LazyTAP: On-Demand Data Minimization for Trigger-Action Applications

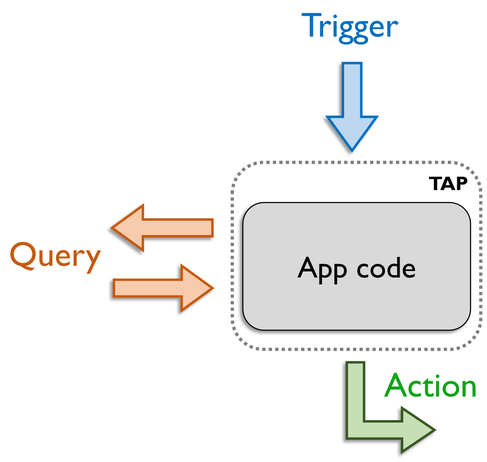
Mohammad M. Ahmadpanah*, Daniel Hedin*,†, and Andrei Sabelfeld*

*Chalmers University of Technology
†Mälardalen University

Paper appeared in IEEE S&P 2023

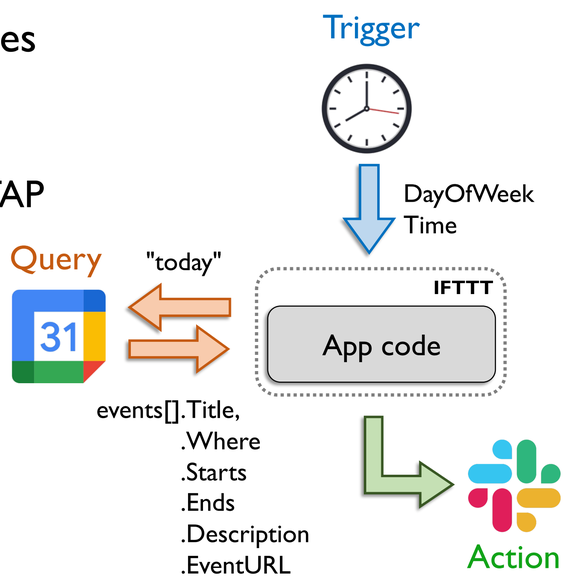
Trigger-Action Platforms (TAPs)

- Connecting otherwise unconnected devices and services
- Upon **Trigger** event, the app performs an **Action**
- **Queries**: additional data source, allowing for complex apps
- Accessing user's private data such as calendar events, watched movies, and locations



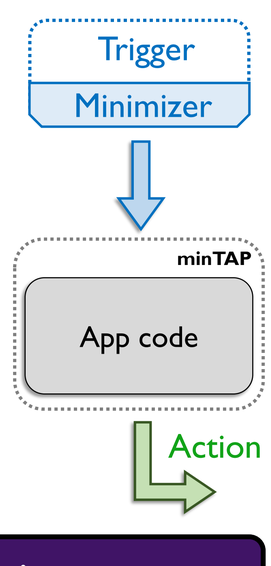
IFTTT: If This Then That

- Over 23M users and 800 services
- ***Push-all*** approach
  Sending all trigger/query data to TAP *independent* of the app code
- Attribute-level overprivilege
  Services should send the *50 most recent events* by default



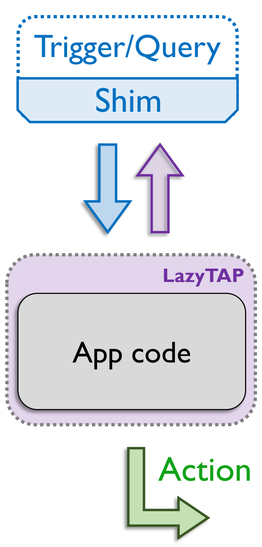
Data minimization

- GDPR: "Only ***necessary*** data should be collected for the specific ***purpose*** the user consented"
- minTAP[USENIX'22]: ***Preprocessing*** approach
  Minimization wrt ill-intended TAP
  Only trigger attributes
  Two modes: Static and Dynamic
  Trusted clients required

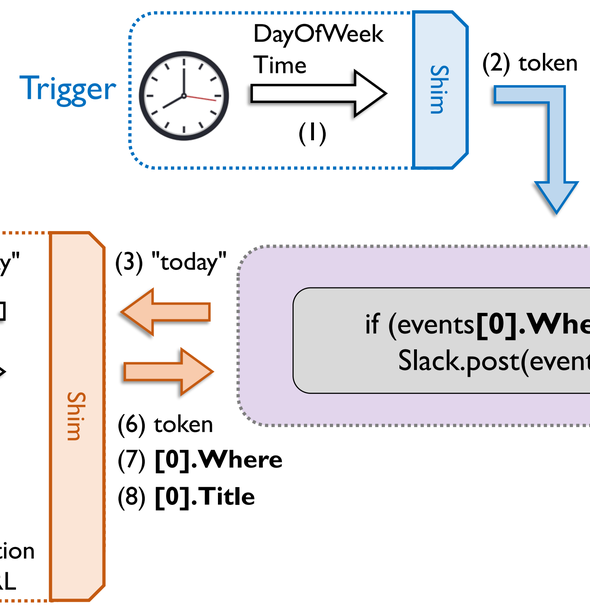


LazyTAP: data minimization by construction

- Minimization wrt **willing-to-minimize** TAP
- **Pull-on-demand** approach
  Pulling attributes of trigger and query data
  Data source unification
- **Input-sensitive** and **fine-grained**
  TAP: lazy runtime supporting *fetch-on-access*
  Trigger/query services: *shim* layers support caching
- **Seamless** for app developers
  Using the same trigger and query APIs
  Supporting *nondeterminism* and *query chains*



LazyTAP by an example



Example: "Every morning, post the title of the first office meeting to Slack"

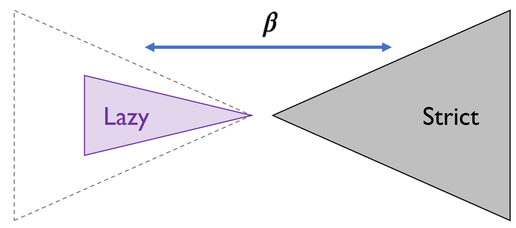
Formal modeling

- Core language: While language with objects

$$e ::= v \mid x \mid e \oplus e \mid f(e) \mid e[e] \mid \{\} \mid T \mid Q(k,e) \mid A(m) \mid () \Rightarrow e$$

- Modeling remote objects, lazy query, and deferred computation

Theorem: LazyTAP is ***correct*** and at least as ***precise*** as preprocessing minimization



Lazy heap extends to a heap isomorphic to strict heap

Evaluation

Minimization **improved** by
95% over **IFTTT**
38% over **static minTAP**

App Id	Distinctive pattern	Total attributes (IFTTT)	Static minTAP	LazyTAP
MeetNotif	Sensitive independent query	2 + (6 * CalendarLength)	2	1 | 2
MovieRec	Nondeterministic query, skip on time	3 + (7 * TraktLength)	TraktLength + 1	1
ParkFinder	Conditional query chain, skip on queries	4 + (6 * CalendarLength) + (7 * YelpLength)	4	1 | 3 | 4

LazyTAP takeaways

Data minimization by construction
- **Pulling** data attributes **on-demand**
- **Input-sensitive** and fine-grained
- Supporting **queries** and **nondeterminism**
- **Seamless** for app developers
- **Correctness** and **precision** formally proved
- Benchmarking: **95%** over IFTTT, **38%** over static minTAP

Lazy runtime by
- Proxied **remote objects**
- Deferred query preparation and property access computation by **thunking**

https://www.cse.chalmers.se/research/group/security/lazytap