Privacy-Preserving Distributed Information Sharing

Lea Kissner
leak@cs.cmu.edu

Advisor: Dawn Song
dawnsong@cmu.edu

Why Share?

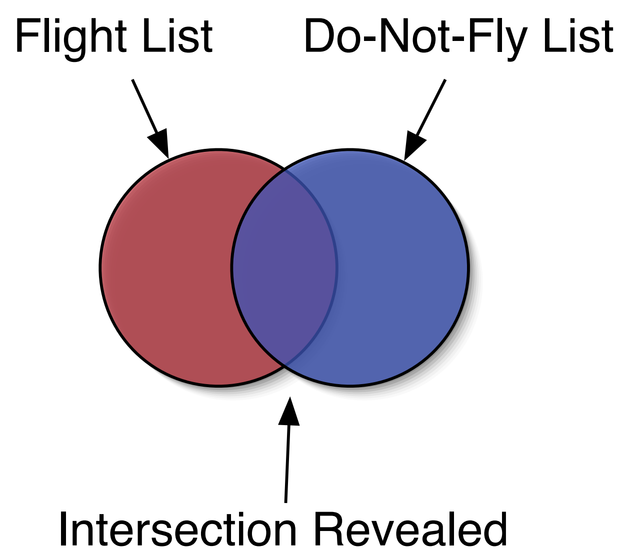
- Many applications require mutually distrustful parties to share information
- Many examples in two major categories
  - *Statistics-gathering*. Determining the number of cancer patients on welfare, distributed network monitoring
  - *Security enforcement.* Enforcing the `do-not-fly' list, catching people who fill prescriptions twice

Why Privacy?

- There are complex laws and customs surrounding the use of many kinds of information
  - HIPPA for health information in the U.S.
  - Broad laws in Canada and Europe
  - Customers may avoid companies who compromise data
- Thus, privacy is an important concern in sharing many types of information

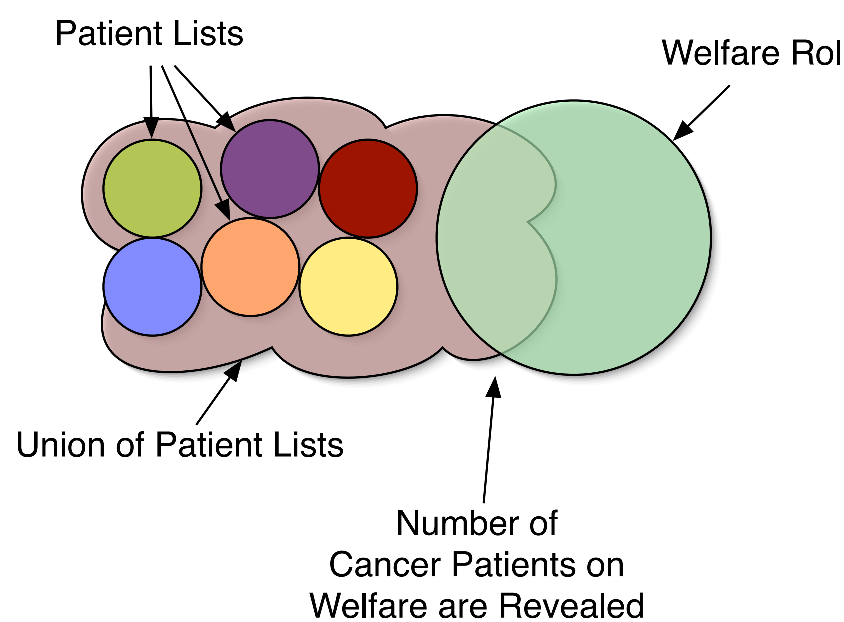
Applications

- Do-not-fly list
  - Airlines must determine which passengers cannot fly
  - Government and airlines cannot disclose their lists



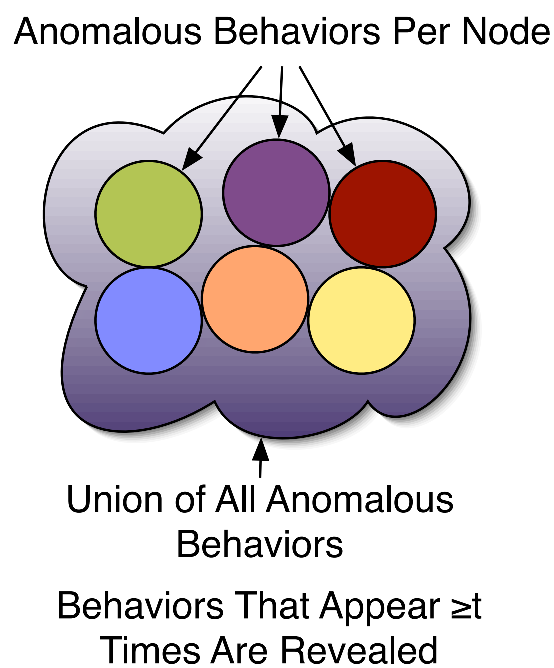
Applications

- Public welfare survey: number of welfare recipients who have cancer
  - Each list of cancer patients is confidential
  - Welfare rolls are confidential
  - To reveal the number of welfare recipients who have cancer, must compute private union and intersection operations



Applications

- Distributed network monitoring
  - Nodes in a network identify anomalous behaviors
  - If a possible attack only appears a few times, it is probably a false positive, and should be filtered out
  - The nodes must privately compute the element reduction and union operations
  - If an element *a* appears t times in S, *a* appears t-1 times in the reduction of S



Current Solutions

- There are some protocols for privacy-preserving information sharing, but:
  - Most applications use a trusted third party (TTP)
  - Some applications are foregone entirely
- A TTP can become a security problem:
  - Betrayal of trust
  - Social engineering
  - Attractive target for attacks

Thesis

- Is it possible to construct protocols for privacy-preserving distributed information sharing such that:
  - eliminate the TTP
  - efficient protocols on large bodies of data
  - applicable to many practical situations

Outline

- Motivation
- Thesis
- *Completed Work*
  - *Privacy-Preserving Set Operations*
  - Privacy-Preserving Hot Item Identification
- Proposed Work
- Timeline
- Conclusion

Set Operations

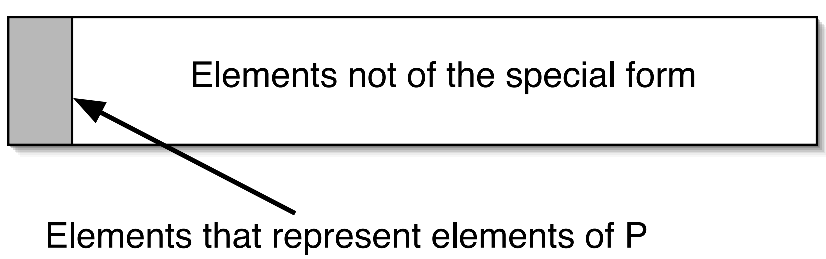
- Each player has a private input multiset
- Composable, efficient, secure techniques for calculating multiset operations:
  - Union
  - Intersection
  - Element reduction (each element *a* that appears *b>0* times in *S*, appears *b-1* times in *Rd(S)*)

Set Operations

- We apply these efficient, secure techniques to a wide variety of practical problems:
  - Multiset intersection
  - Cardinality of multiset intersection
  - Over-threshold set-union
  - Variations on threshold set-union
  - Determining subset relations
  - Computing CNF boolean formulas

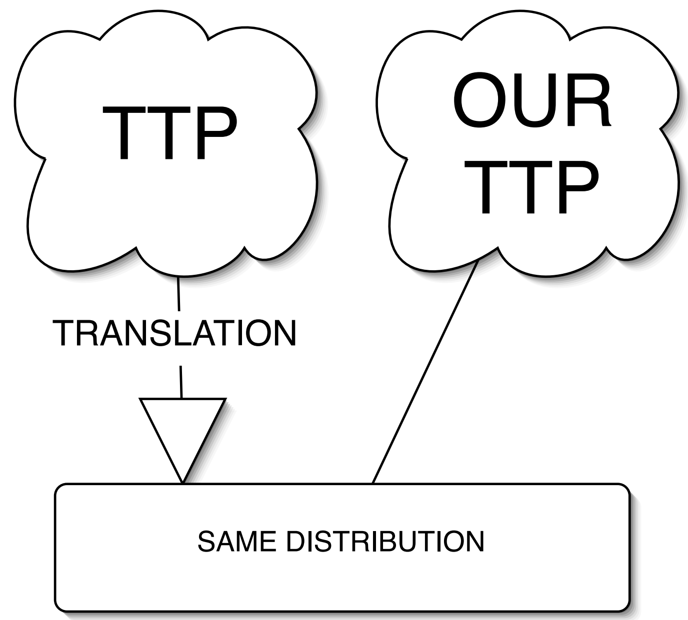
Polynomial Rep.

- To represent the multiset S as a polynomial with coefficients from a ring R, compute $\prod_{a \in S} (x - a)$
- The elements of the set represented by *f* is the **roots of *f* of a certain form** $y \,||\, h(y)$
  - Random elements are not of this form (with overwhelming probability)
  - Let elements of this form *represent elements of P*



Security

- We design our techniques for set operations on polynomials to hide all information but the result
- Formally, we define security (privacy-preservation) for the **techniques** we present as follows:
  - The output of a trusted third party (TTP) can be transformed in probabilistic polynomial time to be identically distributed to a TTP using our techniques



Security

- A *uniformly distributed* polynomial is one with each coefficient chosen uniformly at random
- If A is the multiset result of an operation, the polynomial representation calculated by our techniques is of the following form:

$$\left(\prod_{a \in A} (x - a) \right) * u$$

  - where u is a uniformly distributed polynomial (length depends on previous operations, size of operands)

Techniques

- Let S, T be multisets represented by the polynomials *f, g*. Let *r, s* be uniformly distributed polynomials.
- Union -- S∪T is calculated as ***f*g***
- Intersection -- S∩T is calculated as ***f*r+g*s***
  - Poly. addition preserves shared roots of *f, g*
  - Use of random polynomials ensures correctness and masks other information about S, T
  - The operation can be extended to ≥3 multisets

Techniques

- Standard result: if $f(a)=0$,
  $f^{(d)}(a)=0 \Leftrightarrow (x-a)^{d+1} \mid f$
- Let S be a multiset represented by the polynomial *f*. Let *r*, *s* be uniformly distributed polynomials, and *F* a random public polynomial of degree *d*.
- Element reduction -- $\mathrm{Rd}_d(S)$ is calculated as
  $\boldsymbol{f^{(d)} * F * r + f * s}$
  - According to standard result, desired result is obtained by calculating intersection of $f, f^{(d)}$

Without TTP

- We now give techniques to allow use of our operations in real-world protocols
- Encrypt coefficients of polynomial using a threshold additively homomorphic cryptosystem
- We can perform the calculations needed for our techniques with encrypted polynomials (examples use Paillier cryptosystem)
  - Addition

$$\begin{aligned} h &= f + g \\ h_i &= f_i + g_i \\ E(h_i) &= E(f_i) * E(g_i) \end{aligned}$$

Without TTP

- We can perform the calculations needed for our techniques with encrypted polynomials
  - Formal derivative
  -
  -
  - Multiplication

$$
\begin{aligned}
h &= f' \\
h_i &= (i+1) f_{i+1} \\
E(h_i) &= E(f_i)^{i+1}
\end{aligned}
$$

$$
\begin{aligned}
h &= f * g \\
h_i &= \sum_{j=0}^{k} f_j * g_{i-j} \\
E(h_i) &= \prod_{j=0}^{k} E(f_j)^{g_{i-j}}
\end{aligned}
$$

Multiset Intersection

- Let each player i ($1 \leq i \leq n$) hold an input multiset S_i
- Each player calculates the polynomial f_i representing their private input set and broadcasts $E(f_i)$
- For each i, each player j ($1 \leq j \leq n$) chooses a uniformly distributed polynomial $r_{i,j}$, and broadcasts $E(f_i * r_{i,j})$
- All players calculate and decrypt $E\left(\sum_{i=1}^{n} f_i * \left(\sum_{j=1}^{n} r_{i,j}\right)\right) = E(p)$
- Players determine the intersection multiset: if $(x-a)^b \mid p$ then *a* appears b times in the result

General Functions

- Using our techniques, efficient protocols can be constructed for any function described by (let s be a privately held set):
  - γ ::= s | $Rd_d(\gamma)$ | $\gamma \cap \gamma$ | $s \cup \gamma$ | $\gamma \cup s$
- To compute the operator $A \cup B$, where E(f), E(g) are encrypted polynomial representations of A, B
  - Players additively share g; each player holds g_i
  - Each player computes $E(f*g_i)$, and all players compute $E(f*g_1 + ... + f*g_n) = E(f*g)$

Outline

- Motivation
- Thesis
- *Completed Work*
  - Privacy-Preserving Set Operations
  - *Privacy-Preserving Hot Item Identification*
- Proposed Work
- Timeline
- Conclusion

Hot Item Identification

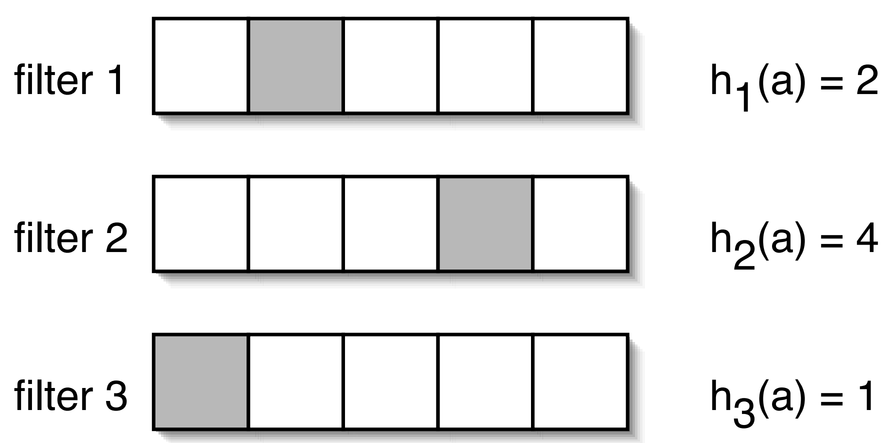
- *Hot Item ID* is the problem of identifying items that appear often in players' private input sets
- Can be addressed by our privacy-preserving set operation techniques
- Requires greater efficiency and flexibility, in many applications
  - Distributed network monitoring
  - Distributed computer troubleshooting

Hot Item Identification

- We give protocols that:
  - use comparable bandwidth to non privacy-preserving protocols
  - use only lightweight, efficient cryptography
  - players can join and leave at any time
  - very robust for ALL connected players
  - use tailored security definitions

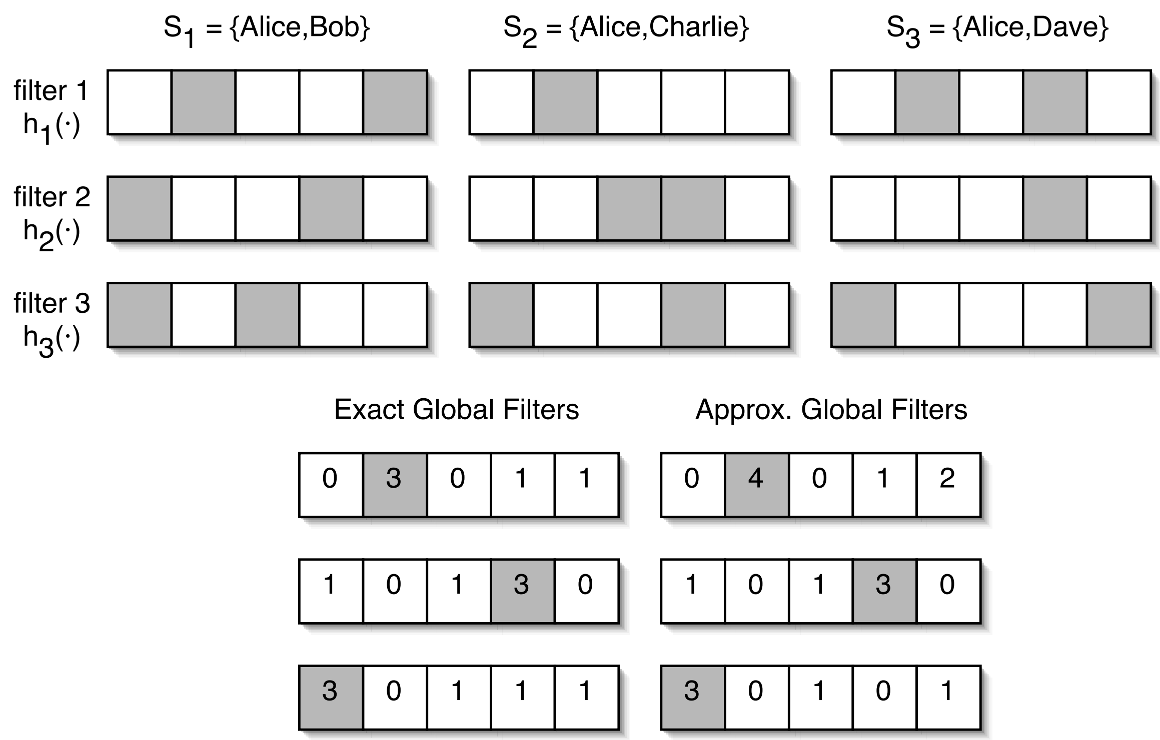
Approx. Filters

- We utilize a strategy of approximate collaborative filtering
  - Each player constructs a set of local filters to represent his private input set
  - For each element *a*, for filter $1 \leq i \leq T$, mark bucket $h_i(a)$ as `hit'



Global Filters

- Each bucket hit by at least t people is marked as `hot'
- An item a is hot if $\forall_{i \in [T]}\, h_i(a)$ is hot



Approx. Counting

- The players construct global filters
  - For each bucket of each filter, the players determine whether at least *t* players hit it
- Exact counting is expensive, so we utilize an approximate counting scheme
- We will count the number of distinct uniformly distributed elements
  - Each player can produce exactly one uniformly distributed element per bucket
  - These *One-Show Tags* can be constructed using a modified group signature scheme

Approx. Counting

- If the *k*th smallest uniform element in *S* is $\alpha \in (0,1]$, then we estimate that $|S|=k/\alpha$
  - $\geq t$ elements iff there are $\geq k$ items s.t. $\alpha \leq k/t$
- Thus, for each bucket in each filter, the players try to collect these *k* items
  - Broadcast eligible tags to neighbors
  - Forward tags until have sent *k* or converges
    - Valid
    - Small (tag value is $\leq k/t$)

Outline

- Motivation
- Thesis
- Completed Work
- *Proposed Work*
    - *Overview*
    - *Secure Cryptographic Substitution Framework*
- Timeline
- Conclusion

Proposed Work

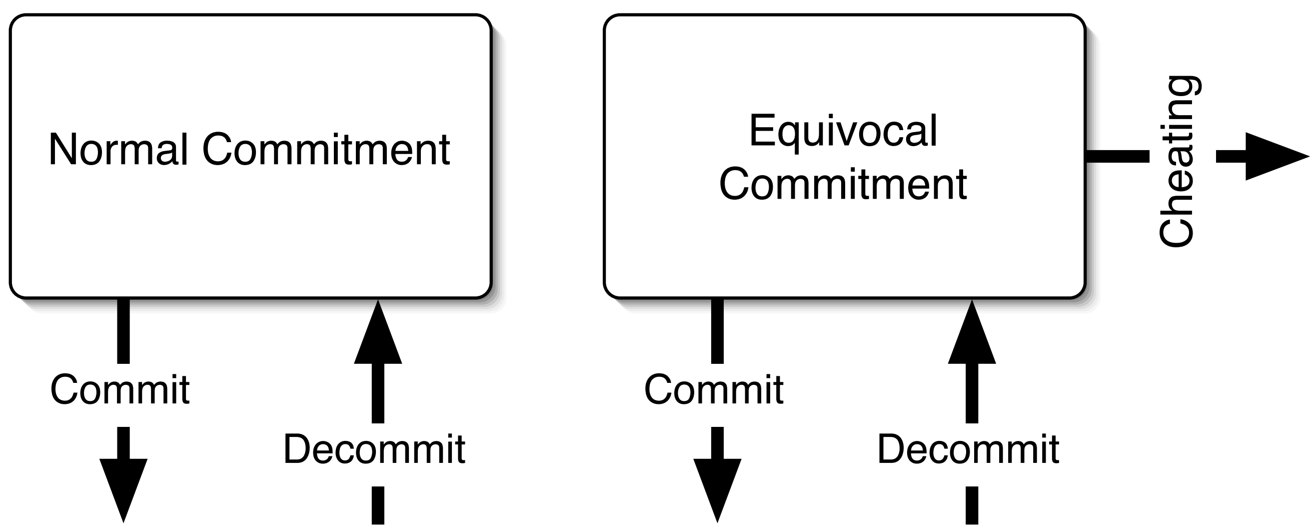
- We wish to explore at least one problem in the following areas, relating to privacy-preserving distributed information sharing:
  - Improved efficiency
  - Extending scope -- there are not efficient protocols for many situations
    - all of our protocols, and most related work, compute on sets or multisets
    - there are interesting opportunities in other structures, such as graphs, junction trees, etc.

Tool Substitution

- Many protocols secure against malicious adversaries are inefficient
- We believe that use of more efficient tools can make many protocols more efficient
- Examples:
  - Equivocal, chameleon, ... commitments (as used in our set operation protocols)
  - *no-key boxes* (undecrypted ciphertexts)
- We wish to allow secure substitution of expensive tools for more efficient ones

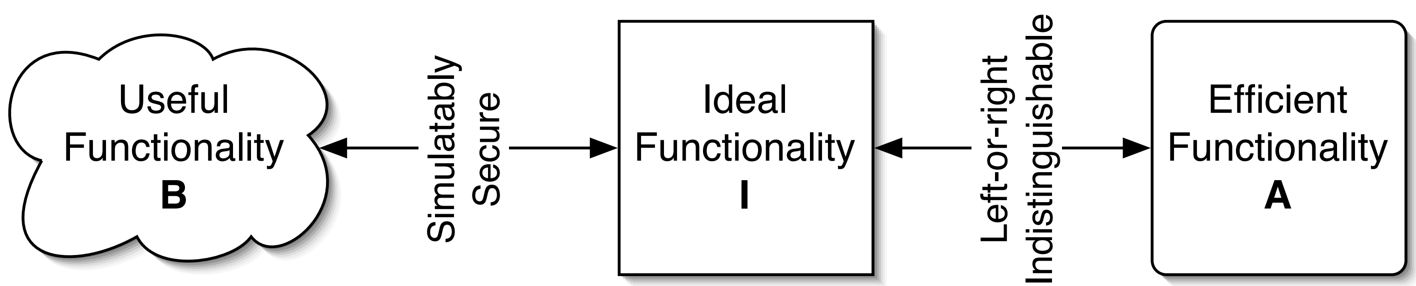
Tool Substitution

- Main idea: any pair of tools that are *interface indistinguishable* can be substituted in almost all protocols secure against malicious parties, even when these substituted tools are composed



Tool Substitution

- A tool is interface indistinguishable if it `acts like' the ideal functionality
- We have multiple ways of proving this -- intuitively, they all show security
- We say A is a *workalike* of B if
  - B is secure with respect to ideal functionality I
  - A is left-or-right indistinguishable from I

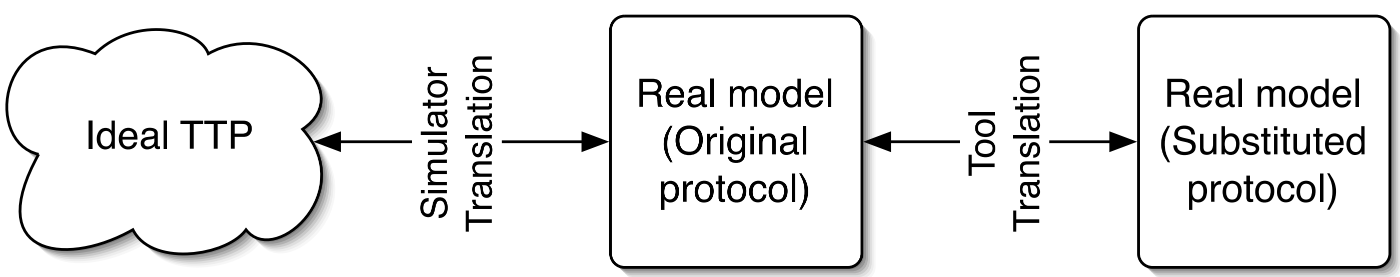


Tool Substitution

- A *handle* is any input/output data that differs between workalikes A and B (commitments, ciphertexts)
- Theorem: we can securely substitute tool A for tool B if
  - A is a workalike of B
  - The protocol does not require any player to send a non-identity function of a handle

Tool Substitution

- Proof by non-uniform reduction
- The tool translator mediates communication between parties using the original tool and the substituted tool
- This translator often must be non-uniform
- Use of the translator gives a simulation proof



Tool Substitution

- Future work
  - Attempt proof in standard model
  - Complete formalization of proofs
    - Non-uniform
    - Non-black-box
    - Possibly standard or other models

Outline

- Motivation
- Thesis
- Completed Work
- Proposed Work
- Related Work
- *Timeline*
- *Conclusion*

Timeline

- *Sept. 2005* -- Complete proofs for tool substitution
- *Nov. 2005* -- Formalize proofs for tool substitution
- *Dec. 2005* -- Begin exploration of other problems
- *May 2006* -- Begin writing thesis draft
- *July 2006* -- Draft thesis completed
- *Aug. 2006* -- Thesis defense

Conclusion

- In my thesis, I will address efficient and secure protocols for privacy-preserving distributed information sharing
  - Privacy-preserving multiset operations
  - Hot item identification and publication
  - Secure cryptographic tool substitution
- These protocols and techniques allow practical and secure use of many important applications.

Thank You!