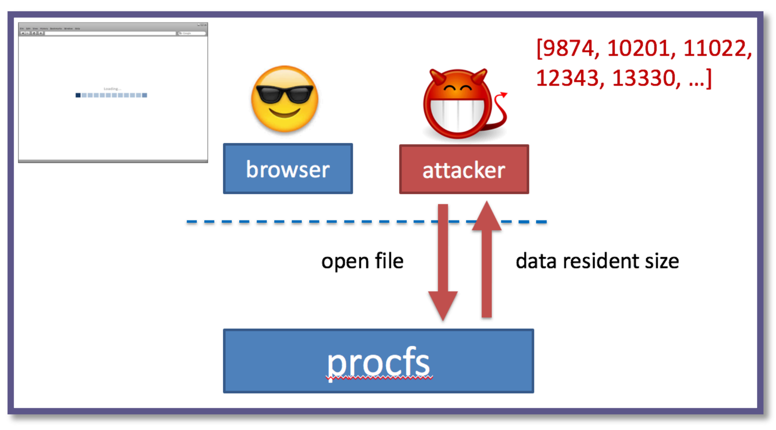
Mitigating Side Channels Using Statistical Privacy Mechanisms

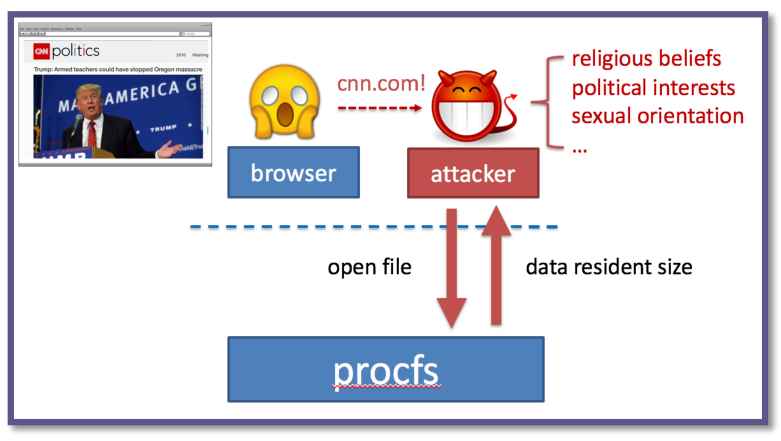
Qiuyu Xiao, Michael Reiter, Yinqian Zhang

qiuyu@cs.unc.edu

Goals

Mitigate side-channels in mobile and cloud computing environment at the same time maintain the utility of the related applications.



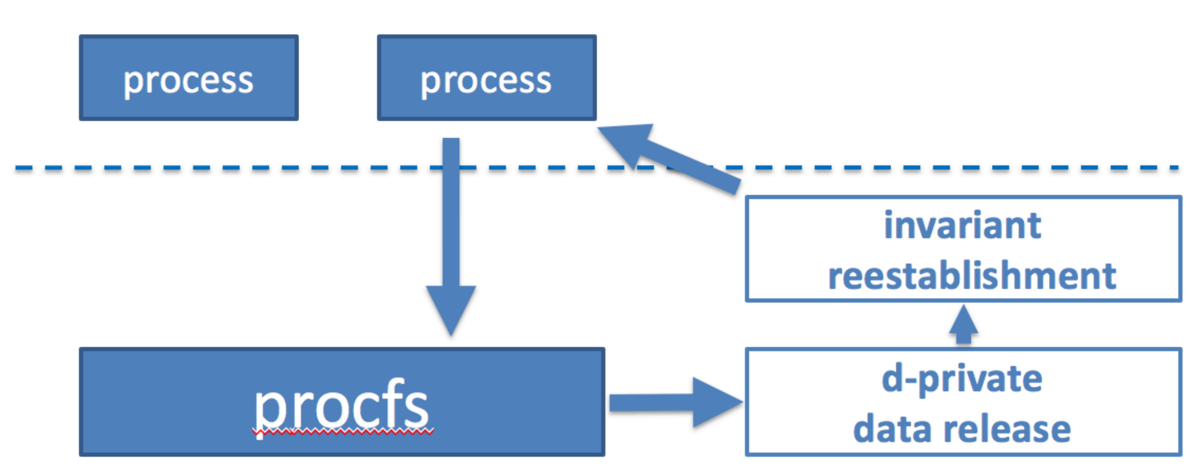


Website Fingerprinting Attack from Storage Side-channel

- **Storage side-channel**: attacker can infer sensitive information from the application run-time data from *proc filesystem*, e.g., fingerprinting website.
- **Timing side-channel**: the private key of the victim VM can be extracted by the attacker VM through the cache timing information.

Approach

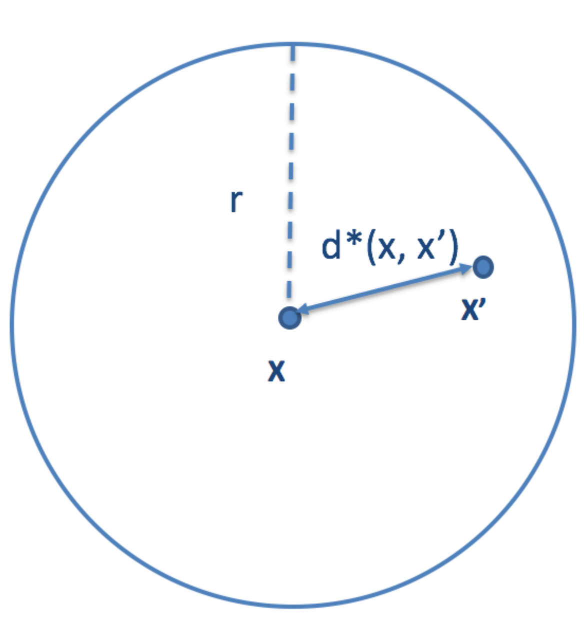
First apply the d-private mechanism to perturb the data, and then reestablish the invariants to maintain the utility of the data before it is outputted by the *proc filesystem*.



d-privacy

- As the attacker observing the obfuscated data, it's hard to distinguish whether the original data is introduced by the sensitive action or the insensitive one.
- M: d-private mechanism, d^*: distance metric.

$$\mathbb{P}(M(X) = \tilde{X}) \leq exp(\epsilon \times d^*(X, X')) \times \mathbb{P}(M(X') = \tilde{X})$$



X: memory stats for *google.com*
X': memory stats for *cnn.com*

Choose a small ϵ make X and X' fall within the indistinguishable range r.

One-field Invariants	Multiple-field Invariants
totalVM ≥ 0	totalVM ≥ sharedVM
utime[i] ≥ utime[i − 1]	hiwaterVM ≥ filePages
starttime[i] = starttime[i − 1]	execVM ≥ filePages + swapEnts

Invariants reestablishment

- Some application depends on the invariants.
- Invariant reestablishment does not erode the privacy achieved by the d-private mechanism.

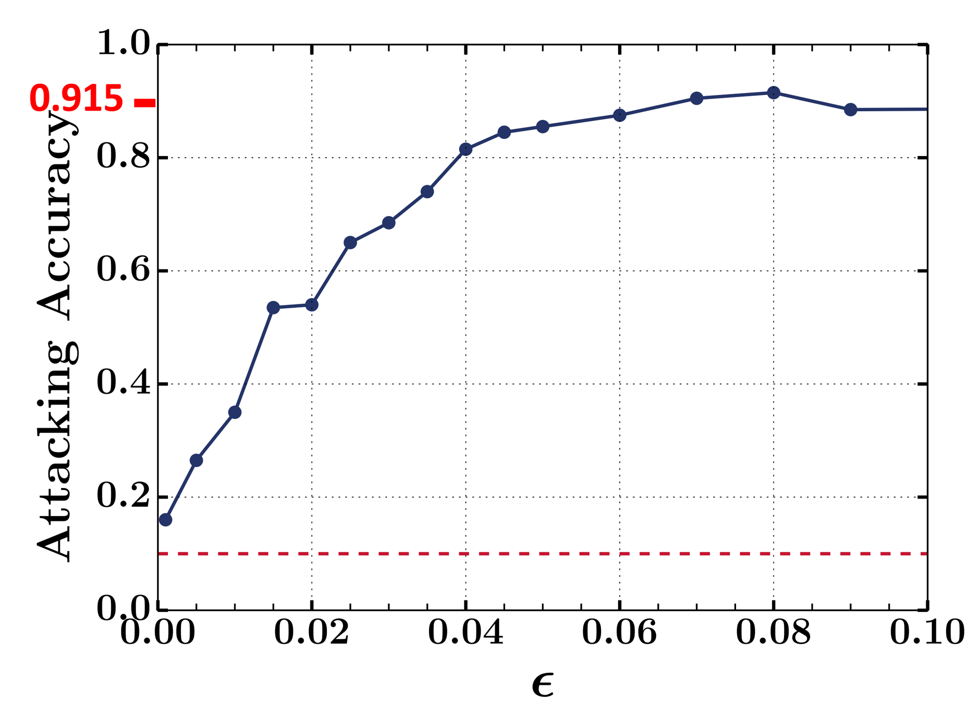
Implementation

- **The prototype system is implemented in Ubuntu 14.04 with kernel version 3.11.**
- **d*-private mechanism is implemented as a kernel routine.**
- **Invariant reestablishment functionality is implemented in a user-space daemon.**

Results

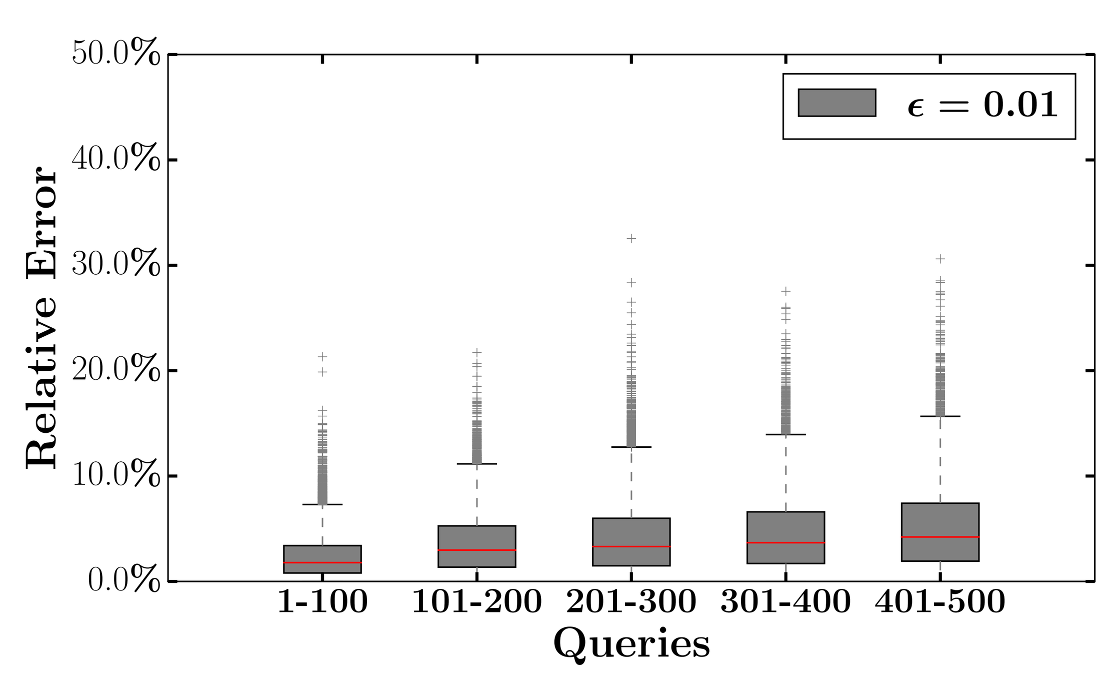
Security evaluation:

- Infer the web page (from a set of ten web pages) visited by the browser based on its *data resident size*.
- Without protection, the attacking accuracy is 0.915.



Utility evaluation:

- Relative error measured for the *data resident size* when ϵ is set to 0.01.
- The relative error is less than 10% most of the time.



- Rank accuracy by the *top command* based on the *resident size field* when ϵ is set to 0.01.
- The utility of *top* is maintained.