Assuring Real-World Differential Privacy

José Manuel Calderón Trilla

Scott Moore

Motivation for Differential Privacy

Motivation for Differential Privacy

- We want to learn facts about *populations* without revealing anything new about *individuals*

Motivation for Differential Privacy

- We want to learn facts about *populations* without revealing anything new about *individuals*
- Differential Privacy can make up one part of a system that accomplishes this

Motivation for Differential Privacy

- We want to learn facts about *populations* without revealing anything new about *individuals*
- Differential Privacy can make up one part of a system that accomplishes this
    - Confidentiality, policy enforcement, etc. all still important.

Motivation for Differential Privacy

- We want to learn facts about *populations* without revealing anything new about *individuals*
- Differential Privacy can make up one part of a system that accomplishes this
    - Confidentiality, policy enforcement, etc. all still important.
- Differential Privacy provides a bound on the additional information that can be learned about an individual if they choose to take part in an analysis

Motivation for Differential Privacy

- We want to learn facts about *populations* without revealing anything new about *individuals*
- Differential Privacy can make up one part of a system that accomplishes this
    - Confidentiality, policy enforcement, etc. all still important.
- Differential Privacy provides a bound on the additional information that can be learned about an individual if they choose to take part in an analysis
    - Information could still be learned via other means! e.g., if you are statistically similar to a population that takes part, the computation will still reveal some information about you

Motivation for Differential Privacy

- We want to learn facts about *populations* without revealing anything new about *individuals*
- Differential Privacy can make up one part of a system that accomplishes this
  - Confidentiality, policy enforcement, etc. all still important.
- Differential Privacy provides a bound on the additional information that can be learned about an individual if they choose to take part in an analysis
  - Information could still be learned via other means! e.g., if you are statistically similar to a population that takes part, the computation will still reveal some information about you
- Differential privacy as a bound on relative risk

What does it mean to verify DP?

What does it mean to verify DP?

- A few interesting questions (Gaboardi, 2018):

What does it mean to verify DP?

- A few interesting questions (Gaboardi, 2018):
    - Given a Program, P, is P differentially private?

What does it mean to verify DP?

- A few interesting questions (Gaboardi, 2018):
    - Given a Program, P, is P differentially private?
    - Given a differentially private program, DP, does DP maintain its privacy/accuracy guarantees?

What does it mean to verify DP?

- A few interesting questions (Gaboardi, 2018):
    - Given a Program, P, is P differentially private?
    - Given a differentially private program, DP, does DP maintain its privacy/accuracy guarantees?
    - Does DP perform its computation efficiently?

What does it mean to verify DP?

- A few interesting questions (Gaboardi, 2018):
    - Given a Program, P, is P differentially private?
    - Given a differentially private program, DP, does DP maintain its privacy/accuracy guarantees?
    - Does DP perform its computation efficiently?
- The answer to any of these could be seen as ‘verifying’ some aspect of a Diff. Priv. system

Traditional Verification

Traditional Verification

- Correct-by-construction

Traditional Verification

- Correct-by-construction
    - Often type-system-based (Fuzz, CompCert, DeepSpec)

Traditional Verification

- Correct-by-construction
    - Often type-system-based (Fuzz, CompCert, DeepSpec)
- Static Analysis

Traditional Verification

- Correct-by-construction
  - Often type-system-based (Fuzz, CompCert, DeepSpec)
- Static Analysis
  - Abstract interpretation (Astree, Infer, ErrorProne)

Traditional Verification

- Correct-by-construction
    - Often type-system-based (Fuzz, CompCert, DeepSpec)
- Static Analysis
    - Abstract interpretation (Astree, Infer, ErrorProne)
- State-space exploration

Traditional Verification

- Correct-by-construction
    - Often type-system-based (Fuzz, CompCert, DeepSpec)
- Static Analysis
    - Abstract interpretation (Astree, Infer, ErrorProne)
- State-space exploration
    - Model-Checking (often used in Circuit design, increasingly used in software)

Traditional Verification

Traditional Verification

- To rephrase: There’s no such thing as ‘proving a program is correct’, it’s really ‘proving a program meets specification X’

Traditional Verification

- To rephrase: There’s no such thing as ‘proving a program is correct’, it’s really ‘proving a program meets specification X’
- Instead of “is program P correct?”, we ask “Does program P perform an out-of-bounds array access?”

Traditional Verification

- To rephrase: There’s no such thing as ‘proving a program is correct’, it’s really ‘proving a program meets specification X’
- Instead of “is program P correct?”, we ask “Does program P perform an out-of-bounds array access?”
    - This we can verify, our program is now ‘verified’, but it does not mean the program does what it is meant to do!

Traditional Verification

- To rephrase: There’s no such thing as ‘proving a program is correct’, it’s really ‘proving a program meets specification X’
- Instead of “is program P correct?”, we ask “Does program P perform an out-of-bounds array access?”
    - This we can verify, our program is now ‘verified’, but it does not mean the program does what it is meant to do!
- We are already comfortable with this nuance with regards to static types and garbage collection (i.e. we rule out *certain and specific* problems)

Traditional Verification

- To rephrase: There's no such thing as 'proving a program is correct', it's really 'proving a program meets specification X'
- Instead of "is program P correct?", we ask "Does program P perform an out-of-bounds array access?"
    - This we can verify, our program is now 'verified', but it does not mean the program does what it is meant to do!
- We are already comfortable with this nuance with regards to static types and garbage collection (i.e. we rule out *certain and specific* problems)
- Historically, not very good at probabilistic reasoning, which is why we are here!

Fuzz-like

Fuzz-like

Fuzz-like

- Anyone who gets a fuzz program past the type checker has a ‘good’ program. That’s great!

Fuzz-like

- Anyone who gets a fuzz program past the type checker has a ‘good’ program. That’s great!
- You are tied to fuzz-the-language

Fuzz-like

- Anyone who gets a fuzz program past the type checker has a ‘good’ program. That’s great!
- You are tied to fuzz-the-language
- Only a specific set of primitives

Fuzz-like

- Anyone who gets a fuzz program past the type checker has a ‘good’ program. That’s great!
- You are tied to fuzz-the-language
- Only a specific set of primitives
- Limited inter-op with other systems

PINQ/Airavat-like

PINQ/Airavat-like

- “Just works”…

PINQ/Airavat-like

- “Just works”…
- … if your computation fits their model

Formalized Model à la CompCert

Formalized Model à la CompCert

Formalized Model à la CompCert

- You had a differential privacy problem

Formalized Model à la CompCert

- You had a differential privacy problem
- Now you have a differential privacy problem and a theorem prover problem.

Correct By Construction

Correct By Construction

- Lots of current research focuses on correct by construction

Correct By Construction

- Lots of current research focuses on correct by construction
- Fantastic for..

Correct By Construction

- Lots of current research focuses on correct by construction
- Fantastic for..
    - prototyping and/or building from the ground up

Correct By Construction

- Lots of current research focuses on correct by construction
- Fantastic for..
  - prototyping and/or building from the ground up
  - experts in formal methods and correct-by-construction techniques.

Correct By Construction

- Lots of current research focuses on correct by construction
- Fantastic for..
    - prototyping and/or building from the ground up
    - experts in formal methods and correct-by-construction techniques.
- What about everyone else?

Differential Privacy In The Wild

Differential Privacy In The Wild

- Apple

Differential Privacy In The Wild

- Apple
    - Device usage statistics (what apps are popular, which health metrics are most used, etc)

Differential Privacy In The Wild

- Apple
    - Device usage statistics (what apps are popular, which health metrics are most used, etc)
- Google (Chrome)

Differential Privacy In The Wild

- Apple
    - Device usage statistics (what apps are popular, which health metrics are most used, etc)
- Google (Chrome)
    - Browser usage statistics

Differential Privacy In The Wild

- Apple
    - Device usage statistics (what apps are popular, which health metrics are most used, etc)
- Google (Chrome)
    - Browser usage statistics
- Census

Differential Privacy In The Wild

- Apple
    - Device usage statistics (what apps are popular, which health metrics are most used, etc)
- Google (Chrome)
    - Browser usage statistics
- Census
    - 2020 Disclosure Avoidance

Differential Privacy In The Wild

- Apple
    - Device usage statistics (what apps are popular, which health metrics are most used, etc)
- Google (Chrome)
    - Browser usage statistics
- Census
    - 2020 Disclosure Avoidance
- Uber

Differential Privacy In The Wild

- Apple
    - Device usage statistics (what apps are popular, which health metrics are most used, etc)
- Google (Chrome)
    - Browser usage statistics
- Census
    - 2020 Disclosure Avoidance
- Uber
    - Trip data

Differential Privacy In The Wild

Differential Privacy In The Wild

- These are big multi-part systems

Differential Privacy In The Wild

- These are big multi-part systems
- Unlikely that entire systems would be built with formal techniques

Differential Privacy In The Wild

- These are big multi-part systems
- Unlikely that entire systems would be built with formal techniques
- How do we guarantee properties when presented with a large system built in many languages?

Differential Privacy In The Wild

Differential Privacy In The Wild

- Prioritize

Differential Privacy In The Wild

- Prioritize
- Divide and Conquer

Differential Privacy In The Wild

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system
    - For example: core differentially private mechanism

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system
    - For example: core differentially private mechanism
- Determine property that is most important

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system
    - For example: core differentially private mechanism
- Determine property that is most important
    - For example: mechanism is implemented according to some spec (e.g. paper's description of mechanism)

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system
    - For example: core differentially private mechanism
- Determine property that is most important
    - For example: mechanism is implemented according to some spec (e.g. paper’s description of mechanism)
- Use more adaptable techniques for ‘plumbing’

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system
    - For example: core differentially private mechanism
- Determine property that is most important
    - For example: mechanism is implemented according to some spec (e.g. paper’s description of mechanism)
- Use more adaptable techniques for ‘plumbing’
    - Control-flow analysis to ensure that all released data passes *through* the verified mechanism

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system
    - For example: core differentially private mechanism
- Determine property that is most important
    - For example: mechanism is implemented according to some spec (e.g. paper's description of mechanism)
- Use more adaptable techniques for 'plumbing'
    - Control-flow analysis to ensure that all released data passes *through* the verified mechanism
- Find high-level properties of the whole system

Differential Privacy In The Wild

- Determine aspects that are most crucial to the system
    - For example: core differentially private mechanism
- Determine property that is most important
    - For example: mechanism is implemented according to some spec (e.g. paper’s description of mechanism)
- Use more adaptable techniques for ‘plumbing’
    - Control-flow analysis to ensure that all released data passes *through* the verified mechanism
- Find high-level properties of the whole system
    - Use property-based testing to gain some assurance of those properties.

Example: Sensitivity

Example: Sensitivity

- Correctly accounting for the sensitivity of your system can be difficult.

Example: Sensitivity

- Correctly accounting for the sensitivity of your system can be difficult.
- Even with static guarantees about your software, there are meta-concerns:

Example: Sensitivity

- Correctly accounting for the sensitivity of your system can be difficult.
- Even with static guarantees about your software, there are meta-concerns:
    - Program crashes and you add code to avoid that situation (S. Garfinkel got me thinking about this)

Example: Sensitivity

- Correctly accounting for the sensitivity of your system can be difficult.
- Even with static guarantees about your software, there are meta-concerns:
    - Program crashes and you add code to avoid that situation (S. Garfinkel got me thinking about this)
        - You may have encoded data-dependent information in your control-flow!

Example: Sensitivity

- Correctly accounting for the sensitivity of your system can be difficult.
- Even with static guarantees about your software, there are meta-concerns:
  - Program crashes and you add code to avoid that situation (S. Garfinkel got me thinking about this)
    - You may have encoded data-dependent information in your control-flow!
  - Re-running algorithms for optimizations

Example: Sensitivity

- Correctly accounting for the sensitivity of your system can be difficult.
- Even with static guarantees about your software, there are meta-concerns:
  - Program crashes and you add code to avoid that situation (S. Garfinkel got me thinking about this)
    - You may have encoded data-dependent information in your control-flow!
  - Re-running algorithms for optimizations
    - Are the optimizations data-dependent?

Conclusions

Conclusions

- Verification techniques for Differential Privacy are powerful and diverse.

Conclusions

- Verification techniques for Differential Privacy are powerful and diverse.
- Still work to be done on ‘whole system’ approaches

Conclusions

- Verification techniques for Differential Privacy are powerful and diverse.
- Still work to be done on ‘whole system’ approaches
- We can learn from how other large systems achieve high-assurance