PATCG Meeting - London

TEE vs MPC for Trusted Servers

Maxime Vono, Senior Research Scientist (Criteo)
Vincent Minet, Senior Staff Site Reliability Engineer (Criteo)

Contact info:

m.vono@criteo.com; v.minet@criteo.com

Landscape

Author	Proposal	API	PET	Supported use-case	Our analysis
Google Chrome	Privacy Sandbox	Aggregated Attribution Reporting API	TEE	Reporting	Criteo blog post n°3
Google Chrome	Privacy Sandbox	Bidding and Auction Services API	TEE	Bidding	Upcoming article
Google Chrome	Privacy Sandbox	Protected Audience API's key/value server	TEE	Remarketing	N/A
Meta/Mozilla	Interoperable Private Attribution	Interoperable Private Attribution	MPC	• Attribution • Reporting	Criteo blog post n°2
Microsoft Edge	PARAKEET	MaskedLARK	MPC	• Reporting • Campaign optimisation	N/A

Cost for Attribution, Reporting & Campaign Optimisation

API	PET	Use-case	Share of campaign cost (overhead compared to not using PET)	Our analysis
Aggregated Attribution Reporting API	TEE	Reporting	0.004%	Criteo blog post n°3
N/A	TEE	Campaign optimisation via ML training	N/A (x1.1)	Criteo blog post n°3
Interoperable Private Attribution	MPC	Attribution	0.6%	Criteo blog post n°2
Interoperable Private Attribution	MPC	Reporting	0.2%	Criteo blog post n°2
N/A	MPC	Campaign optimisation via ML training	N/A (x180)	Criteo blog post n°2

Assumptions:

- *Computing cost*: $3.2/hour per server
- *Networking cost*: $0.09/GB
- *Campaign cost*: CPM of $2
- *Number of events (1:1 ratio)*: 1M

Main Takeaways

- **Attribution & Reporting** : seems scalable via both MPC-based and TEE-based trusted servers

- **Campaign optimisation** :
  - *MPC-based*: the overhead seems prohibitive. Could it be mitigated via a more efficient implementation?
  - *TEE-based*: the overhead seems acceptable but
    - Potential side-channel attacks --> oblivious ML approaches?
    - How to audit the ML workload to guarantee that the ML training approach meets user privacy?

Opportunities to Discuss

- On-premise TEE hosting

- Private campaign optimisation on a trusted server (e.g. TEE-based)

Back-up / IPA overview

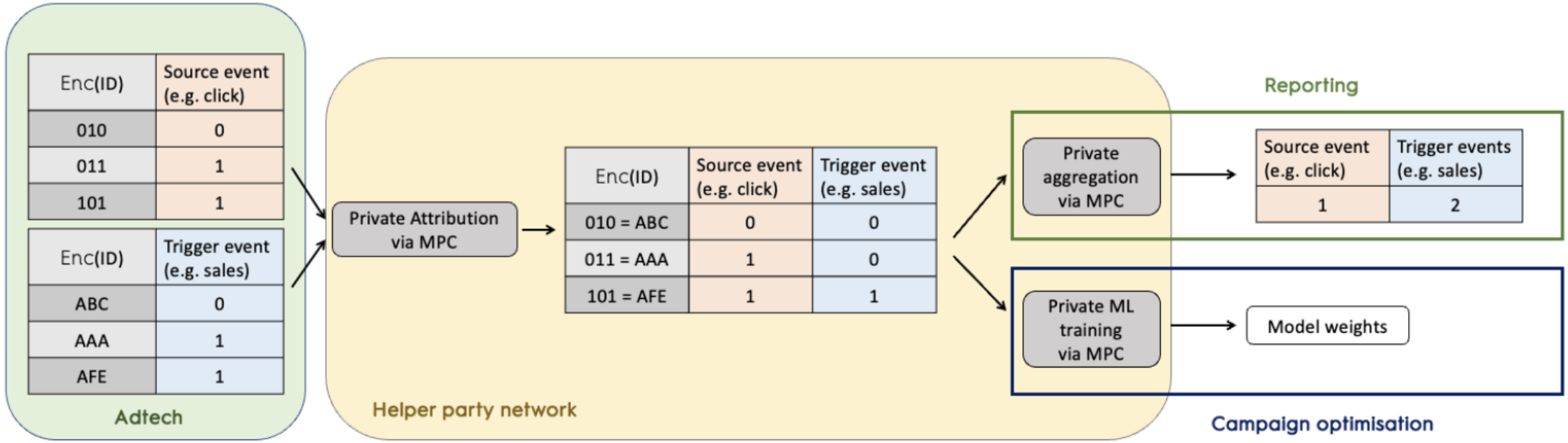
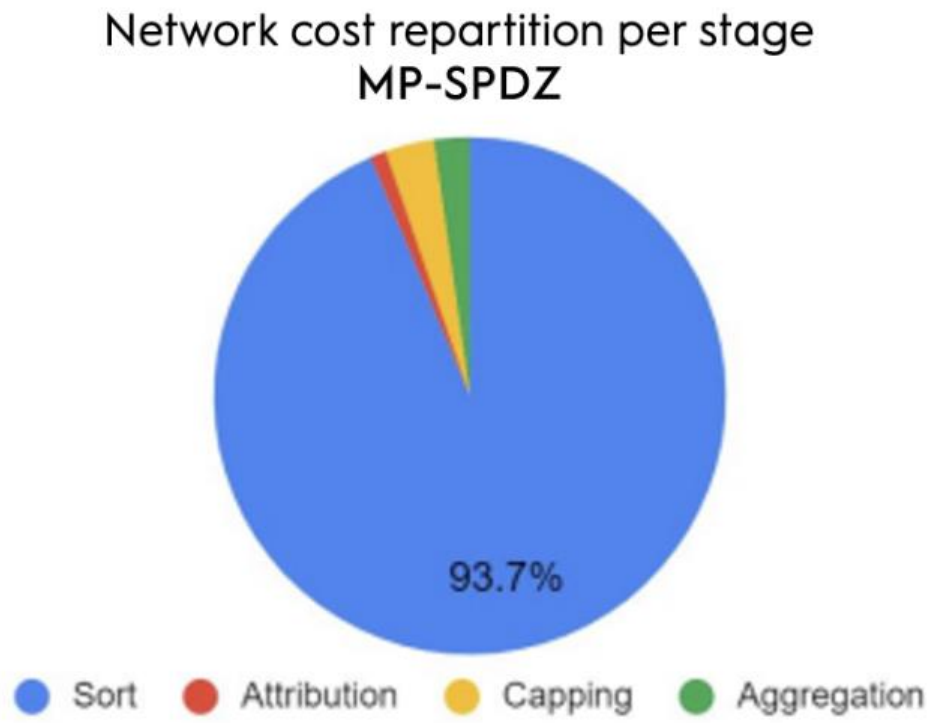


Figure 1 - From fragmented source (e.g. publisher) and trigger (e.g. advertiser) events to reporting and campaign optimisation, passing through attribution.

Back-up / IPA attribution cost (from Meta/Mozilla)

Implementation of IPA on 1M events	Stage	Network Cost	Computing Cost	Total Cost	Share of campaign cost
MP-SPDZ (Aug 2022)	Sorting	$28	$14	$42	2%
MP-SPDZ (Aug 2022)	Attribution	$3	Non available	$4 (estimate)	0.2%
MP-SPDZ (Aug 2022)	Sorting + Attribution	$31	Non available	$46 (estimate)	2.2%
Rust (Feb 2023)	Sorting	$6	Non available	$9 (estimate)	0.4%
Rust (Feb 2023)	Attribution	$2	Non available	$3 (estimate)	0.2%
Rust (Feb 2023)	Sorting + Attribution	$8	Non available	$12 (estimate)	0.6%

Back-up / IPA attribution cost (from Meta/Mozilla)



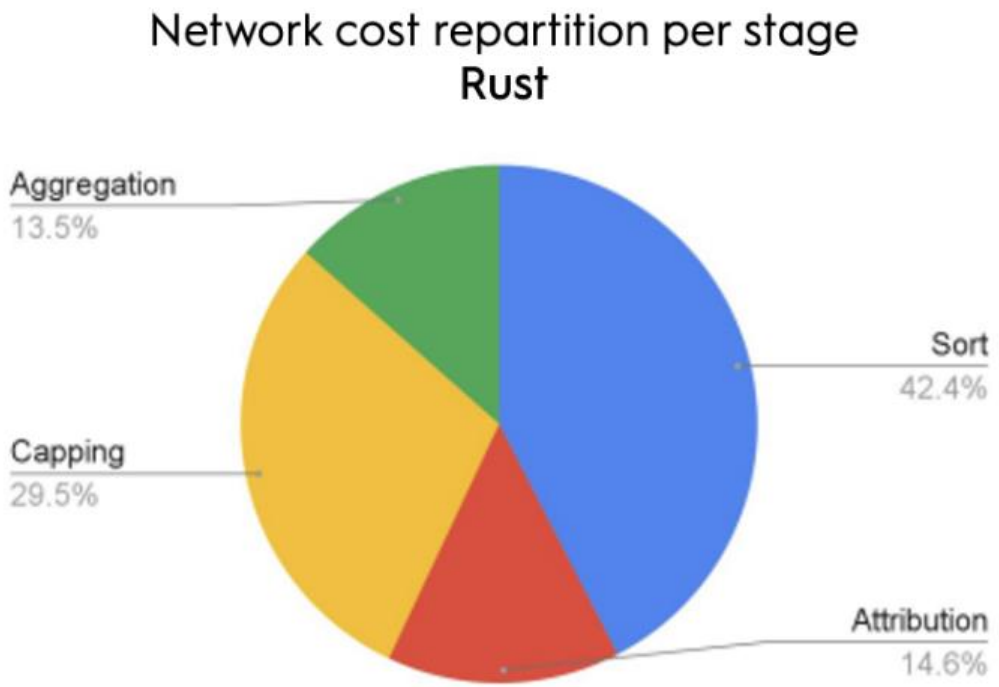


Figure 2. Cost repartition for the IPA Attribution API using MP-SPDZ (August 2022) and Rust (February 2023) implementations.

Back-up / ML training via MPC

- **Dataset** : open-source Criteo dataset publicly available at http://go.criteo.net/criteo-ppml-challenge-adkdd21-dataset-raw-granular-data.csv.gz
- **Pre-processing** : 5,000 sparse features selected via hashing & one-hot encoding + dense 4-dim embedding
- **Model** : Small shallow neural network with 128-dim hidden layer and ReLU activation function
- **Loss** : log-loss
- **Training details** : SGD with batch size 512, Adagrad optimizer with learning rate 0.05, fixed-precision arithmetic with 15 bits after decimal point
- **MPC protocol** :
  - 3 helper parties, each running an Amazon AWS c6i.2xlarge instance in the us-west-1 time zone
  - ABY^3 protocols
  - Implementation based on the CipherCore MPC engine written in Rust

Back-up / ML training via MPC

	Baseline LR - without MPC 89M observations 2^{16} (65,536) features	**Shallow NN - with MPC** 1M observations 5,000 features
Training time (hours)	1	2 (180 if trained on 89M observations)
AUC (higher is better)	0.86	0.85 (-1% vs baseline LR)
Log-loss (lower is better)	0.24	0.27 (-12% vs baseline LR)
Relative uplift in log-loss compared to naive model* (higher is better)	25%	16%

Table 3. Comparison between standard logistic regression (LR) training and the proposed shallow neural network (NN) training with MPC. AUC stands for *area under curve*.

*The naive model always outputs the mean label value of the dataset, in our case 0.1.

Back-up / ML training via TEE

- **Dataset** : open-source Criteo dataset publicly available at http://go.criteo.net/criteo-ppml-challenge-adkdd21-dataset-raw-granular-data.csv.gz
- **Pre-processing** : Hashing & one-hot encoding -> 2^{16} sparse features
- **Model** : Logistic regression
- **Loss** : log-loss
- **Training details** : SGD with batch size 512, Adam optimizer
- **TEE** :
  - AMD SEV-SNP VM on Azure
  - standard_DC4as_v5 (confidential) vs standard_D4as_v5 (non-confidential) VMs
  - On confidential VM: 14% slower; 16% more memory; 6% more CPU usage

Back-up / ML training via TEE

	Baseline LR 89M observations 2^{16} (65,536) features	**Shallow NN - with MPC** 89M observations 5,000 features	**Baseline LR - with VM-based TEE** 89M observations 2^{16} (65,536) features
Training time	1 hour	180 hours	1 hour and 8 minutes

Table 3. Comparison between standard logistic regression (LR) training, the shallow neural network (NN) model considered in our previous article and trained with secure multi-party computation (MPC), and logistic regression training in a vm-based TEE.