



Sensitivity Analysis

- Forgetting Parameter Sensitivity; α is the forgetting parameter and UAN is Unique A-Numbers

α	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.99
UAN	211	210	203	192	180	175	158	123	93	66	12

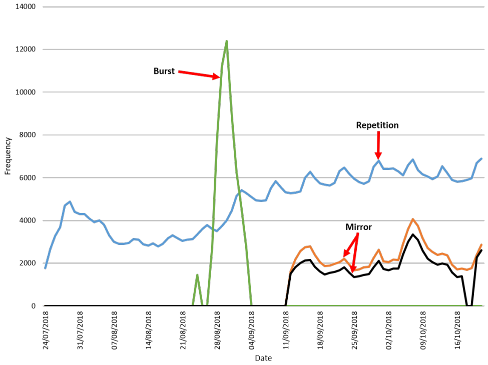
Performance Analysis

- Performance comparison of the Lossy Counting (LC) vs Lossy Counting with Fast Forgetting (LCFF)

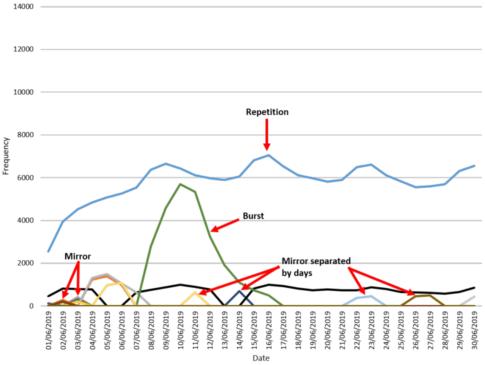
Algorithm	Runtime (s)	Memory (MB)	SpeedUp (Examples/s)
LC	88	75.8	947 345
LCFF	72	28.8	1 157 866

Experts Annotation

Data set 1



Data set 2



Discussion

- **Contributions:**
  - **Application level:** Real-time identification of suspicious behaviours of A numbers.
  - **Methodology level:** with the extension of the Lossy Counting algorithm with a fast forgetting mechanism to rapidly detect abnormal behaviours.

Discussion

- **Achievements:**
  - Inability of the Lossy Counting algorithm to detect recent items with abnormal behaviours.
  - The results show that our proposal improved the detection of these recent items.
  - The forgetting mechanism reduces the execution and memory used to compute the data stream, increasing the speedup of the algorithm.

Experimental Work – HHH

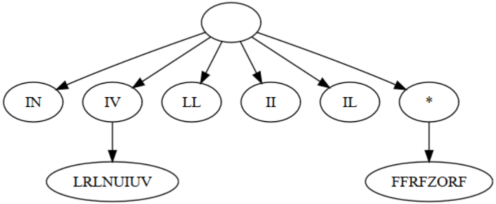
- Each A number is described by (Example phone number "IVLRLNUIUV"):
  - Country code – first two digits – "IV"
  - Sub-range – five digits – "LRLNU"
  - Number – last one, two or three digits – "IUV"
- Use a hierarchical heavy hitters, to find the most frequent items in a specific hierarchy

Hierarchical Heavy Hitters

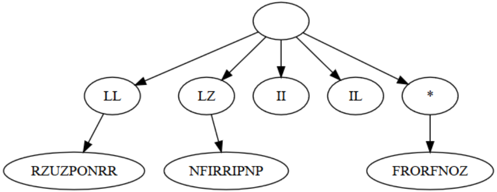
Figure: Stream of calls [fig source:en.antrax.mob]

Experimental Work – HHH – Country Code

Data set 1



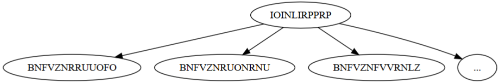
Data set 2



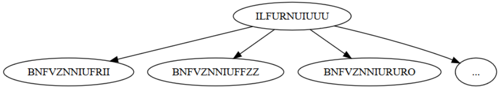
Experimental Work – HHH: ANumber-BNumber

A-numbers that call to too many B-numbers

Data set 1



Data set 2



Experimental Work: Top-k ANumber-BNumber

Data set 1

Rank	ANumber	# BNumber
1	IVVPUPOUUP	26868 (301)
2	LLNZNLIVLZNV	26686 (299)
3	LPVVLPIOIRU	26478 (297)
4	IRUOVOZOFOP	26473 (297)
5	LVUVZFIIUOOZ	26399 (296)
6	LLNVZRLLNOLO	23342 (262)
7	LLVUIZOLFLUZ	19116 (214)
8	LIRVUPPNUZF	13703 (153)
9	ILFFVZZUZNN	12000 (134)
10	IOINLIRPPRP	8595 (96)

Data set 2

Rank	ANumber	# BNumber
1	ILFFVZZUZNN	6002 (207)
2	IOINLIRPPRP	5782 (199)
3	ILFFVZZINZI	5055 (174)
4	LFPNVLNUIPN	3654 (126)
5	LFPNVLNUOVI	3643 (125)
6	LFPNVLNUIRP	3517 (121)
7	ILFURNUIUUU	2855 (98)
8	ILZOVIFOVIF	2782 (96)
9	LLNRUZIORILO	2220 (77)
10	ILZNUPPRNNF	2214 (76)

Experimental Work: ANumber-BNumber HHH vs HH

Data set 1

Rank	ANumber	# BNumber
10	IOINLIRPPRP	8595

- One new ANumbers identified by the HHH when compared with HH

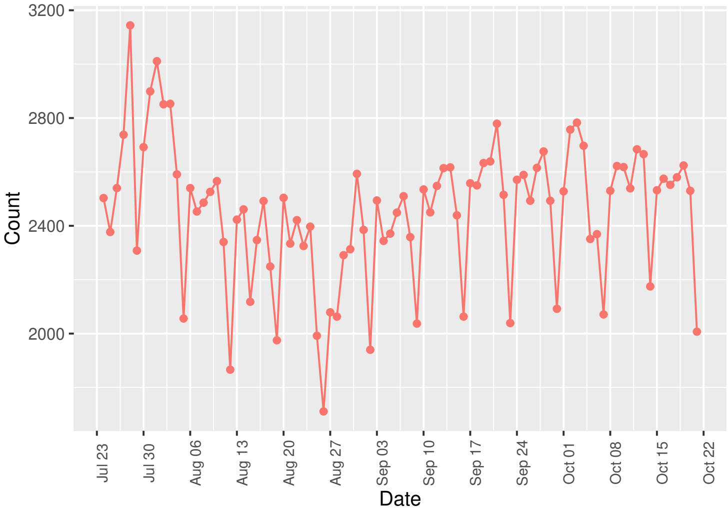
Data set 2

Rank	ANumber	# BNumber
7	ILFURNUIUUU	2855
9	LLNRUZIORILO	2220

- Two new ANumbers identified by the HHH when compared with HH

HyperLogLog

- HyperLogLog is an algorithm for the count-distinct problem, approximating the number of distinct elements in a multiset.



Experimental Work – Frequent sets

- The data is aggregated by intervals of time
  - We analyse 3 interval duration's: 1, 5 and 10 minutes.
- Each interval corresponds to several calls and is described by the set of A numbers of a call.
- The set of A numbers of a call in an interval is a transaction
- the set of transactions is a transaction database
- Use a frequent pattern miner, OpusMiner, to find frequent sets of A numbers in the transaction database

Data set 1

Interval (min)	Transaction (#)	Avg # Calls per Transaction
1	258574	178 ± 147
5	51769	892 ± 735
10	25903	1784 ± 1470

Data set 2

Interval (min)	Transaction (#)	Avg # Calls per Transaction
1	43199	306 ± 235
5	8639	1530 ± 1174
10	4319	3060 ± 2346

Experimental Work – Frequent sets (Interval: 1 min)

The set of A numbers of a call in 1m interval is a transaction

Data set 1

Pairs Element 1	Element 2	Element 3	Frequency	Leverage
LLNZNLIVLZNV	LPVVLPIOIRU		14383	0.040122
IRUOVOZOFOP	LVUVZFIIUOOZ		14272	0.039999
LPVVLPIOIRU	LVUVZFIIUOOZ		14312	0.039967
LLNZNLIVLZNV	LVUVZFIIUOOZ		14270	0.039777
LPVVLPIOIRU	IRUOVOZOFOP		14234	0.039761
LLNZNLIVLZNV	IRUOVOZOFOP		14187	0.039551
LZIVVVZZZUULV	LZIVVVZZZUULL		12090	0.038945
LLNZNLIVLZNV	LPVVLPIOIRU	LVUVZFIIUOOZ	8123	0.024518
LPVVLPIOIRU	IRUOVOZOFOP	LVUVZFIIUOOZ	8096	0.024446
LLNZNLIVLZNV	LPVVLPIOIRU	IRUOVOZOFOP	8093	0.024440

Data set 2

Pairs Element 1	Element 2	Element 3	Frequency	Leverage
LFPNVLNUOVI	LFPNVLNUIPN		1210	0.026576
LFPNVLNUIRP	LFPNVLNUOVI		1193	0.026216
LFPNVLNUIRP	LFPNVLNUIPN		1170	0.025698
LLVUIZIFOFLP	LLNFNRFPNURL		1176	0.023678
LFPNVLNUIRP	LFPNVLNUOVI	LFPNVLNUIPN	980	0.021653
IOINLIRPPRP	ILFFVZZUZNN		5770	0.017198
ILFFVZZUZNN	IIFVRUUUUUU		4007	0.016508
LLVUIZIFOFLP	LLNVVOFIRNIL		796	0.015678
LLNFNRFPNURL	LLNVVOFIRNIL		757	0.014762
LLNFNRFPNURL	LLNFNLONLPON		814	0.014612

Experimental Work – Frequent sets (Interval: 5 min)

The set of A numbers of a call in 5m interval is a transaction

Data set 1

Pairs Element 1	Element 2	Element 3	Frequency	Leverage
LZIVVVZZZUULV	LZIVVVZZZUULL		8484	0.126459
LLNZNLIVLZNV	IRUOVOZOFOP		11034	0.123471
LPVVLPIOIRU	LVUVZFIIUOOZ		10999	0.123231
LPVVLPIOIRU	IRUOVOZOFOP		10976	0.122781
LLNZNLIVLZNV	LVUVZFIIUOOZ		10997	0.122762
LLNZNLIVLZNV	LPVVLPIOIRU		11056	0.122645
IRUOVOZOFOP	LVUVZFIIUOOZ		10903	0.122603
LLNZNLIVLZNV	IRUOVOZOFOP	LVUVZFIIUOOZ	9327	0.116535
LLNZNLIVLZNV	LPVVLPIOIRU	LVUVZFIIUOOZ	9338	0.116186
LPVVLPIOIRU	IRUOVOZOFOP	LVUVZFIIUOOZ	9292	0.116164

Data set 2

Pairs Element 1	Element 2	Element 3	Frequency	Leverage
LLVUIZIFOFLP	LLNFNRFPNURL		942	0.091273
LLVUIZIFOFLP	LLNVVOFIRNIL		749	0.069489
LLNFNRFPNURL	LLNVVOFIRNIL		746	0.069455
LLVUIZIFOFLP	LLNFNRFPNURL	LLNVVOFIRNIL	660	0.062563
IONVNUIOVOPZ	LLNRUZIORILO		686	0.058545
IOIVNLVOZVUU	LLNRUZIORILO		775	0.056892
LLVUIZIFOFLP	IOINLIRIFON		603	0.054615
LLVUIZIFOFLP	LLVUIONLOIVO		581	0.054443
LLNRUZIORILO	LLNFNRFPNURL		640	0.053331
LLNFNRFPNURL	LLVUIONLOIVO		566	0.052935

Experimental Work – Frequent sets (Interval: 10 min)

The set of A numbers of a call in 10m interval is a transaction

Data set 1

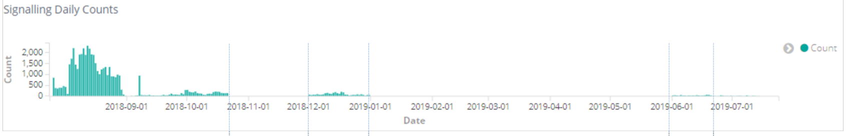
Pairs Element 1	Element 2	Element 3	Frequency	Leverage
LZIVVVZZZUULV	LZIVVVZZZUULL		5607	0.163297
LPVVLPIOIRU	IRUOVOZOFOP		7507	0.139353
LPVVLPIOIRU	LVUVZFIIUOOZ		7514	0.139095
LLNZNLIVLZNV	IRUOVOZOFOP		7518	0.138751
LLNZNLIVLZNV	LPVVLPIOIRU		7572	0.138662
IRUOVOZOFOP	LVUVZFIIUOOZ		7440	0.138373
LLNZNLIVLZNV	LVUVZFIIUOOZ		7519	0.138257
LPVVLPIOIRU	IRUOVOZOFOP	LVUVZFIIUOOZ	6486	0.138210
LLNZNLIVLZNV	LPVVLPIOIRU	IRUOVOZOFOP	6532	0.138202
LLNZNLIVLZNV	LPVVLPIOIRU	LVUVZFIIUOOZ	6525	0.137825

Data set 2

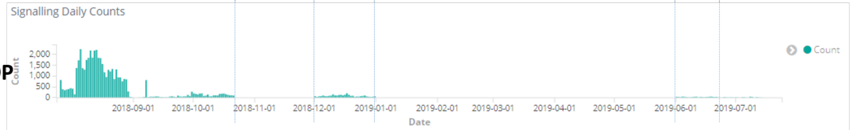
Pairs Element 1	Element 2	Element 3	Frequency	Leverage
LLVUIZIFOFLP	LLNFNRFPNURL		635	0.120668
LLVUIZIFOFLP	LLNVVOFIRNIL		543	0.097824
IOIVNLVOZVUU	LLNRUZIORILO		696	0.096951
IONVNUIOVOPZ	LLNRUZIORILO		566	0.096620
LLVUIZIFOFLP	LLNFNRFPNURL	LLNVVOFIRNIL	506	0.092416
ILFURNUIUUU	IOIVNLVOZVUU		826	0.092347
ILZIOLRZFZV	IOIVNLVOZVUU		791	0.088790
IOIVNLVOZVUU	INVURNOIIIO		736	0.086777
ILZIOLRZFZV	INVURNOIIIO		721	0.085489
IOIVNLVOZVUU	IINRFZRRFUO		666	0.084993

Discussion

LPVVLPIOIRU



IRUOVOZOFOP



Conclusions

The experiments shows:

- Real-time identification of anomalous behaviors.
- Approximate counting algorithms are efficient to identify anomalous beaviours:
  - The Lossy Counting algorithm can be improved with forgetting techniques.
  - efficient to detect recent items with abnormal behaviours:
    burst of calls, repetition and mirror behaviours
- The hierarchical heavy hitters can identify the ranges and numbers with higher volumes of calls with a defined structure
- The frequent sets shows that some ranges in different countries have the same behaviour

Open Issues

- GDPR
- Distributed attacks

Published Papers

- Veloso, B., Martins, C., Espanha, R., Azevedo, R., & Gama, J. (2020). Fraud detection using heavy hitters: a case study. In Proceedings of the 35th Annual ACM Symposium on Applied Computing (pp. 482-489).
- Veloso, B., Tabassum, S., Martins, C., Espanha, R., Azevedo, R., & Gama, J. (2020). Interconnect bypass fraud detection: a case study. Annals of Telecommunications, 75(9), 583-596.
- Veloso, B., Gama, J., Martins, C., Espanha, R., & Azevedo, R. (2020). A case study on using heavy-hitters in interconnect bypass fraud. ACM SIGAPP Applied Computing Review, 20(3), 47-57.

Thank you!

Questions: bruno.m.veloso@inesctec.pt

References I

- G. S. Manku and R. Motwani, "Approximate frequency counts over data streams," in *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pp. 346–357, Elsevier, 2002.
- G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava, "Finding hierarchical heavy hitters in streaming data," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 4, p. 2, 2008.