Privacy Preserving Ads

Agenda

- Quick Recap of PARAKEET
- MaCAW
- Test setup update for PARAKEET

PARAKEET Private and Anonymized Requests for Ads that Keep Ad Efficiency and Enhance Transparency

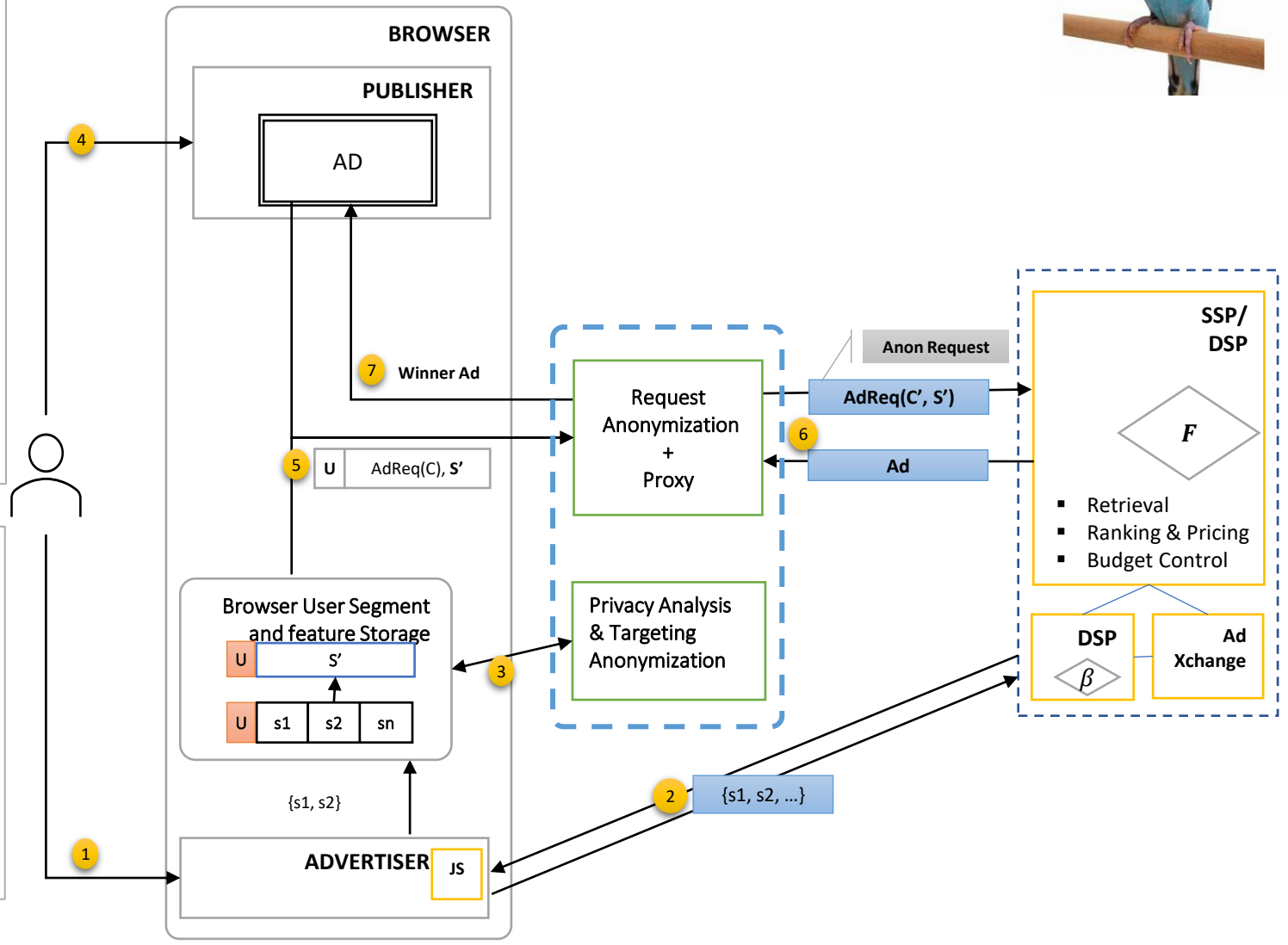
Summary

- Anonymized context and Differentially Private segments $< \mathbf{c}', \mathbf{s}' >$ in the ad request

  supports retrieval, ranking, bid model, auction on Ad network, DSP and SSP servers
- Proxy service to control fingerprinting in ad request
- Direct trade-off between monetization efficacy and privacy parameters
- Less disruptive changes in ads ecosystem and improves privacy through anonymization

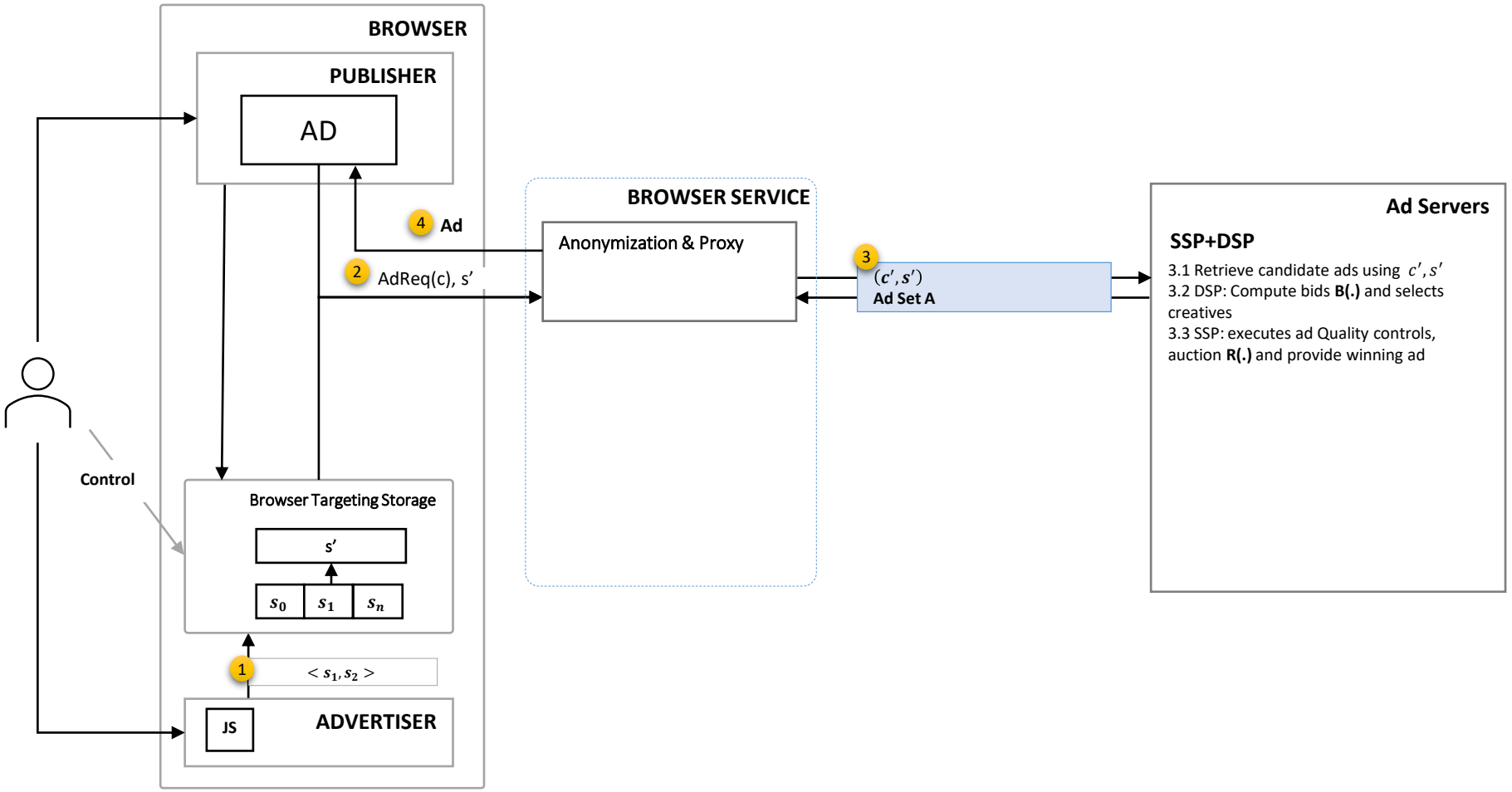
Limitation

- $< \mathbf{c}', \mathbf{s}' >$ are in plaintext in ad request – prone to adversarial privacy attacks – Issue
- Privacy functions on targeting profile and user features **impact Bid models, Ranking and Auction**
- Anonymity thresholds on $\mathbf{c}'$ **impact brand safety controls**



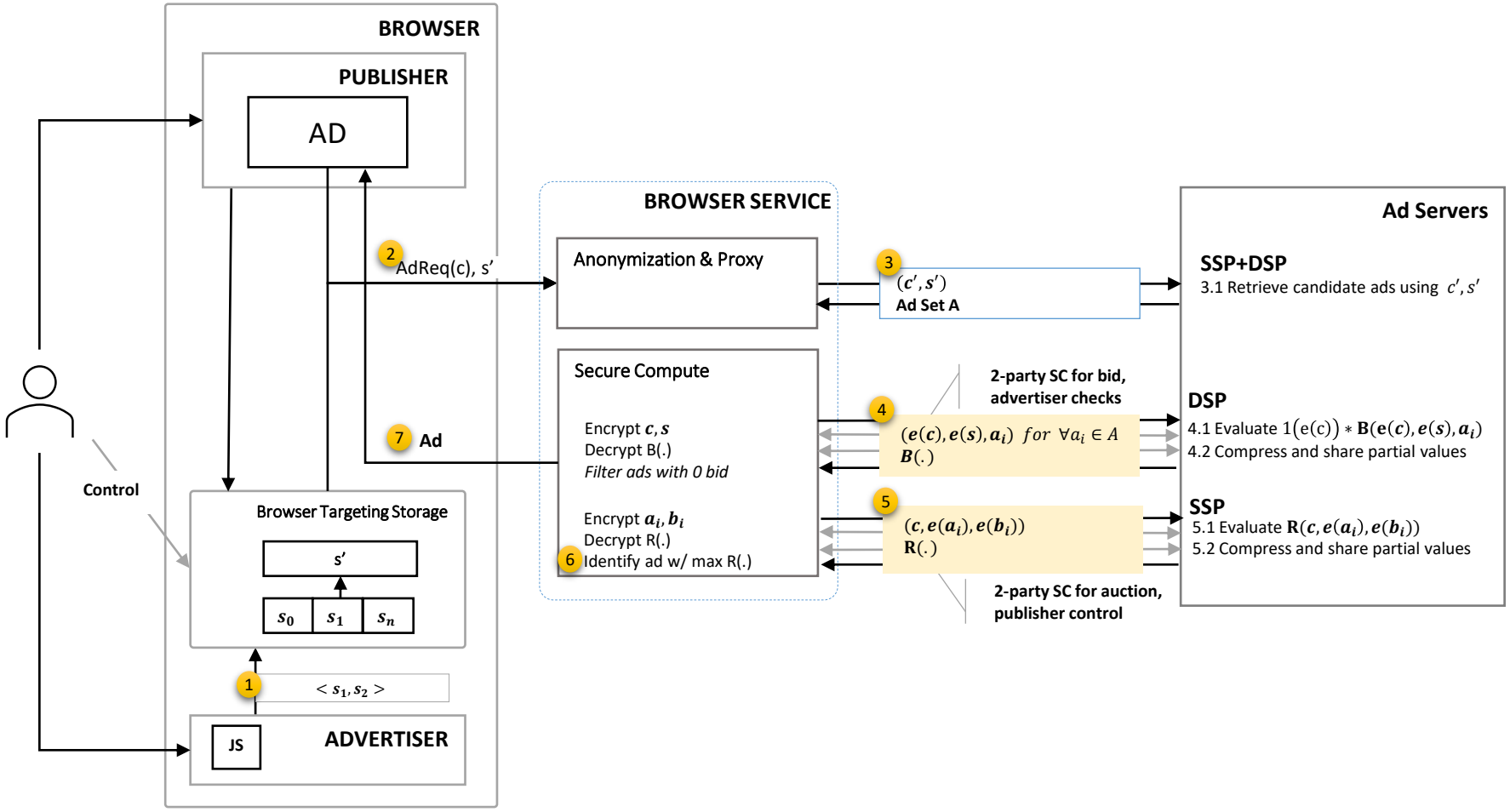
LEGEND:
C – publisher context
C'– anonymized publisher context - link scrubbing, keywords and category scrubbing
S'– differentially private user targeting and features

PARAKEET simplified



LEGEND:
c : publisher context
c' : anonymized publisher context
s': differentially private user segments/features set
B(.): bid functions, executed by DSP servers
R(.): ranking function, executed by SSP servers

MaCAW – Multi-party Computation of Ads on the Web



LEGEND:
c : publisher context
c' : anonymized publisher context
s': differentially private user segments/features set
$e(s)$: encrypted user targeting and features
$e(c)$: encrypted publisher context features
$B(.)$: bid functions, executed by DSP servers
$R(.)$: ranking function, executed by SSP servers

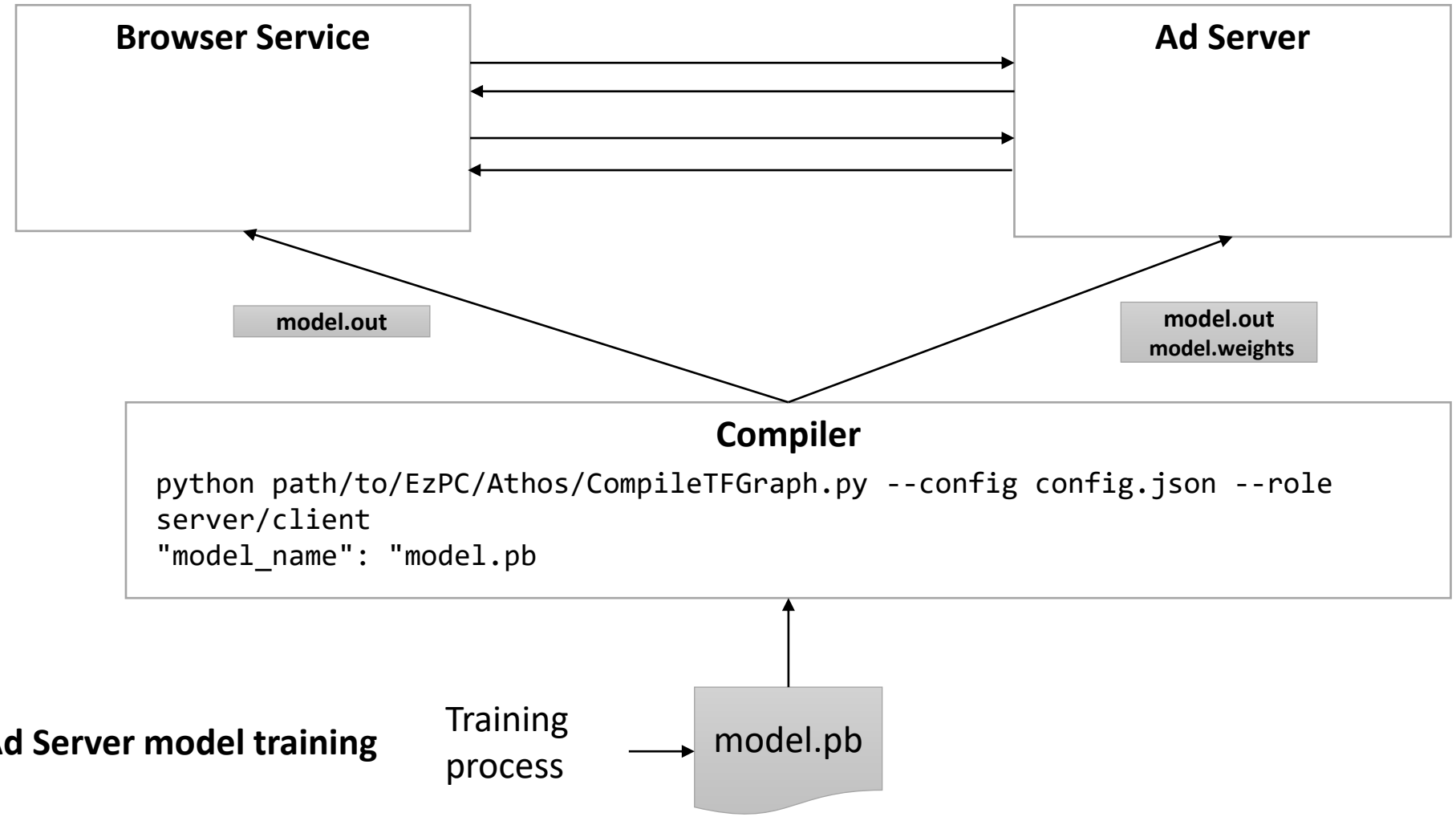
Efficient and generalized secure compute for ad serving

- Step 1: Anonymized $< c', s' >$ for retrieval to limit computation complexity
- Step 2: Two-Party Secure Compute for accurate bid, ranking, brand controls and auction using $< c, s >$

Ad Response

```
const adResponse = {
    'ads': [{
      creative-id: 'hashid'
      creative: 'https://ad-creative.cdn',
      bid-inference-origin: 'https://dsp.example',
      bid-model-format: 'https://dsp.example/bidmodel-structure-for-service.out',
      contextual-signal-processor: 'https://dsp.example/feature-processing.js'
      fallback-bid: 1.00,
      lang: 'en-us',
      adtype: 'image/native'
      },
    ],
    auction-inference-origin: 'https://ssp.ad-network.example',
    auction-model-format: 'https://dsp.example/rankingmodel-structure-for-service.out',
    fallback-ranking: 'highest-bid'
    ad-signal-processor: 'https://ssp.example/feature-processing.js'
    ranking-signals: ['coarse-geolocation', 'coarse-ua', 'encrypted-ad', 'encrypted-bid']
}
```

Working with 2 party secure compute



Leverage EzPC complier for generic code

Key Considerations

Performance and Comm requirements for 2PC

N	Time (ms)	Total Comm (MiB)	Peak Mem Server (MiB)	Peak Mem Client (MiB)
400	201	51.46	65.73	50.89
1600	2585	804.73	693.43	378.01
3200	9913	3215.76	2654.21	1365.53

Improvements

- Mixed mode computation – random subset of variables are encrypted
- 3PC – non-colluding helper party to support secure compute

N	**Time (ms)**	**Client Total Comm (MiB)**	**Server Total Comm (MiB)**	**Helper Total Comm (MiB)**	**Peak Mem Client (MiB)**	**Peak Mem Server (MiB)**	**Peak Mem Helper (MiB)**
400	13	2.44	2.45	0.003	17.39	17.44	17.46
1600	677	39.08	39.09	0.012	148.38	148.02	128.84
3200	1661	156.29	156.32	0.024	558.01	558.57	482.54

PARAKEET test setup

PARAKEET test setup

Polyfill library for ad interests, ad request and rendering

Request:

- Participants for alpha trials

Dev setup

- Advertiser.com JS changes to leverage polyfill library to handle user features
- Ad Request changes to access user features
- Ad server changes to accept user features in request (instead of cookie), and enable ad serving

Goal:

- Implement ad serving flows – user features in browser and runtime read access
- Provide robust ad parameter setup – placement size, location, device etc.
- Data flows and key management for user feature encryption/decryption

Next Steps

- Post your questions or issues via GitHub repo
- We'll provide regular updates on progress on development, test and design
- Participate in the design discussions in bi-weekly calls

Discussion