CS5430 – System Security Spring 2018

Lecture 27: Privacy

May 7, 2018 *Instructor: Eleanor Birrell*

1 What is Privacy?

There is no universally accepted definition of privacy. It is variously referred to as a right or a privilege, a descriptive concept or a normative one, a question of secrecy or of control. Several efforts have been made to formally define privacy, but none are universally accepted.

One definition that has had a lot of influence over the years, particularly on legal interpretations of privacy, was introduced by Samuel Warren and Louis Brandeis in a Harvard Law Review article in 1890. They described privacy as "the right to be left alone," more specifically the right of individuals to prevent publication of certain information against their will. This definition focused exclusively on secrecy: Warren and Brandeis believed that the right to privacy ceases after consensual publication.

More recently, philosopher Helen Nissenbaum has defined privacy as contextual integrity. By this she means that uses are considered to be privacy violations if they occur outside an appropriate context. Whether a context is appropriate might depend on time, location, purpose, and/or participating principals.

Despite the lack of consensus regarding a formal definition of privacy, most people have their own intuitive notion of what privacy means to them; at the very least, they know it when they see it. These ideals can impose external design constraints—either in the form of legal requirements or customer preferences—that influence how computer systems store, use, and share data.

2 Database Privacy

One context in which technical systems commonly have to deal with problems relating to privacy relates to data storage. Customers (and governments) commonly have privacy preferences regarding what data is stored, under what conditions such storage is permitted, and who is allowed to access this data. Efforts to respect such preferences have given rise to some general guidelines about how to avoid infringing on user privacy: (1) seek consent before using data about users, especially sensitive or identifiable information, (2) minimize data use, (3) limit storage, and (4) avoid linking between different data or data sets.

The primary motivation behind these rules is the repeated observation that unanticipated information can be inferred from disclosed data sets, especially if that data can be linked to another data set. In one now (in)famous example, in the 1990s the Massachusetts Group Insurance Commission released anonymized health records

for state employees. But researchers observed that those records could be cross-referenced against publicly available voter rolls, thus re-identifying many individuals in the dataset, including William Weld who was then governor of Massachusetts. In follow-up work, they found that 87% of Americans can be uniquely identified by date of birth, gender, and zipcode. Similar de-anonymization attacks have successfully occurred against other high-profile anonymous datasets including AOL's search history dataset and the Netflix Prize dataset.

Even with the best intentions, ad-hoc efforts to implement privacy-enhancing principles often prove insufficient to actually ensure privacy. The response from the research community has been an effort to develop more effective privacy-preserving algorithms. These efforts come in two flavors—offline mechanisms, which produce a "private" dataset that can be analyzed offline, and online mechanisms, which act as a filter between the database and the querier—and have given rise to several privacy-enhancing mechanisms.

k-Anonymity. The key idea behind k-anonymity is that individuals should be able to hide in a crowd; individual records are not released unless there are at least k identical rows. In theory, this ensures that individual entries cannot be linked to particular individuals (because uniquely identifying entries are removed from the dataset) and aggregate analysis (e.g., averages) can still be conducted. There are two general techniques for implementing k-anonymity: suppression (in which particular cells are redacted) and generalization (in which precise values, such as age=21, are replaced with coarser-grained values, such as age=20-29). Unfortunately, this technique often removes interesting or valuable data from the dataset, particularly in the case of high-dimensional datasets, rendering the result useless for many applications. It also does not prevent attackers from drawing (correct) conclusions that might adversely affect an individual; for example, a health insurance provider might learn from the k-anonymized results of a research study that John Doe has either cancer or heart disease and might therefore decide to raise his insurance rates, even if it can't learn what disease he has.

Fuzzing. Fuzzing addresses the problem of maintaining individual privacy by adding noise to the data. This can be applied in either the offline or the online case. However, in the offline case, large datasets require adding so much noise that the utility of the results is significantly degraded. In the online case, an interlocutor who can as multiple queries can cancel out the noise and infer the precise (often privacy-sensitive) data.

Differential Privacy. Differential privacy, like k-anonymity but unlike fuzzing, is actually a definition not a mechanism. The idea behind differential privacy is that it shouldn't matter whether or not an individual elects to be included in a particular database; if the adversary can learn the same information either way, then the system

ensures user privacy. More precisely, a mechanism M is said to be ε-differentially private if, for any two databases D_1 and D_2 that differ only on the presence of a single row, and for all sets of answers S that the mechanism might return,

$$\Pr[M(D_1) \in S] \leq e^{\varepsilon} \Pr[M(D_2) \in S]$$

Conveniently, there turn out to be many mechanisms that satisfy differential privacy in practice; on simple mechanism simply adds noise according to a particular distribution (the Laplacian distribution).

Because differential privacy is both philosophically sound and effective (thus far) in practice, it is beginning to see real-world adoption. Google's RAPPOR system uses a differentially private mechanisms for reporting errors and malware, and Apple has adopted a differentially private system for training its suggestions technology.

3 Internet Privacy

Another area where technological systems collide with user privacy is the Internet. Web applications rely on accurately predicting user interests to derive revenue. This behavior arises in both paid services (which try to accurately recommend which products a user might be interested in next) and free services (which both try to personalize content and try to deliver advertisements that the user is likely to click on, thereby deriving revenue). In order to make accurate predictions, applications try to learn as much information as possible about their users; on the Internet, as much information as possible turns out to actually be a lot.

To learn about their users, web applications use cookies to track user behaviors. As discussed last week, cookies are small pieces of local state stored by the browser and sent along with all subsequent requests to the origin that set that cookie. In order to track users, web applications make business agreements with other pages and applications that encourage those third parties to place a request to the tracking server; this request is often in the form of an icon for a button such as "Like this on Facebook" or "Share this on Twitter." Requests can also be for tiny (one pixel by one pixel) images or for content that is not rendered visibly to the user. The unique identifier defined by the cookie indicates which user triggered that request; the exact request and/or the referrer header indicate what site the user has visited. By compiling and analyzing lots of data about where a user has visited, the tracking site can infer detailed interest and demographic profiles for the user; these profiles are used to personalize content and ads, thereby increasing revenue.

Many people find this behavior intrusive or "creepy" and consider it to be a privacy violation, despite the fact that this behavior is consistent with the privacy policies typically posted by major sites (and implicitly agreed to by anyone who visits that site or uses the service). These privacy concerns have given rise to tools designed to prevent tracking and enhance privacy. Perhaps the best known is third-party cookie blocking. The idea behind third party cookie blocking is that cookies

should really only be set by the sites you visit; sites that you have never visited should be allowed to set cookies (that can subsequently be used to track you). A browser can implement third-party cookie blocking by discarding any set-cookie commands issued by sites that don't match the domain the browser is currently visiting. All major browsers now support third-party cookie blocking. Note, however, that typical implementations only block third parties from setting cookies, not from receiving cookies; sites that you have previously visited (e.g., Google or Facebook) will still receive cookies when a third-party site issues a request for content from their site.

Another effort to empower user control over online tracking is the Do Not Track movement. The Do Not Track specification defines an HTTP header field `DNT` that indicates that the user has expressed an explicit preference for or against tracking (according to the specification, the DNT header must not be enabled by default.[1]) All major browsers now support Do Not Track, but compliance with the preference indicated by the `DNT` header is optional and is not currently implemented by many sites.

Finally, most browser now give users the ability to explicitly manage (and delete) cookies stored by the browser. This gives users the facility to manually remove any tracking cookies set by untrusted (or unwanted) tracking services. However, full control of cookies is not necessarily sufficient to prevent all tracking: determined and unscrupulous services have developed additional techniques to track users on the Internet.

Zombie Cookies. Zombie cookies, also sometimes called super cookies, is a general term for a class of techniques that involve storing a unique tracking identifier in some alternative non-cookie local storage. This can be used as an alternative to traditional cookies or, more commonly, can be used as backup storage from which a cookie can be repopulated after deleted by the user (hence the name zombie cookies). The earliest form of zombie cookies used Flash local storage to store the unique identifier. Alternative storage locations include HTML storage (session, local, or global), local state associated with other plugins (e.g., Silverlight local storage), browser history (i.e., by redirecting the browser to visit a subset of a set of special-purpose pages that represents a binary encoding of the unique identifier), or the browser cache (i.e., by caching an auto-generated image that encodes the unique identifier in the RBG values). Tools like Evercookie—a Javascript API that attempts to store backup copies of the unique identifier in as many places as possible and repopulates the tracking cookie—make zombie cookies easy to implement. The diverse set of possible storage locations supported by modern browsers makes the almost impossible to delete.

Browser Fingerprinting. Browser fingerprinting is an alternative way of tracking users. The key idea behind browser fingerprinting is that it is not in fact necessary to

[1]Internet Explorer 10 announced that it would enable `DNT` by default. In response, many servers explicitly ignored DNT headers set by IE10. Microsoft has since reversed this decision.

store a unique identifier for each users because the details of an individual browser are sufficiently close to unique that those browser characteristics themselves can be used to track the individual user. Browser fingerprinting combines a large collection of factors including browser version, OS version, language, time zone, screen resolution, browser settings (use of local storage, global storage, cookies, and DNT), supported plugins, installed fonts, and canvas fingerprinting (the precise way that an HTML5 canvas element renders a fixed set of instructions depends on GPU, device drivers, browser, and OS). Recent studies indicate that 89% of browsers (and 81% of mobile browsers) can be uniquely identified by browser fingerprinting.