Privacy for Infrastructure: Addressing Privacy at the Root

How (Not) To Externalize Privacy Costs Onto Infrastructure Clients

Joshua O’Madadhain (Google)  Gary Young (Google)
Intro
Joshua O’Madadhain

- O-Mad-uh-ahn ;)
- software engineer leading privacy review for infrastructure at Google
- vocalist, horn player, punster
- background
  - search infrastructure
  - social network infrastructure
  - developer tools infrastructure
  - OS Java graph libraries
  - ML for social network analysis
Gary Young

- Privacy Software Engineer
- Baker, guitarist
- Privacy and Security are innately distributed systems properties.

Background
- GMail
- Social Networks
- Infrastructure
Overview

- Definitions
- Purpose
- Perspective
- Future
Definitions
(“What”)

(No further content available on this page.)
What is infrastructure?
infrastructure

systems that provide other systems, or products, with capabilities
Types of infrastructure

- storage systems
- network systems
- data processing systems
- server frameworks
- libraries
- system integrations
- (etc.)
data-agnostic system

- not aware of the kinds of data it handles
- why?
  - generality (work with any kind of data)
  - simplicity (avoid client-specific features)
  - avoiding responsibility
    - "we just handle data, it's the client's job to do it right"
- related: “data processor” (vs. “data controller”)
Purpose
(“Why”)
why infrastructure privacy reviews?

● Can’t we just review the products rather than the infrastructure?
  ○ security: “can’t we just review the applications, not the operating system?” ;)

● Scaling: solving privacy at the infrastructure level benefits all users of all clients
  ○ scaling “traditional engineering” but not the Privacy dimension creates scaling problems for Privacy functions
Perspective
(“How”)
product privacy review concerns

- what (user) data does the product handle (collect, read, write, process)?
  - whose data, and what is it?
- what does the product use the data for?
  - is all the data required, or can some collection/handling be optional?
- where is the data stored, and who has access to it?
- how long is the data retained?
- etc.
infrastructure privacy review concerns: the usual, plus:

how does the infrastructure help its clients to meet their data handling needs?
**infrastructure privacy concerns (1)**

- **data**
  - client-provided: what kinds of data? (data-agnostic?)
  - system-generated: usage logs, error messages, …

- **clients**
  - who are the current, and intended, clients? (how does the system know?)
  - how many clients can the system handle? (not system load, but **configuration load**)

- **use cases**
  - what categories of data are in scope? (personal data?)
  - current uses?
  - planned uses?
  - possible uses?
    - could unplanned use cases present privacy issues?
infrastructure privacy concerns (2)

- access control
  - how is access to the system controlled?
  - how do the clients control access to their data?
  - Is access to the data logged?
    - who, what, when, how, why
    - people who manage a system should not have unfettered access to it

- retention/deletion
  - (how) can clients delete data?
  - how long does each step of deletion take?

- meta
  - what infrastructure does the system depend on? Is it properly configured?
configuration and cost externalization

how much configuration is needed by clients to achieve a good privacy stance?

1. **Zero configuration** (bad stance not possible)
2. Good privacy stance **by default**
3. Good privacy stance **requires per-client configuration/code**
   - who performs this work? clients, infrastructure team, both?
   - how difficult/specialized is it?
4. Good privacy stance **not possible**

configuration documentation is critical: *list sharp edges and how to avoid them*
build vs. buy

build:
- +: can be tailored to your requirements (including privacy)
- -: requires time and investment

buy: (infrastructure- or software-as-a-service)
- +: off-the-shelf, (mostly) predictable costs*
- -: less visibility into/control over privacy stance
  - provider may not have privacy as a differentiator

decide on your requirements before you choose

*costs: including any required investment to get and maintain good privacy stances
Infrastructure privacy warning signs

1. negotiating with infrastructure teams only indirectly via their clients
2. evaluating infrastructure using **product-focused** methodologies
3. **undocumented** infrastructure standards & expectations
4. assuming **off-the shelf infrastructure** will satisfy bespoke privacy innovations/commitments
5. infrastructure goals **not aligned** with client goals
6. with great power comes great vulnerabilities: **Turing-complete** is not your friend
7. uncontrolled **externalization of privacy costs** onto clients
Future
future of infrastructure privacy review

● **systematization**: identifying, documenting, and applying common solutions
  ○ help privacy engineers to apply consistent principles and practices
  ○ help infrastructure teams understand requirements, and criteria for evaluation
  ○ push privacy requirements as deeply into the stack as possible

● **infrastructure-oriented risk frameworks**
  ○ common language for evaluation
  ○ highlighting cost scaling issues
  ○ APIs are contracts; include privacy expectations too

● **annotation and automation**
  ○ discover and report bad configurations via alerting, auditing, lint checks, metrics…
  ○ enforce good configuration automatically based on the nature of the data
    ■ annotations for data => automating configuration & use case exclusion
thank you!

questions?

Offline Questions: jrtom@google.com, gdyoung@google.com