

FlowChronicle



Synthetic Network Flow Generation Through Pattern Set Mining

Joscha Cüppers¹; Adrien Schoen²; Gregory Blanc³; **Pierre-François Gimenez**²

¹CISPA, Germany; ²Inria, France; ³Telecom SudParis, France



Information system security

How to protect information system?

- ▶ Prevent the attack, detect it, and react
- ▶ Detection with Intrusion Detection System (EDR/NDR)



Information system security

How to protect information system?

- ▶ Prevent the attack, detect it, and react
- ▶ Detection with Intrusion Detection System (EDR/NDR)

Detection relies on observation

- ▶ **System** : OS and applications logs
- ▶ **Network** : network communications

```
2024-05-
06T23:24:16.806598+02:00
stellar-sheep sshd[16039]:
Failed password for pfg from
192.168.1.36 port 48650 ssh2
```

```
"ts": 1591367999.305988,
"id.orig_h": "192.168.4.76",
"id.resp_h": "192.168.4.1",
"id.resp_p": 53, "proto":
"udp", "service":
"dns", "duration":
0.066851, "orig_bytes":
62, "resp_bytes":
141, "conn_state":
"SF", "orig_pkts":
2, "orig_ip_bytes":
118, "resp_pkts": 2,
"resp_ip_bytes": 197
```

Information system security

How to protect information system?

- ▶ Prevent the attack, detect it, and react
- ▶ Detection with Intrusion Detection System (EDR/NDR)

Detection relies on observation

- ▶ **System** : OS and applications logs
- ▶ **Network** : network communications

Main issues

- ▶ Detect APT attacks on long period of time
- ▶ Limit false positives
- ▶ Good quality data?

```
2024-05-
06T23:24:16.806598+02:00
stellar-sheep sshd[16039]:
Failed password for pfg from
192.168.1.36 port 48650 ssh2
```

```
"ts": 1591367999.305988,
"id.orig_h": "192.168.4.76",
"id.resp_h": "192.168.4.1",
"id.resp_p": 53, "proto":
"udp", "service":
"dns", "duration":
0.066851, "orig_bytes":
62, "resp_bytes":
141, "conn_state":
"SF", "orig_pkts":
2, "orig_ip_bytes":
118, "resp_pkts": 2,
"resp_ip_bytes": 197
```



The issue of data in security

Why do we need data?

- ▶ For evaluating security measures, most notably detection
- ▶ For using machine learning in cybersecurity



The issue of data in security

Why do we need data?

- ▶ For evaluating security measures, most notably detection
- ▶ For using machine learning in cybersecurity

Current state of datasets

- ▶ Public datasets are typically run in testbed with no real users
 - ▶ They can suffer from mislabelling, network and attack configurations errors, etc.
 - ▶ We cannot access private data due to confidentiality and privacy reasons
- ⇒ we cannot confidently evaluate intrusion detection systems because of this dubious quality



The issue of data in security

Why do we need data?

- ▶ For evaluating security measures, most notably detection
- ▶ For using machine learning in cybersecurity

Current state of datasets

- ▶ Public datasets are typically run in testbed with no real users
 - ▶ They can suffer from mislabelling, network and attack configurations errors, etc.
 - ▶ We cannot access private data due to confidentiality and privacy reasons
- ⇒ we cannot confidently evaluate intrusion detection systems because of this dubious quality

Our goal: **to use AI to generate synthetic network data**

Other applications of synthetic data

Cyber range realism

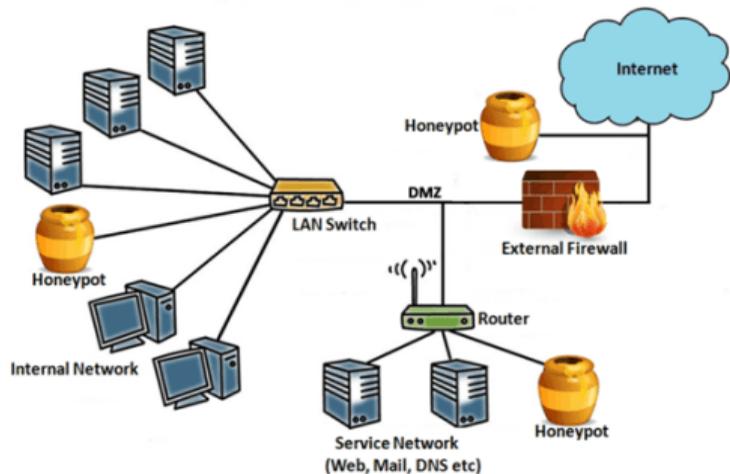
- ▶ Cyber ranges are emulated IT environments with vulnerabilities
- ▶ They are used to train red team (pentesters) and blue team (defenders)
- ▶ They are also useful in education and in CTF competitions
- ▶ Without realistic background network traffic, the scenario can become too easy



Other applications of synthetic data

Honeynets

- ▶ Honeypots (and honeynets) are deliberately vulnerable (networks of) computers to attract and monitor attackers
- ▶ They must be attractive *but* contain nothing of value
- ▶ Honeypots and honeynets must be realistic so attackers (and their tools) generate traces
- ▶ Realistic network communications contribute to this realism



Network data example

Network data

- ▶ Raw data consist of packets, regrouped in conversation
- ▶ Cybersecurity analysis typically rely on network flow records that describe conversations statistically
- ▶ This is the kind of data we want to generate

No.	Time	Source	Destination	Protocol	Length	Info
17	0.709049029	193.51.196.138	131.254.252.23	DNS	126	Standard query response 0x178d AAAA pfgimenez.fr SOA dns12.ovh.net
18	0.709149062	131.254.252.23	185.199.109.153	TCP	74	42578 → 443 [WIN] Seq=0 Min=64240 Len=0 MSS=1460 SACK_PERM TSval=17310600
19	0.718482967	185.199.109.153	131.254.252.23	TCP	74	443 → 42578 [WIN, ACK] Seq=1 Ack=1 Min=59355 Len=0 MSS=1440 SACK_PERM TS
20	0.718506446	131.254.252.23	185.199.109.153	TCP	66	42578 → 443 [ACK] Seq=1 Ack=1 Min=64256 Len=0 TSval=1731066668 TSrec=259
21	0.718615194	131.254.252.23	185.199.109.153	TLSv1.3	590	Client Hello [SN]pfgimenez.fr
22	0.730561279	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=1 Ack=534 Min=143872 Len=0 TSval=2597043199 TSrec=
23	0.742171740	185.199.109.153	131.254.252.23	TLSv1.3	519	Server Hello, Change Cipher Spec, Application Data, Application Data, Ap
24	0.742187989	131.254.252.23	185.199.109.153	TCP	66	42578 → 443 [ACK] Seq=534 Ack=454 Min=63872 Len=0 TSval=1731066692 TSrec=
25	0.741771063	131.254.252.23	185.199.109.153	TLSv1.3	158	Change Cipher Spec, Application Data
26	0.743856561	131.254.252.23	185.199.109.153	TLSv1.3	158	Application Data
27	0.747939849	131.254.252.23	185.199.109.153	TLSv1.3	566	Application Data
28	0.763212420	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=454 Ack=598 Min=143872 Len=0 TSval=2597043226 TSrec=
29	0.765612735	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=454 Ack=690 Min=143872 Len=0 TSval=2597043226 TSrec=
30	0.765612978	185.199.109.153	131.254.252.23	TLSv1.3	131	Application Data
31	0.765763178	131.254.252.23	185.199.109.153	TLSv1.3	97	Application Data
32	0.766914783	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=519 Ack=1190 Min=145408 Len=0 TSval=2597043230 TSrec=
33	0.784918188	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=519 Ack=1221 Min=145408 Len=0 TSval=2597043248 TSrec=
34	0.851903286	185.199.109.153	131.254.252.23	TLSv1.3	324	Application Data
35	0.851204999	131.254.252.23	185.199.109.153	TLSv1.3	101	Application Data
36	0.857904663	131.254.252.23	185.199.109.153	TLSv1.3	206	Application Data
37	0.857947165	131.254.252.23	185.199.109.153	TLSv1.3	293	Application Data, Application Data
38	0.860272768	131.254.252.23	185.199.109.153	TLSv1.3	162	Application Data
39	0.864607086	131.254.252.23	185.199.109.153	TLSv1.3	102	Application Data
40	0.867857307	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=777 Ack=1256 Min=145408 Len=0 TSval=2597043330 TSrec=
41	0.877029312	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=777 Ack=1396 Min=146432 Len=0 TSval=2597043338 TSrec=
42	0.877029938	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=777 Ack=1623 Min=147456 Len=0 TSval=2597043338 TSrec=
43	0.878190357	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=777 Ack=1719 Min=147456 Len=0 TSval=2597043342 TSrec=
44	0.883252688	185.199.109.153	131.254.252.23	TCP	66	443 → 42578 [ACK] Seq=777 Ack=1755 Min=147456 Len=0 TSval=2597043346 TSrec=
45	0.909052163	185.199.109.153	131.254.252.23	TLSv1.3	178	Application Data
46	0.959652475	185.199.109.153	131.254.252.23	TLSv1.3	177	Application Data
47	0.959740916	131.254.252.23	185.199.109.153	TCP	66	42578 → 443 [ACK] Seq=1755 Ack=1000 Min=64128 Len=0 TSval=1731066900 TSrec=
48	0.968002125	131.254.252.23	185.199.109.153	TLSv1.3	101	Application Data
49	0.963572039	185.199.109.153	131.254.252.23	TLSv1.3	178	Application Data
50	0.963712830	131.254.252.23	185.199.109.153	TLSv1.3	136	Application Data, Application Data

```

> Frame 25: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bytes) on interface eth0
> Ethernet II, Src: Intel, Dest: Intel, Protocol: TCP, Src: 131.254.252.23, Dst: 185.199.109.153
> Internet Protocol Version 4, Src: 131.254.252.23, Dst: 185.199.109.153
> Transmission Control Protocol, Src Port: 42578, Dst Port: 443
> Transport Layer Security
  Content Type: Change Cipher Spec (20)
  Version: TLS 1.2 (0x0303)
  Length: 1
  Change Cipher Spec Message
  > TLSv1.3 Record Layer: Application Data Protocol:
  
```

ts,proto,src_ip,dst_ip,dst_port,fwd_packets,bwd_packets,fwd_bytes,bwd_bytes
1730800143,TCP,131.254.252.23,216.58.213.78,443,33,41,5988,1950



Just use an LLM!

State of the part

- ▶ Several approaches have been tried to generate network flows or pcap: VAE, GAN, LLMs
- ▶ The results are not very good:
 - A significant portion of generated data do not comply with network protocols
 - Generated data do not reflect the diversity of the original data
 - The models are not explainable
 - More generally, the dependencies are not well replicated



Just use an LLM!

State of the part

- ▶ Several approaches have been tried to generate network flows or pcap: VAE, GAN, LLMs
- ▶ The results are not very good:
 - A significant portion of generated data do not comply with network protocols
 - Generated data do not reflect the diversity of the original data
 - The models are not explainable
 - More generally, the dependencies are not well replicated

Dependencies

- ▶ Intra-flow dependency
 - the port depends on the destination IP
 - the number of packets depends on the application protocol
- ▶ Inter-flow dependency:
 - DNS query then HTTP(S)
 - IMAP request then HTTP(S)



Contribution: FlowChronicle

Intuition

Network data have a specific structure

- ▶ They are many interleaved and uncorrelated flows
- ▶ They are many hard constraints (HTTP is not over UDP, DNS port is 53, etc.)
- ▶ The inter-flow dependencies are not arbitrary:
 - $A \rightarrow B$, and then $A \rightarrow C$ can happen (for example: DNS request and then an HTTP request)
 - $A \rightarrow B$, and then $B \rightarrow C$ can happen (for example: request to a Website, that then contacts the database)
 - $A \rightarrow B$, and then $C \rightarrow A$ cannot happen: C cannot coordinate with A

With FlowChronicle, we identify *flow patterns* that are constrained with basic networking expert knowledge and are explainable



Contribution: FlowChronicle

FlowChronicle: A Novel Approach

▶ Pattern Language

- Captures intra-flow and inter-flow dependencies
- Summarizes data with non-redundant patterns

▶ Data Generation

- Produces realistic traffic respecting protocols
- Preserves temporal dependencies

▶ Interpretability

- Patterns are interpretable and auditable

What is a pattern?

Frequently occurring substructure in data

Pattern Mining

- ▶ Define the set of possible patterns, named the "pattern language"
- ▶ Find a small set of patterns that best describes the data
- ▶ More precisely, we use the patterns to compress the data: higher the compression, better the patterns

Pattern description

Pattern language

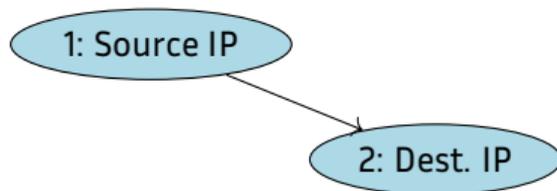
Each pattern has two parts: a partially defined flow, and a Bayesian network

- ▶ **Fixed** values are defined in the partial flow
- ▶ the distribution of **Free** variables is defined in the Bayesian network
- ▶ **Reused** variables are always equal to some **Free** variable

Partial flows

Source IP	Dest. IP	Dest. Port
β_A	8.8.8.8	53
A	β	80

Bayesian Network



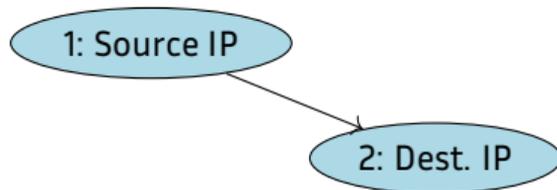
In reality there are more columns!

Pattern description

Partial flows

Source IP	Dest. IP	Dest. Port
β_A	8.8.8.8	53
A	β	80

Bayesian Network



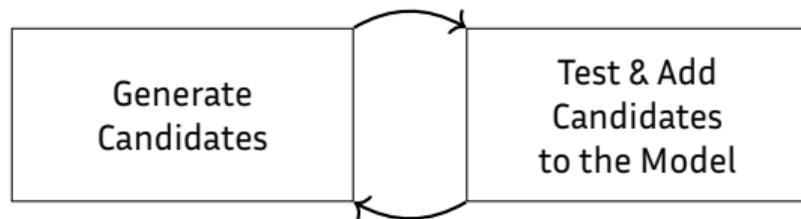
Example

- ▶ Here, there are two flows
- ▶ The first flow is contacting 8.8.8.8 on port 53 (DNS). The source IP is random
- ▶ The second flow has the same source IP as the first flow, and is contacting a destination IP that is random and depends on the first source IP, on port 80 (HTTP)

Our goal is to learn ("mine") such patterns

Mining process

Basic Idea - Two Steps:





Candidate generation

Extending existing pattern with attribute:

Existing Pattern:

Flow	Src IP	Dst IP	Port
1	β_A	8.8.8.8	53
2	A		443

New Pattern Candidate:

Flow	Src IP	Dst IP	Port
1	β_A	8.8.8.8	53
2	A		443
3			3306



Candidate generation

Extending existing pattern with attribute:

Existing Pattern:

Flow	Src IP	Dst IP	Port
1	β_A	8.8.8.8	53
2	A		443

New Pattern Candidate:

Flow	Src IP	Dst IP	Port
1	β_A	8.8.8.8	53
2	A		443
3			3306

Merging existing patterns:

Existing Patterns:

Flow	Src IP	Dst IP	Port
1	β_A	8.8.8.8	53
2	A		443

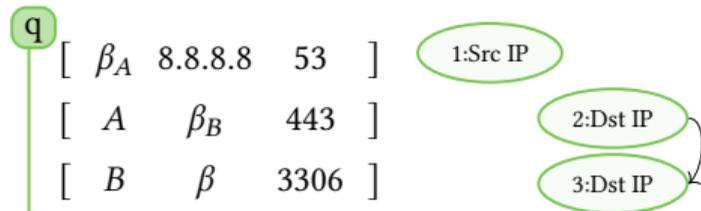
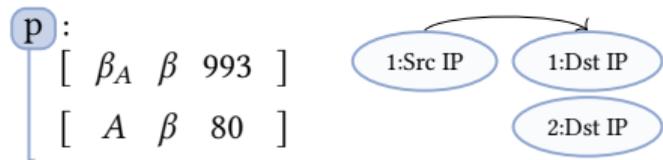
Flow	Src IP	Dst IP	Port
1		8.8.8.8	53

New Pattern Candidate:

Flow	Src IP	Dst IP	Port
1	β_A	8.8.8.8	53
2	A		443
3		8.8.8.8	53

Dataset cover

Model – Pattern and Bayesian Network:



Data and Pattern Windows:

Time	Src IP	Dst IP	Port
12	134.96.235.78	142.251.36.5	993
56	134.96.235.129	8.8.8.8	53
89	134.96.235.78	212.21.165.114	80
113	134.96.235.129	198.95.26.96	443
145	198.95.26.96	198.95.28.30	3306
156	134.96.235.78	134.96.234.5	21
178	134.96.235.36	185.15.59.224	993
206	134.96.235.36	128.93.162.83	80



Loss function: $L(M) + L(D|M)$



Loss function: $L(M) + L(D|M)$

Length of Model:

$$L(M) = L_{\mathbb{N}}(|M|) + \sum_{p \in M} L(p)$$

Length of one pattern:

$$L(p) = L_{\mathbb{N}}(|p|) + \left(\sum_{j=1}^{|p|} L(X[j]|p) \right) + L(BN_p)$$



Loss function: $L(M) + L(D|M)$

Length of Model:

$$L(M) = L_{\mathbb{N}}(|M|) + \sum_{p \in M} L(p)$$

Length of one pattern:

$$L(p) = L_{\mathbb{N}}(|p|) + \left(\sum_{j=1}^{|p|} L(X[j]|p) \right) + L(BN_p)$$

Length of data given the model:

$$L(D | M) = \sum_{p \in M} (L_{\mathbb{N}}(|W_p|) + L(W_p))$$

where:

$$L(W_p) = \sum_{i=1}^{|W_p|} \left(L(t_1 \text{ of } w_i) + \sum_{k=2}^{|p|} L(t_k \text{ of } w_i | t_{i-1}) \right) - \log(\text{Pr}(w_i | BN_p, \{w_j | j < i\}))$$



Generating network flows from a model

Key Steps

Select patterns sample patterns from the model.

Generate timestamp of the first flow sample a timestamp from the timestamp distribution.

Generate delays between the flows sample a delay from the delay distribution.

Fill values in the following order

- ▶ Fixed cells: Predefined values.
- ▶ Free cells: Sampled from the Bayesian network.
- ▶ Reuse cells: Context-based values.



Data quality evaluation

Hard to evaluate

- ▶ No standard metrics
- ▶ Evaluation often partial

Proposition

A set of evaluating metrics:

Realism : could the data actually exist?

Diversity : do we generate the diversity of behavior from the training set?

Novelty : can the generator create data absent from the training set?

Compliance : do the generated data comply with the technical specifications?

We do not consider privacy yet



Experimental protocol

Training data

We use the CIDDS 001 dataset: train on one week of traffic and generate one week of traffic

Baselines

We compare FlowChronicle with:

- ▶ Bayesian networks
- ▶ Variational autoencoders
- ▶ GAN
- ▶ Transformers
- ▶ "Reference"

Reference

Actual data from the same dataset to simulate the best generative method



Non-temporal Evaluation

	Density	CMD	PCD	EMD	JSD	Coverage	DKC	MD	Rank
	<i>Real.</i>	<i>Real.</i>	<i>Real.</i>	<i>Real./Div.</i>	<i>Real./Div.</i>	<i>Div.</i>	<i>Comp.</i>	<i>Nov.</i>	<i>Average</i>
	↑	↓	↓	↓	↓	↑	↓	=	<i>Ranking</i>
Reference	0.69	0.06	1.38	0.00	0.15	0.59	0.00	6.71	-
IndependentBN	0.24	0.22	2.74	<i>0.11</i>	0.27	0.38	0.05	5.47	5.25
SequenceBN	0.30	0.13	2.18	0.08	0.21	0.44	0.02	5.51	3.875
TVAE	0.49	0.18	1.84	0.01	0.30	0.33	0.07	5.17	4.125
CTGAN	0.56	0.15	1.60	0.01	0.15	0.51	<i>0.11</i>	5.70	3.0
E-WGAN-GP	<i>0.02</i>	0.34	<i>3.63</i>	0.02	0.38	<i>0.02</i>	0.07	4.66	7.0
NetShare	0.32	0.28	1.47	0.03	0.36	0.22	0.05	3.82	5.25
Transformer	0.62	<i>0.78</i>	3.62	0.00	<i>0.55</i>	0.03	0.05	<i>3.75</i>	<i>5.375</i>
FlowChronicle	0.41	0.03	2.06	0.02	0.10	0.59	0.02	5.87	2.125

FlowChronicle produces overall the best traffic among the generative methods



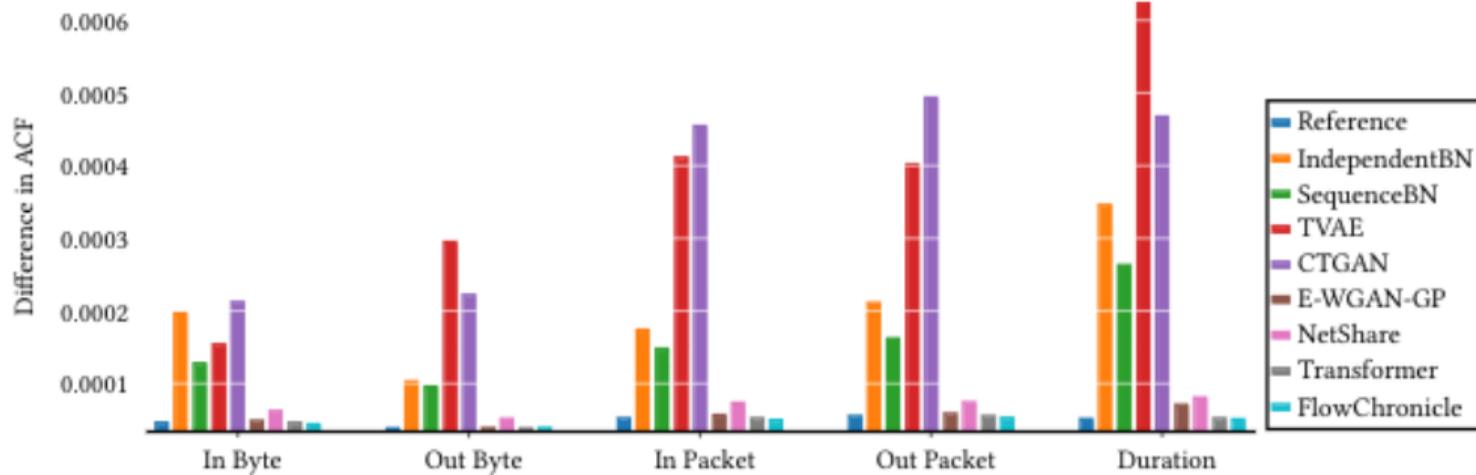
Temporal Dependencies: Numerical Features

Difference in Autocorrelation Functions

- ▶ Autocorrelation function: correlation between the value of a feature and the value of this feature at other timestamps
- ▶ Evaluation: difference between autocorrelation of training data and synthetic data for each feature
- ▶ Lower is better



Temporal Dependencies: Numerical Features





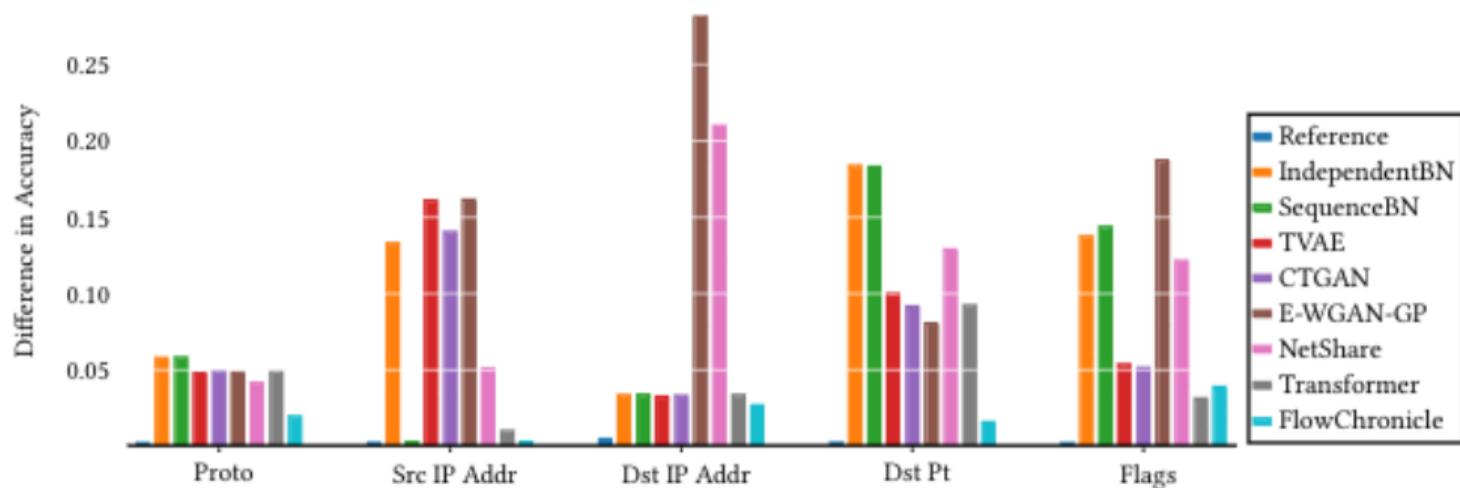
Temporal Dependencies: Categorical Features

Difference in the accuracy of LSTM autoregressive models

- ▶ Train an LSTM to predict the value of a feature
 - Input: Previous value of the feature → autoregressive task
- ▶ Difference of accuracy between two LSTMs on real data:
 - First LSTM trained on the Training Dataset
 - Second LSTM trained on the Synthetic Dataset
- ▶ Lower is better



Temporal Dependencies: Categorical Features





Conclusion

The need of data

- ▶ Good quality data is of utmost importance for security system evaluation and for cyber ranges and honeypots realism
- ▶ One way to achieve such quality is through generative AI

Contributions of FlowChronicle

- ▶ Innovative pattern set mining approach for synthetic network traffic generation
- ▶ Maintains both flow quality and temporal dependencies
- ▶ High performances: outperforms other generative models
- ▶ Auditable patterns: enables explainable and adaptable generation

Future works

We are building upon FlowChronicle for pcap generation