Web Application for Privacy-Preserving Scheduling

Ágnes Kiss, Oliver Schick, Thomas Schneider
Engineering Cryptographic Protocols Group (ENCRYPTO), TU Darmstadt, Germany
kiss@encrypto.cs.tu-darmstadt.de, oliver.schick92@gmail.com, schneider@encrypto.cs.tu-darmstadt.de

Goal

Overall

- Doodle-like solution for scheduling events with privacy guarantees

Privacy Requirements

- Neither the poll initiator nor any user learns anything about any other user's availabilities besides what can be deduced from the found time slot.
- The bulk of the computation is not performed by the potentially malicious users (who are not always online) but by two non-colluding semi-honest servers.

Scheduling Functionalities

- Participation only once following a unique link
- Votes can be updated by submitting once again
- Poll initiator can add/remove users
- Allow for multiple options (e.g., yes-no-maybe)

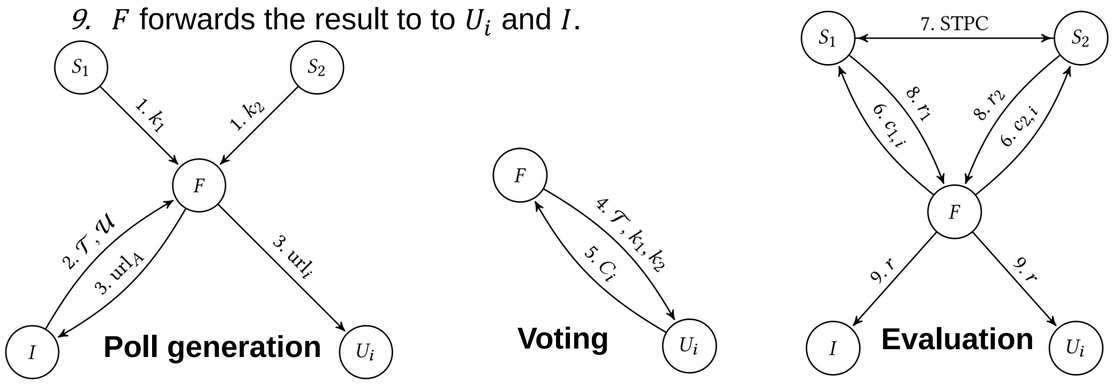
Related Work

Privacy-Enhanced Scheduling

- B. Kellermann, R. Böhme. Privacy-enhanced event scheduling. In *12th Conference on Computational Science and Engineering (CSE'09)*, pages 52–59. IEEE Computer Society, 2009.
- B. Kellermann. Privacy-enhanced web-based event scheduling with majority agreement. In *26th IFIP Information Security Conference (SEC'11)*, volume 354 of IFIP Advances in Information and Communication Technology, pages 235–246. Springer, 2011.
- Both implemented in https://dudle.inf.tu-dresden.de
- **Problems**
  - Reveals the sum of all votes in each time slot to all users
  - Users perform computation, need to be online for any change
  - Restricted functionality, high runtimes

Architecture

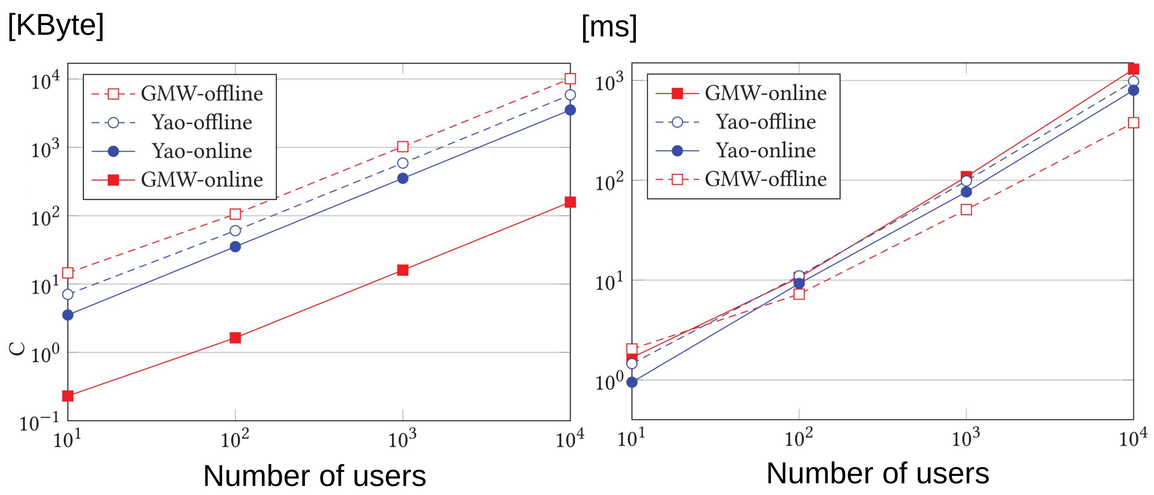
- **Parties**
  - Poll initiator I, users $U_i \in U$
  - Frontend server F
  - Non-colluding backend servers S_1 and S_2
- **Phase I: Poll generation**
  1. F receives public keys k_1 and k_2 from backend servers S_1 and S_2, resp.
  2. I sets up a poll, and specifies the available time slots T and users U.
  3. F sends unique URLs (for submitting their votes) to U_i and I.
- **Phase II: Voting**
  4. U_i receives time slots T, k_1 and k_2 from the frontend server F.
  5. U_i submits two encrypted random shares $C_i = \{c_{1,i}, c_{2,i}\}$ of its availability $c_i = c_{1,i} \oplus c_{2,i}$ (votes for up to 735 slots fit in one 2048-bit RSA ciphertext).
- **Phase III: Evaluation**
  6. F sends the encrypted shares of all users to S_1 and S_2.
  7. S_1 and S_2 decrypt these and perform secure two-party computation to compute shares of the result $r = r_1 \oplus r_2$.
  8. S_1 and S_2 forward r_1 and r_2 to F, resp., who recombines the result r.
  9. F forwards the result to to U_i and I.



Phase III Performance (with 30 Time Slots)

Performance Measurements

- Semi-honest secure two-party computation protocols run between S_1 and S_2: GMW and Yao's garbled circuit
- 2 standard PCs connected with 1 Gbps LAN network



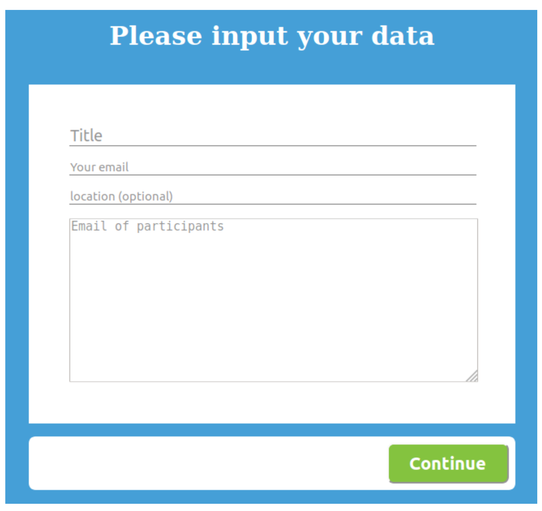
- **Communication in KBytes**
  - Less than 10 MBytes for a poll with 10 000 users
  - More than half of the communication can be shifted offline
- **Runtime in milliseconds**
  - Around a second online runtime for 10 000 users
  - Offline runtimes are similar to online runtimes

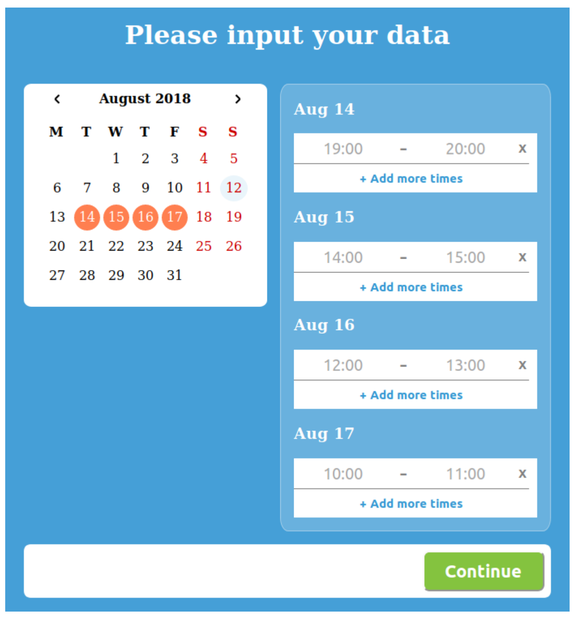
Our Web Application

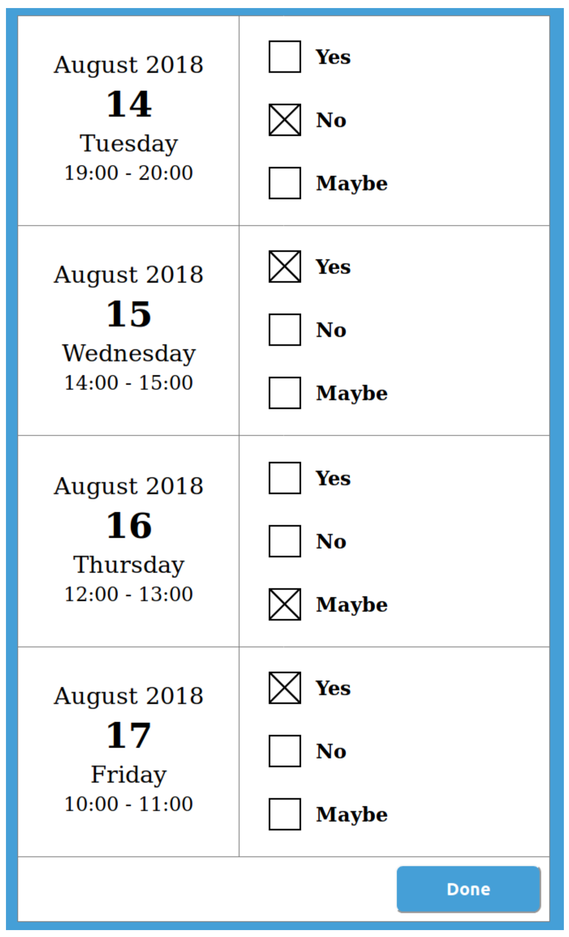
Full-fledged web application for privacy-preserving scheduling

- Efficient Javascript implementation
- Two RSA encryptions per user, takes about one second on a standard PC (also depends on browser)
- Malicious user cannot gain advantage by providing invalid inputs

- **Efficient secure two-party computation on backend servers**
  - Based on the efficient, passively secure ABY framework https://github.com/encryptogroup/ABY
  - D. Demmler, T. Schneider, M. Zohner. ABY – a framework for efficient mixed-protocol secure two-party computation. In *NDSS'15*. The Internet Society, 2015.







Thank you for participating!

CRISP Center for Research in Security and Privacy

CROSSING

TECHNISCHE UNIVERSITÄT DARMSTADT