TaintStream: Fine-Grained Taint Tracking for Big Data Platforms through Dynamic Code Translation

Chengxu Yang, Yuanchun Li, Mengwei Xu, Zhenpeng Chen

Yunxin Liu, Gang Huang, Xuanzhe Liu

北京大学
PEKING UNIVERSITY

Microsoft

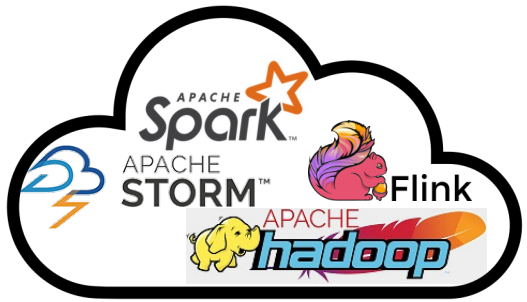
Privacy Compliance in Big Data Platforms

Data audit team

Ensure privacy compliance of data and scripts

Name	Email	Date
Alice	Hi there, …	07/11
Bob	Let's meet …	06/30
Clare	Dear Prof. …	06/01
…	…	…

Provide raw data




```
df.filter(date > 06/01)
  .groupBy(name)
  .aggregate(wordCount)
  …
```

Submit scripts to process data

Data provider
(e.g. end-users and customers)

Data analysts
(e.g. developers and researchers)

Privacy Compliance in Big Data Platforms

Privacy requirements

(e.g. GDPR, HIPAA, COPPA, data provider's requirements, etc.)

User data can be used only if:

- The requester has the permission
- Data is within retention period
- Data is properly anonymized
- The user doesn't choose opt-out
- …

Privacy Compliance in Big Data Platforms

Privacy requirements

(e.g. GDPR, HIPAA, COPPA, data provider's requirements, etc.)

User data can be used only if:

- The requester has the permission
- Data is within retention period
- Data is properly anonymized
- The user doesn't choose opt-out
- …

Current practice:

Lots of manual code/data review

Case-by-case solution for each privacy requirement

Only support coarse-grained data management

Privacy Compliance in Big Data Platforms

Privacy requirements
(e.g. GDPR, HIPAA, COPPA, data provider's requirements, etc.)

User data can be used only if:
- The requester has the permission
- Data is within retention period
- Data is properly anonymized
- The user doesn't choose opt-out
- …

Current practice:

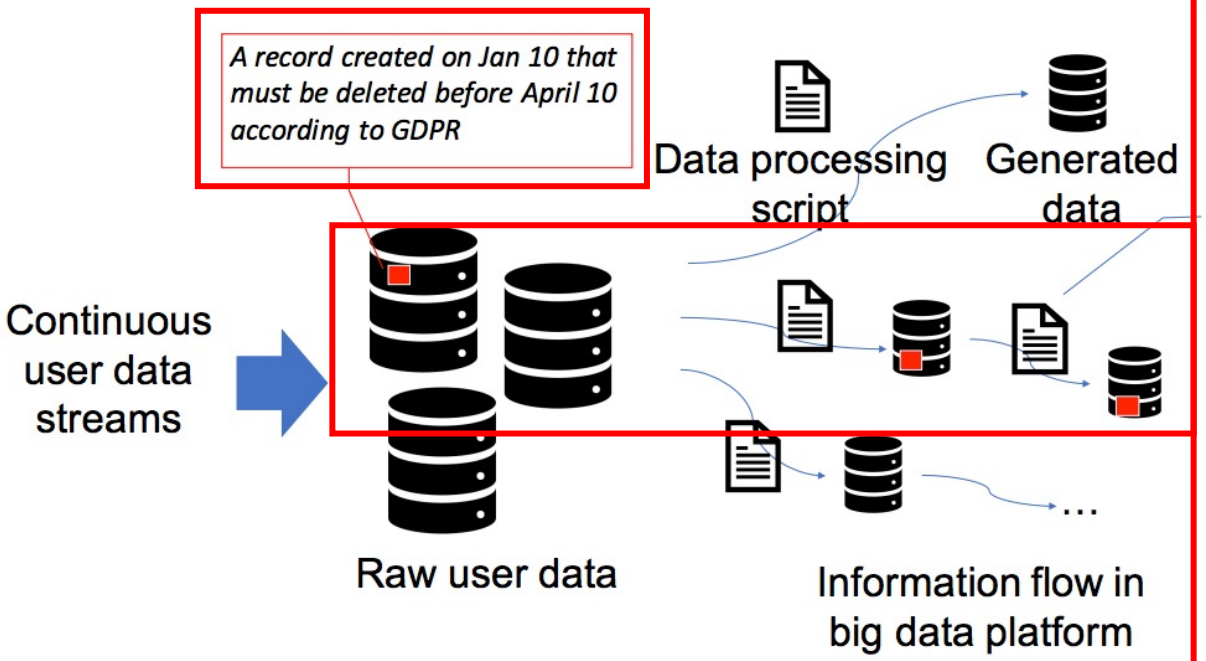
- Lots of manual code/data review
- Case-by-case solution for each privacy requirement
- Only support coarse-grained data management

Problems:

- Time-consuming
- Error-prone
- Data fragmentation & redundancy

A more systematic solution is needed!

An Example



A sample data processing script:

```
# user-defined function that constructs email reply pairs.
func build_pairs(arg, …) {…}
# dataset "emails" contains emails in different conversations.
# select conversations containing more than one email.
convs = emails.groupBy($conversationId)
              .agg(count(*).alias($count),
                   collect_list($msg).alias($msgList))
              .filter($count > 1)
# construct email reply pairs
emails = emails.join(convs, on=$conversationId, "inner")
emails = emails.orderBy($date)
emails = emails.withColumn($replyPairs, build_pairs($msgList))
write_to_hdfs(emails)
```

- Sensitive information may be propagated *across various datasets*
- Data is transformed by *complicated scripts*
- *Different compliance policies* need to apply to all the datasets

Potential Solution: Taint Tracking

```
c = taint_source()
...
a = b + c
...
network_send(a)
```

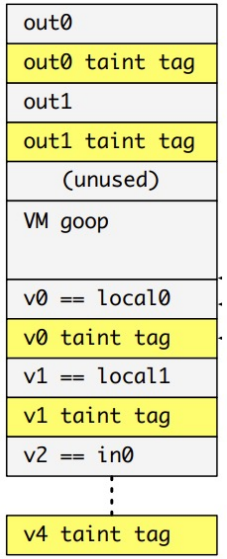
- The original data is tainted with a tag
  - e.g. variable c is tagged as “sensitive”
- Taint tags are propagated during program execution
  - e.g. a is tainted after $a = b + c$
- Privacy policies can be enforced by checking the tags
  - e.g. a cannot be leaked if it’s tagged as “sensitive”

Typical way to achieve taint tracking:

System modification

- Add a taint tag field to each register/variable
- Propagate taint tags in each operator

  [TaintDroid OSDI’10, LIFT MICRO’06]



Op Format	Op Semantics	Taint Propagation
const-op v_A C	$v_A \leftarrow C$	$\tau(v_A) \leftarrow \emptyset$
move-op v_A v_B	$v_A \leftarrow v_B$	$\tau(v_A) \leftarrow \tau(v_B)$
move-op-R v_A	$v_A \leftarrow R$	$\tau(v_A) \leftarrow \tau(R)$
return-op v_A	$R \leftarrow v_A$	$\tau(R) \leftarrow \tau(v_A)$
move-op-E v_A	$v_A \leftarrow E$	$\tau(v_A) \leftarrow \tau(E)$
throw-op v_A	$E \leftarrow v_A$	$\tau(E) \leftarrow \tau(v_A)$
unary-op v_A v_B	$v_A \leftarrow \otimes v_B$	$\tau(v_A) \leftarrow \tau(v_B)$
binary-op v_A v_B v_C	$v_A \leftarrow v_B \otimes v_C$	$\tau(v_A) \leftarrow \tau(v_B) \cup \tau(v_C)$
binary-op v_A v_B	$v_A \leftarrow v_A \otimes v_B$	$\tau(v_A) \leftarrow \tau(v_A) \cup \tau(v_B)$
binary-op v_A v_B C	$v_A \leftarrow v_B \otimes C$	$\tau(v_A) \leftarrow \tau(v_B)$
aput-op v_A v_B v_C	$v_B[v_C] \leftarrow v_A$	$\tau(v_B[\cdot]) \leftarrow \tau(v_B[\cdot]) \cup \tau(v_A)$
aget-op v_A v_B v_C	$v_A \leftarrow v_B[v_C]$	$\tau(v_A) \leftarrow \tau(v_B[\cdot]) \cup \tau(v_C)$
sput-op v_A f_B	$f_B \leftarrow v_A$	$\tau(f_B) \leftarrow \tau(v_A)$
sget-op v_A f_B	$v_A \leftarrow f_B$	$\tau(v_A) \leftarrow \tau(f_B)$
iput-op v_A v_B f_C	$v_B(f_C) \leftarrow v_A$	$\tau(v_B(f_C)) \leftarrow \tau(v_A)$
iget-op v_A v_B f_C	$v_A \leftarrow v_B(f_C)$	$\tau(v_A) \leftarrow \tau(v_B(f_C)) \cup \tau(v_B)$

Challenges

- Privacy compliance is not a set of precise computational rules
  - The taint tags should be flexible enough to support diverse privacy polices

Taint flexibility required

- Big data platforms are complex and difficult to do system hacking
  - Runtime modification is unwanted for security and maintainability reasons

System modification unwanted

- Static approaches usually have oversimplified assumptions for real programs
  - Data schema is difficult to infer statically

Runtime information needed

Our approach (TaintStream)

Idea: *Statically rewrite* the script to let it *self-translate at runtime* and propagate *taint tags embedded in data frames*. Meanwhile, support diverse compliance requirements with *formal and flexible policy definition*.

Our approach (TaintStream)

Idea: *Statically rewrite* the script to let it *self-translate at runtime* and propagate *taint tags embedded in data frames*. Meanwhile, support diverse compliance requirements with *formal and flexible policy definition*.

- Create a parallel element to store taint tags for each data element
  - ✓ The tag type can be flexibly customized according to the privacy policy
- Annotate the scripts through static code instrumentation
  - ✓ Program logic is changed by modifying the scripts instead of the system
- Complete translation at runtime and propagate taint tags on the fly
  - ✓ Data schema is obtained in the dynamic context

Inspirations from source-to-source compilation approaches [JFlow POPL'99, TaintART CCS'16], while tailored for big data processing.

Workflow of TaintStream

```
...
emails = emails.orderBy($date)
emails = emails.withColumn($pairs,
            build_pairs($msgList))
...
```

Original script

conversationId	msg	date
xxx	“Hi, I am …”	10/3
yyy	“Thanks …”	12/4
…	…	…

Original data

Flexible support of diverse privacy management tasks

Data Retention (GDPR)

The raw data and the data inferred from it must be deleted after a certain retention period (e.g., three months)

Tag type: Integer T. # timestamp of the expiration date
Tag init: : Set T to the time when the data expires.
Tag merge: $Minimum(T_1, T_2)$
Enforcement: Periodically scan the datasets and delete any record x if the current timestamp is behind its timestamp T_x.

Access Control (organizational)

The access to certain data should be restricted if the requester is unauthorized.

Tag type: Set of identifiers S. # set of authorized analysts
Tag init: : $S_x \leftarrow$ the analysts with access to the raw data x.
Tag merge: $Intersect(S_1, S_2)$
Enforcement: A data record x is only visible to the analysts in its authorized analysts set S_x.

User Data Erasure (GDPR)

When a user requests to be forgotten, the data collected from him/her and generated based on it must be deleted.

Tag type: Set of identifiers S. # set of involved data providers
Tag init: : $S_x \leftarrow$ the provider the raw data x
Tag merge: $Union(S_1, S_2)$
Enforcement: If a user u asks to be forgotten, delete any record x whose tag S_x contains u.

Dynamic translation

Index	Original	Translated
①	$\phi(source)$	$source_{tagged}$
②	$\phi(df.transformation)$	$\phi(df).\phi(transformation)$
③	$\phi(select(Col))$	$select(\phi(Col))$
④	$\phi(drop(Col))$	$drop(\phi(Col))$
⑤	$\phi(orderBy(Col))$	$orderBy(V_{\phi(Col)})$
⑥	$\phi(filter(Col))$	$filter(V_{\phi(Col)})$
⑦	$\phi(join(df_2, on = Col))$	$join(\phi(df_2), on = V_{\phi(Col)})$
⑧	$\phi(union(df_2))$	$union(\phi(df_2))$
⑨	$\phi(withColumn(str, col))$	$withColumn(str, \phi(col))$
⑩	$\phi(groupBy(Col_1)$ $.agg(F_{agg_1}(Col_2), ...))$	$groupBy(V_{\phi(Col_1)})$ $.agg(tagAgg(T_{\phi(Col_1)}), \phi(F_{agg_1}(Col_2), ...)$ $.map(x \rightarrow (\langle x_1, x_2\rangle, x_3, x_4, ...))$
⑪	$\phi(map(x \rightarrow$ $(F_i(x_{i_1}, x_{i_2}, ...),$ $F_j(x_{j_1}, x_{j_2}, ...),$ $...)))$	$map(x \rightarrow$ $(\langle F_i(V_{x_{i_1}}, V_{x_{i_2}}, ...), tagMerge(T_{x_{i_1}}, T_{x_{i_2}}, ...)\rangle,$ $\langle F_j(V_{x_{j_1}}, V_{x_{j_2}}, ...), tagMerge(T_{x_{j_1}}, T_{x_{j_2}}, ...)\rangle,$ $...))$
⑫	$\phi(reduce(a, b \rightarrow$ $(G_i(a_i, b_i), ...)))$	$reduce(a, b \rightarrow$ $(\langle G_i(V_{a_i}, V_{b_i}), tagMerge(T_{a_i}, T_{b_i})\rangle,$ $\langle ...\rangle, ...)$
⑬	$\phi(column\_name))$	$column\_name$
⑭	$\phi(const)$	$\langle const, \perp\rangle$
⑮	$\phi(Func(Col_1, Col_2, ...))$	$\langle Func(V_{\phi(Col_1)}, V_{\phi(Col_2)}, ...),$ $tagMerge(T_{\phi(Col_1)}, T_{\phi(Col_2)}, ...)\rangle$

Translation rules subset

Parse the code at runtime and recursively apply the rules until no rule can apply.

Two design principles of translation rules:

1. (**Non-interference**) The translated code must not change the effect of the original code.
2. (**Conservativeness**) Should not tolerant false negatives.

Other designs:

- **Correctness guarantee**: Formally prove the non-interference property of the rules
- **Exception handling**: Snapshot and fallback to the conservative taint status on exceptions
- **Performance optimization**: Fuse non-conflicting taint propagation operations to save time

Demo: The datasets in TaintStream with taint tags

- Example: each taint tag is a Boolean value

UserId	Filename	DocumentTitle	Author
System Account	a5eb6988-c9ad-44b...	a5eb6988-c9ad-44b...	System Account
System Account	a5eb6988-c9ad-44b...	a5eb6988-c9ad-44b...	System Account
温 盈盈	20200907-yingying...	The Upper Bound o...	温 盈盈
	学校同意函.pdf	学校同意函	
	学校同意函.pdf	学校同意函	
	学校同意函.pdf	学校同意函	
	健康宝.png	健康宝	
	健康宝.png	健康宝	
	健康宝.png	健康宝	
	自愿返岗承诺书.pdf	自愿返岗承诺书	
	自愿返岗承诺书.pdf	自愿返岗承诺书	
	自愿返岗承诺书.pdf	自愿返岗承诺书	
Chengxu Yang (FA ...	学校同意函.pdf	学校同意函	Chengxu Yang (FA ...
Chengxu Yang (FA ...	学校同意函.pdf	学校同意函	Chengxu Yang (FA ...
Chengxu Yang (FA ...	学校同意函.pdf	学校同意函	Chengxu Yang (FA ...
Chengxu Yang (FA ...	学校同意函.pdf	学校同意函	Chengxu Yang (FA ...
Chengxu Yang (FA ...	学校同意函.pdf	学校同意函	Chengxu Yang (FA ...
Chengxu Yang (FA ...	学校同意函.pdf	学校同意函	Chengxu Yang (FA ...
Chengxu Yang (FA ...	学校同意函.pdf	学校同意函	Chengxu Yang (FA ...
Chengxu Yang (FA ...	自愿返岗承诺书.pdf	自愿返岗承诺书	Chengxu Yang (FA ...

only showing top 20 rows

The output dataset of the original script

UserId	Filename	DocumentTitle	Author
[System Account, ...	[a5eb6988-c9ad-44...	[a5eb6988-c9ad-44...	[System Account, ...
[System Account, ...	[a5eb6988-c9ad-44...	[a5eb6988-c9ad-44...	[System Account, ...
[温 盈盈, true]	[20200907-yingyin...	[The Upper Bound ...	[温 盈盈, true]
[, true]	[学校同意函.pdf, false]	[学校同意函, false]	[, true]
[, true]	[学校同意函.pdf, false]	[学校同意函, false]	[, true]
[, true]	[学校同意函.pdf, false]	[学校同意函, false]	[, true]
[, true]	[健康宝.png, false]	[健康宝, false]	[, true]
[, true]	[健康宝.png, false]	[健康宝, false]	[, true]
[, true]	[健康宝.png, false]	[健康宝, false]	[, true]
[, true]	[自愿返岗承诺书.pdf, false]	[自愿返岗承诺书, false]	[, true]
[, true]	[自愿返岗承诺书.pdf, false]	[自愿返岗承诺书, false]	[, true]
[, true]	[自愿返岗承诺书.pdf, false]	[自愿返岗承诺书, false]	[, true]
[Chengxu Yang (FA...	[学校同意函.pdf, false]	[学校同意函, false]	[Chengxu Yang (FA...
[Chengxu Yang (FA...	[学校同意函.pdf, false]	[学校同意函, false]	[Chengxu Yang (FA...
[Chengxu Yang (FA...	[学校同意函.pdf, false]	[学校同意函, false]	[Chengxu Yang (FA...
[Chengxu Yang (FA...	[学校同意函.pdf, false]	[学校同意函, false]	[Chengxu Yang (FA...
[Chengxu Yang (FA...	[学校同意函.pdf, false]	[学校同意函, false]	[Chengxu Yang (FA...
[Chengxu Yang (FA...	[学校同意函.pdf, false]	[学校同意函, false]	[Chengxu Yang (FA...
[Chengxu Yang (FA...	[学校同意函.pdf, false]	[学校同意函, false]	[Chengxu Yang (FA...
[Chengxu Yang (FA...	[自愿返岗承诺书.pdf, false]	[自愿返岗承诺书, false]	[Chengxu Yang (FA...

only showing top 20 rows

The output dataset when running with TaintStream

Experiment setup

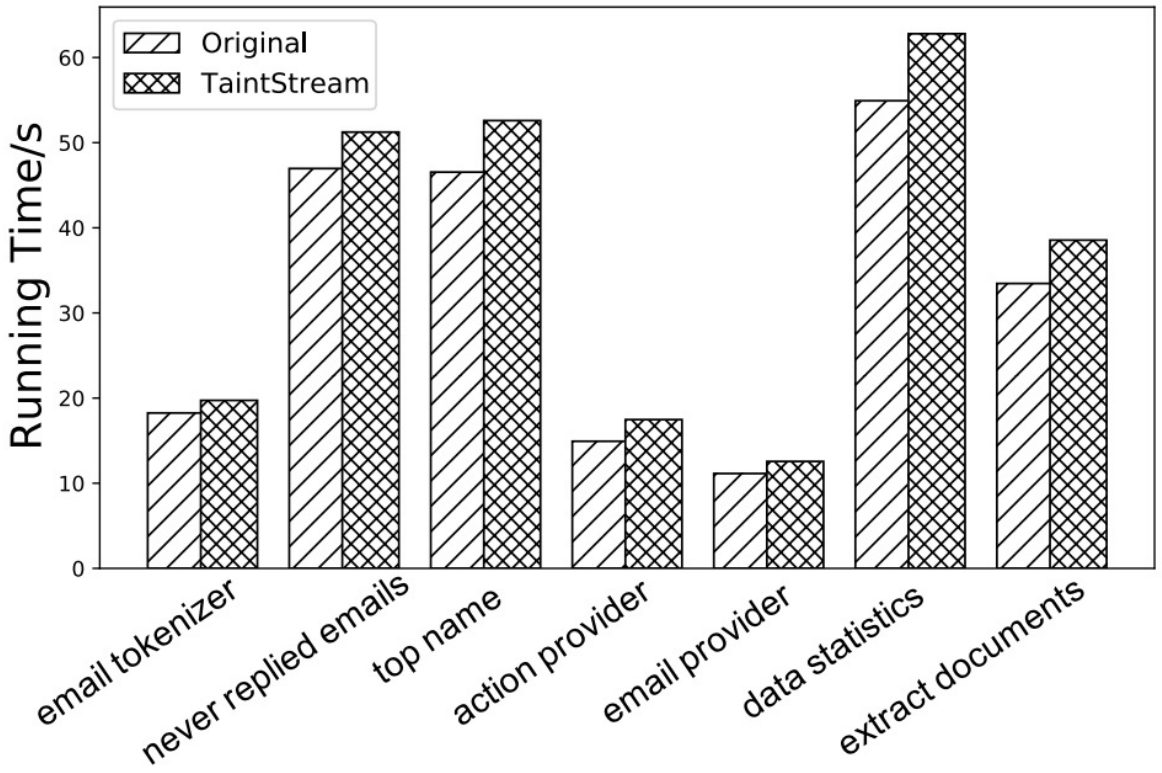
- Query scripts: 7 real-world scripts and 33 self-built scripts
- Baseline: PlanAnlyzer, PySa (SOTA static analyzers)
- Environment: 4-node cluster
- Metrics:
  - precision, recall
  - running time, storage overhead

Accuracy

93.0% precision and 100% recall

Script Name	Column Level			Cell Level	
	PySa	*PlanAnalyzer*	*TaintStream*	*TaintStream*	
	correct/total			precision	recall
Basic					
orderBy_1	4/6	6/6	6/6	100.0%	100.0%
orderBy_2	1/3	3/3	3/3	100.0%	100.0%
select_1	3/6	6/6	6/6	100.0%	100.0%
select_2	1/2	2/2	2/2	100.0%	100.0%
withColumn_1	3/3	3/3	3/3	100.0%	100.0%
withColumn_2	3/5	5/5	5/5	100.0%	100.0%
withColumn_3	3/3	3/3	3/3	100.0%	100.0%
GroupBy					
count_1	1/3	3/3	3/3	100.0%	100.0%
count_2	2/3	3/3	3/3	100.0%	100.0%
count_3	2/5	5/5	5/5	100.0%	100.0%
count_4	2/4	4/4	4/4	100.0%	100.0%
statistics_1	4/5	5/5	5/5	100.0%	100.0%
statistics_2	3/5	5/5	5/5	100.0%	100.0%
Join					
inner_join_1	2/4	4/4	4/4	100.0%	100.0%
inner_join_2	3/5	5/5	5/5	100.0%	100.0%
inner_join_3	2/5	5/5	5/5	100.0%	100.0%
left_join	3/5	5/5	5/5	100.0%	100.0%
right_join	3/5	5/5	5/5	100.0%	100.0%
outer_join	3/5	5/5	5/5	100.0%	100.0%
Structured Data					
array_type_1 *	4/5	5/5	4/5	80.0%	100.0%
array_type_2	2/3	3/3	3/3	100.0%	100.0%
struct_type_1	2/5	1/5	4/5	100.0%	100.0%
struct_type_2	2/3	2/3	3/3	100.0%	100.0%
struct_type_3 *	3/4	2/4	3/4	66.7%	100.0%
UDF					
udf_1	1/4	4/4	4/4	100.0%	100.0%
udf_2	1/1	1/1	1/1	100.0%	100.0%
udf_3	0/1	0/1	0/1	0.0%	100.0%
class_udf_1	2/2	2/2	2/2	100.0%	100.0%
class_udf_2	1/1	1/1	1/1	100.0%	100.0%
Map Reduce					
map_reduce_1	0/2	0/2	2/2	100.0%	100.0%
map_reduce_2	1/4	0/4	4/4	100.0%	100.0%
map_reduce_3	1/4	0/4	4/4	100.0%	100.0%
map_reduce_4 *	1/4	0/4	1/4	21.9%	100.0%
Summary	69/125	103/125	118/125	93.0%	100.0%

System Overhead



Script Name	Original Output Size (Byte)	Tag Size	Overhead
email tokenizer	2,697,017	19,952*x*	0.74%
never replied emails	246,129	5,000*x*	2.03%
top name	16,229	153*x*	0.94%
action provider	32,073	636*x*	1.98%
email provider	10,246,702	44,275*x*	0.43%
data statistics	633	25*x*	3.95%
extract documents	1,097	48*x*	4.38%

12.7% running time overhead

2.06% storage overhead

Take Away

- Light-weight fine-grained taint tracking for big data platforms
- Various privacy management tasks with little overhead
- With no changes to the system and developers

Thanks!

Data and framework:
https://github.com/PrivacyStreams/TaintStream

北京大学
PEKING UNIVERSITY

Microsoft

yangchengxu@pku.edu.cn