Privacy, Integrity, and Incentive Compatibility in Computations with Untrusted Parties

Sheng Zhong
Yale University

Dissertation Director: Joan Feigenbaum

Committee Members: James Aspnes,

Markus Jakobsson (RSA Labs),

Yang Richard Yang.

Thesis Statement

"Privacy, integrity, and incentive compatibility, when properly formulated, can often be achieved in new distributed-computing scenarios."

— Supported by studies of efficient mix, secure storage on untrusted servers, privacy-preserving mining of association rules, secure mobile-agent computation, and security in ad hoc networks.

— Privacy and integrity are party of the traditional study of secure multiparty computation, but incentive compatibility is a relatively new consideration.

Summary of Major Work: Privacy, Integrity, and Incentive Compatibility

Component of Thesis Work	Privacy	Integrity	Incentive compatibility
Efficient Mix ([GZ+02], *ASIACRYPT'02*)	✓	✓	
Secure Storage on Untrusted Servers ([AFYZ04], *ESORICS'04*)		✓	
Privacy-Preserving Data Mining ([Z04])	✓		
Security of Mobile Agents ([ZY03], *DIALM-POMC'03*)	✓	✓	
Security in Mobile Ad hoc Networks ([ZCY03], *INFOCOM'03*)		✓	✓

Outline of Talk

- Quick Summary of Frequently Used Techniques

(5 Components of Thesis:)

- Efficient Mix
- Secure Storage on Untrusted Servers
- Privacy-Preserving Mining for Association Rules
- Security of Mobile Agents
- Security in Mobile Ad Hoc Networks

Summary of Frequently Used Techniques

- Homomorphic Encryption (especially ElGamal Encryption — See next slide)
- (A Variant of) Selective Disclosure [AIR01]
- Feldman's Verifiable Secret Sharing [Fel87]
- Desmedt-Frankel Threshold Decryption [DF89]

ElGamal Encryption

- Probabilistic encryption of message m (in a group where discrete log is hard):

$$C = (M, G) = (my^r, g^r),$$

where g is a generator, r is a random exponent, and $y=g^x$ is the public key.

- Decrypting a ciphertext "requires" knowledge of private key x:

$$m = M / G^x.$$

ElGamal Encryption (Cont’d)

- Without knowledge of private key, one can reencrypt (rerandomize) a ciphertext — compute another ciphertext having the same cleartext:

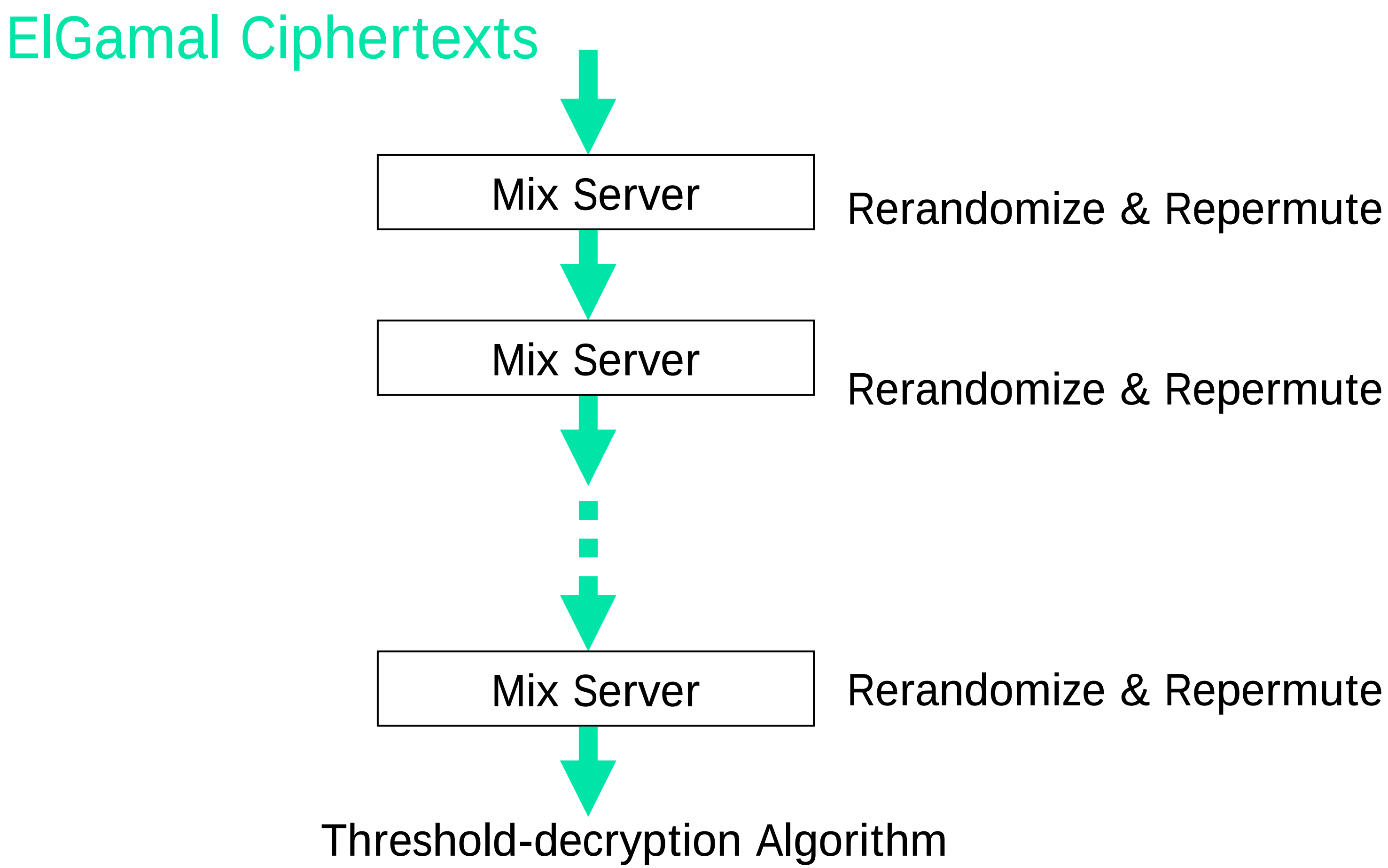
$$(M', G') = (My^s, Gg^s)$$

- (M’,G’) is called an reencryption (rerandomization) of (M,G).

Component 1: Efficient Mix [GZ+02]

- A mix network (consisting of a group of mix servers) is a construction for anonymizing communications.
- Security requirements:
  - Privacy: Infeasible to associate any input with the corresponding output.
  - Verifiability: Can ensure that outputs are a permutation of the decryptions /reencryptions of inputs.

Global Picture: ElGamal-based Decryption Mix



Proof of Product with Checksum

- Question: How do we ensure that each server rerandomizes and repermutes messages correctly?
- Answer: Let the server prove

  Product of Inputs = Product of Outputs

  - This is easy, because ElGamal is multiplicatively homomorphic.
  - With an additional checksum, if any messages were corrupted, cheating would be detected.

Double Encryption

- Observation: If cheating is detected because of an invalid checksum, then detection is after decryption.

$\Rightarrow$ Problem: Privacy can be violated before cheating is detected.

- Solution: Additional layer of encryption.
  - Cheating is detected after outer-layer decryption but still before inner-layer decryption.

Analysis

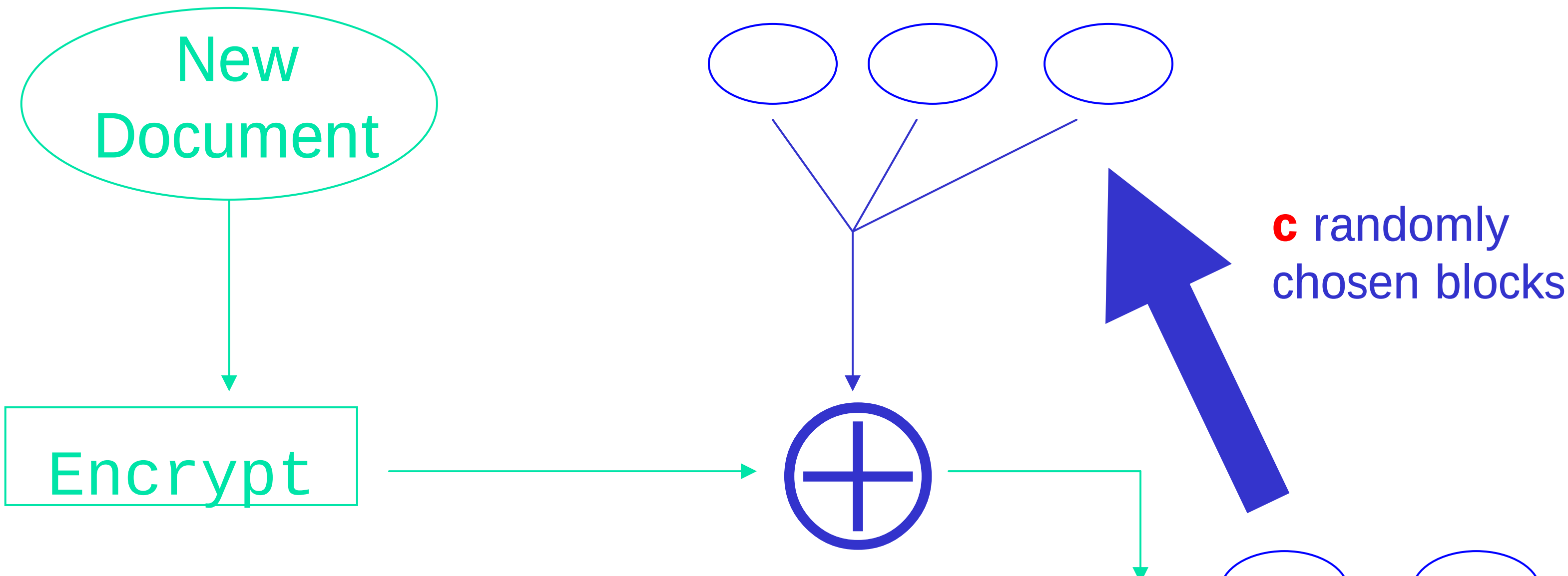
- Efficiency: In normal cases (no cheating), our mix is highly efficient. It is the only mix in which reencryption & decryption (not proofs) are the major overhead.
- Privacy: With proper proofs of knowledge of inputs, our mix net achieves privacy similar to standard ElGamal-based mix nets.
- Public Verifiability: The operations of our mix net on the *well-formed* messages can be verified.

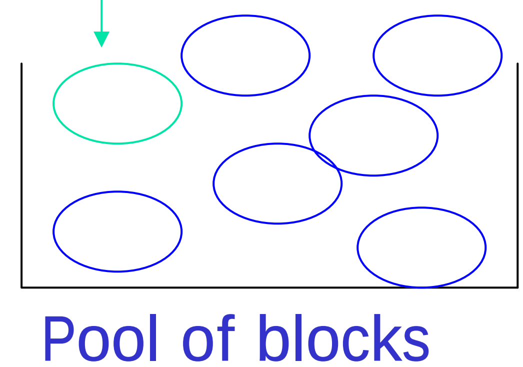
Component 2: Secure Storage with Untrusted Server [AFYZ04]

- Question: Suppose you store your data on a remote server. How do you ensure that it is not corrupted by the server?
- Answer: Have your data entangled with some VIPs' such that

corruption of your data $\Rightarrow$ corruption of theirs.

Previous Work: Dagster

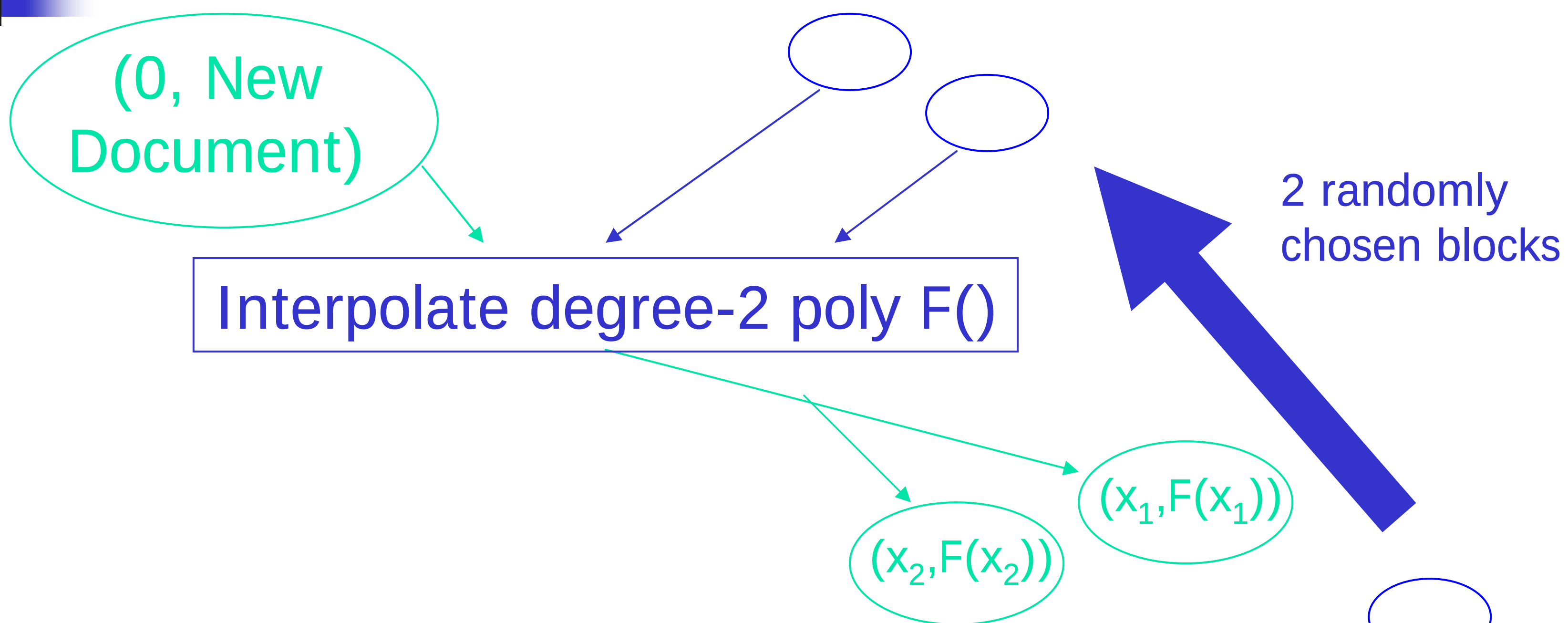




Analysis:

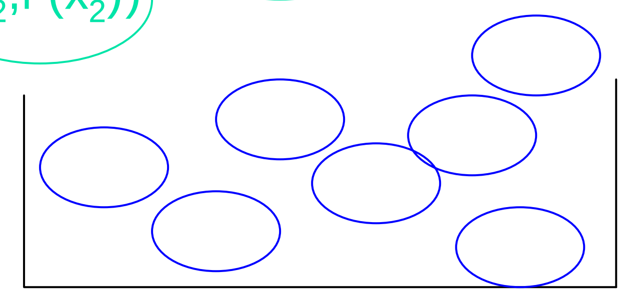
Deleting a typical document
$\Rightarrow$ loss of O(c) documents

Previous Work: Tangler



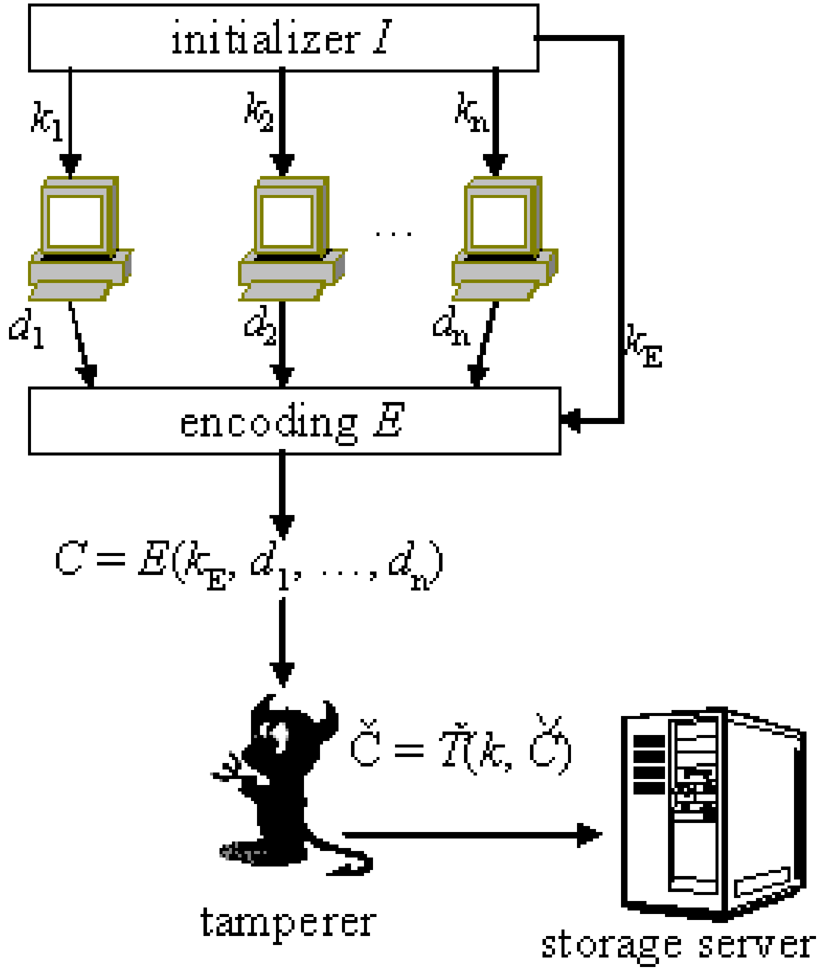
Analysis:

Deleting a typical document ⇒ loss of O(logn/n) documents



Pool of n blocks

Our Model: Basic Framework



Our Model: Classification

- Classification based on recovery algorithm:
  - All users use a standard-recovery algorithm provided by the system designer.
  - All users use a public-recovery algorithm provided by the adversary.
  - Each individual uses a private-recovery algorithm provided by the adversary.
- Classification based on corrupting algorithm:
  - Destructive adversary that reduces the entropy of the data store
  - Arbitrary adversary

Our Definitions

- Data dependency: d_i depends on d_j if with high probability

  d_i is recovered $\Rightarrow$ d_j is recovered.

- All-or-Nothing Integrity (AONI): Every document depends on every other document.

Possibility of AONI in Standard-Recovery Model

- When combining data, mark data store using an unforgeable Message Authentication Code (MAC).
- Standard-recovery algorithm checks MAC:
  - If MAC is valid, recover data.
  - If MAC is invalid, refuse to recover data.

Impossibility of AONI in Public- and Private-Recovery Models

- Recovery algorithm can flip a coin to decide whether to recover data or not.
- With high probability, not all coin flips will have same result.

⇒ With high probability, some data are recovered while others are not.

⇒ Cannot guarantee AONI.

Possibility of AONI for Destructive Adversaries

- When combining data, interpolate a polynomial using points (key, data item).
- Store = polynomial.
- AONI is achieved if sufficient entropy is removed.
  - Many stores are mapped to single corrupted store.

  $\Rightarrow$ With high probability, no data item can be recovered.

Component 3: Privacy-Preserving Mining for Association Rules [Z04]

Trans#	Bread	Milk	Egg	Apple	Cereal
1001	✓	✓	✓		✓
1002		✓		✓	✓
1003	✓	✓			✓
1004		✓	✓	✓	✓

- Association Rule: Milk $\Rightarrow$ Cereal.
  - {Milk, Cereal} is frequent (i.e., #{Milk, Cereal} is large).
  - #{Milk, Cereal}/#{Milk} is close to 1.
- The key technical problem in association-rule mining is to find frequent itemsets.

Privacy in Distributed Mining

- Distributed Mining:
  - Two (or more) miners.
  - Each miner holds a portion of a database.
  - Goal: Jointly mine the entire database.
- Privacy: Each miner learns nothing about others' data, except the output.

Vertical Partition: Weakly Privacy-Preserving Algorithm

- Vertical Partition — Each miner holds a subset of the columns.
- Algorithm provides weak privacy — only *support count* (# of appearances of candidate itemset) is revealed.
- Computational Overhead: Linear in # of transactions.
  - Previous solution has a quadratic overhead.

Vertical Partition: Strongly Privacy-Preserving Algorithm

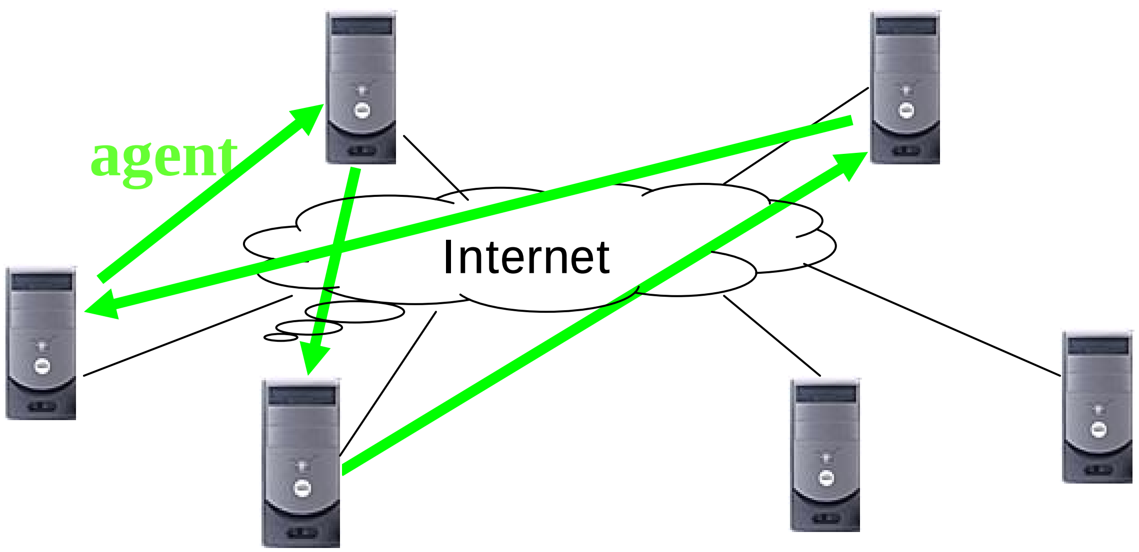
- Algorithm provides strong privacy — no information (except the output) is revealed.
- Computational Overhead: Also linear in # of transactions.
  - Slightly more expensive than weakly privacy-preserving algorithm.

Horizontal Partition

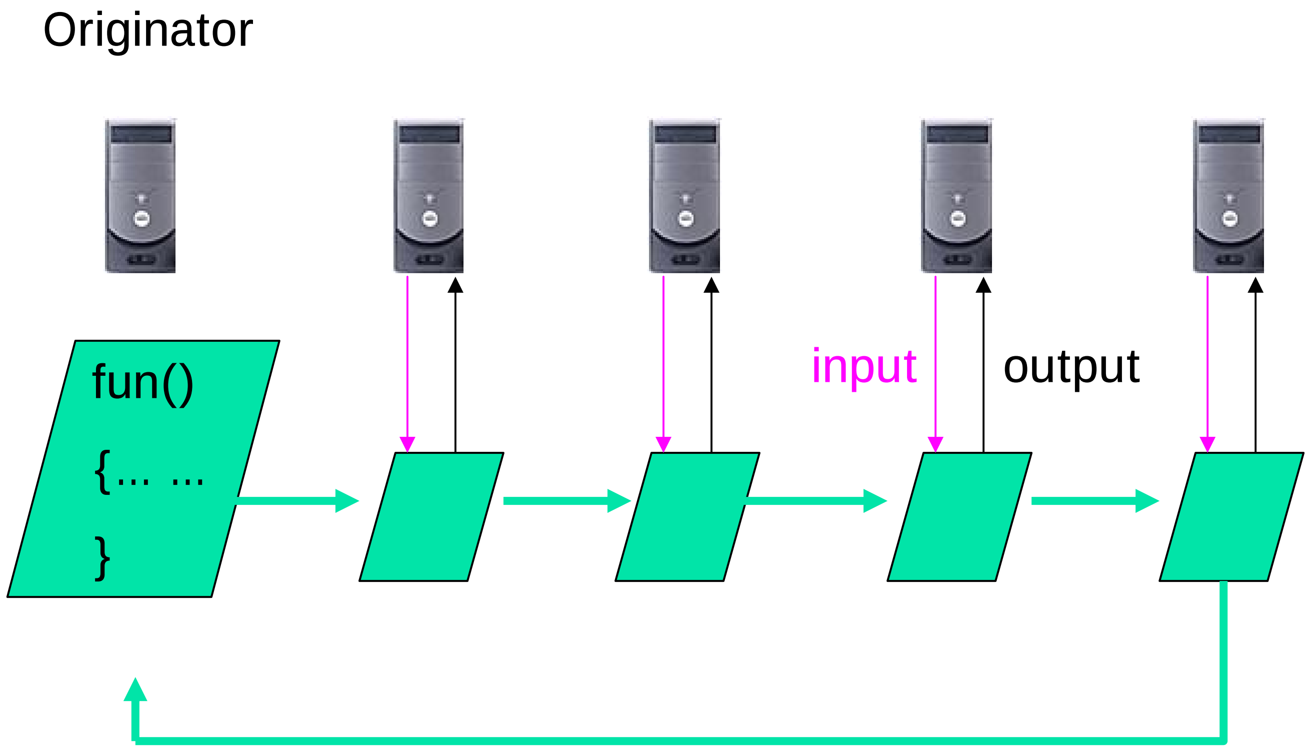
- Horizontal Partition — Each miner holds a subset of rows.
- Computational Overhead: Still linear in # of transactions.
- Works for two or more parties.
  - Previous solution only works for three or more parties.

Component 4: Secure Mobile-Agent Computation [ZY03]

- Mobile Agent: a piece of software moving around the network, performing a specific task
- Example: an agent searching for airline tickets



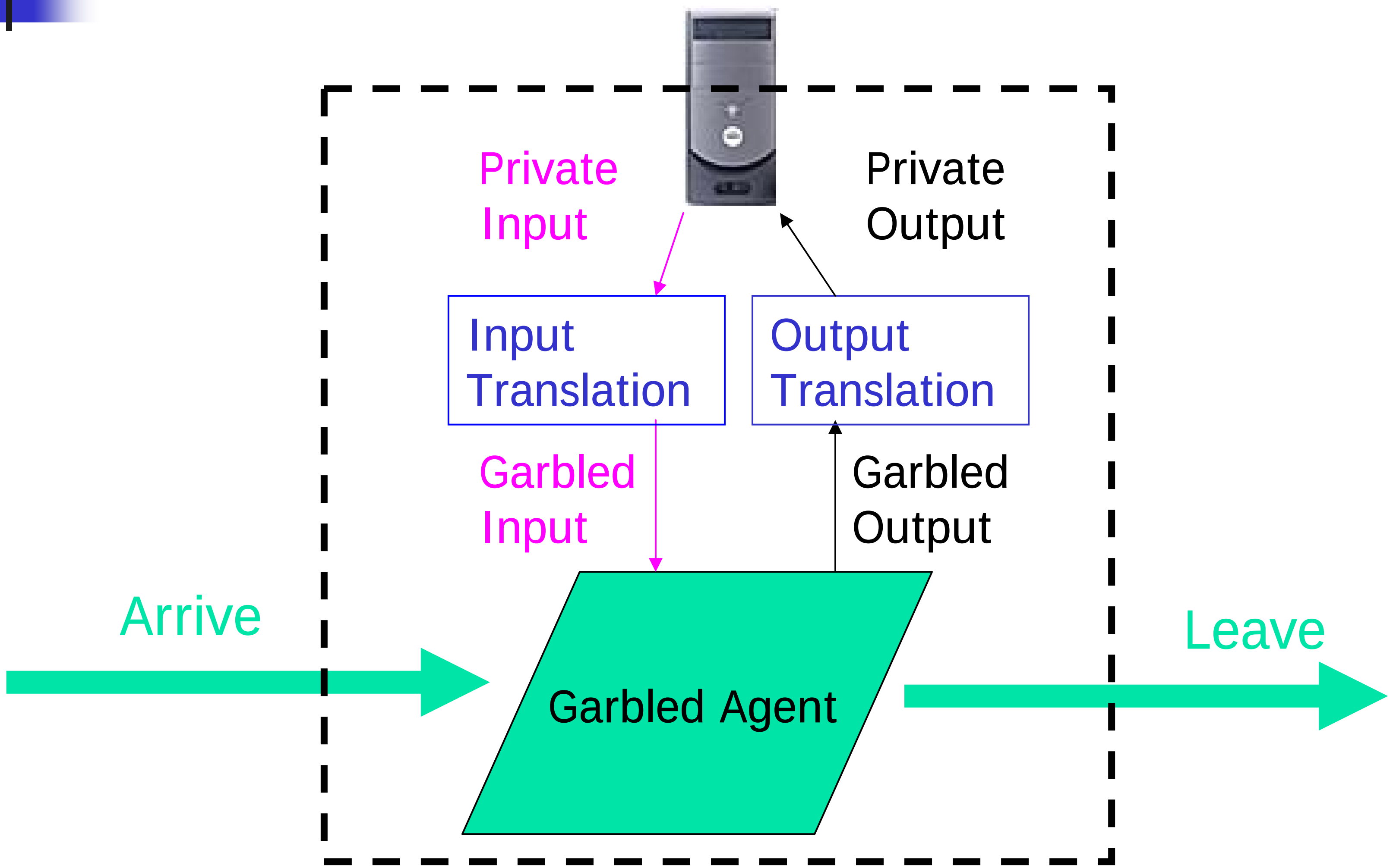
Problem Formulation (Cont'd)



Security Requirements

- Agent Originator's Privacy: Originator's private information (*e.g.*, a *buy-it-now* price in airline-ticket-agent example), even if stored in the agent, is not revealed to hosts.
- Host's Privacy: Each host's private input (*e.g.*, the ask price) and output (*e.g.*, whether to make a reservation) to the agent is not revealed to other hosts or to the originator.

Solution Framework [ACCK01]



Need for a Crypto Primitive

- Question: How to enable each host to translate I/O?
  - Output: Easy — Agent supplies translation table to host.
  - Input: Tricky — Must guarantee that only one value of input is translated. Don't want host to "test" the agent with many possible inputs.

Verifiable Distributed Oblivious Transfer (VDOT)

- Introduce a group of proxy servers.
- For each input bit: proxy servers hold garbled input for 0/1: G(0)/G(1).
  - Input bit = b $\rightarrow$ transfer G(b) to host.
  - No information about G(1-b) is revealed to host.
  - No information about b is revealed to proxy servers.
  - Proxy servers cannot cheat host with incorrect G(b).

Analysis of VDOT Security Requirements

- Input bit = b → transfer G(b) to host
- No information about G(1-b) is revealed to host
- No information about b is revealed to proxy servers

→ 1-out-of-2 Oblivious Transfer (OT)

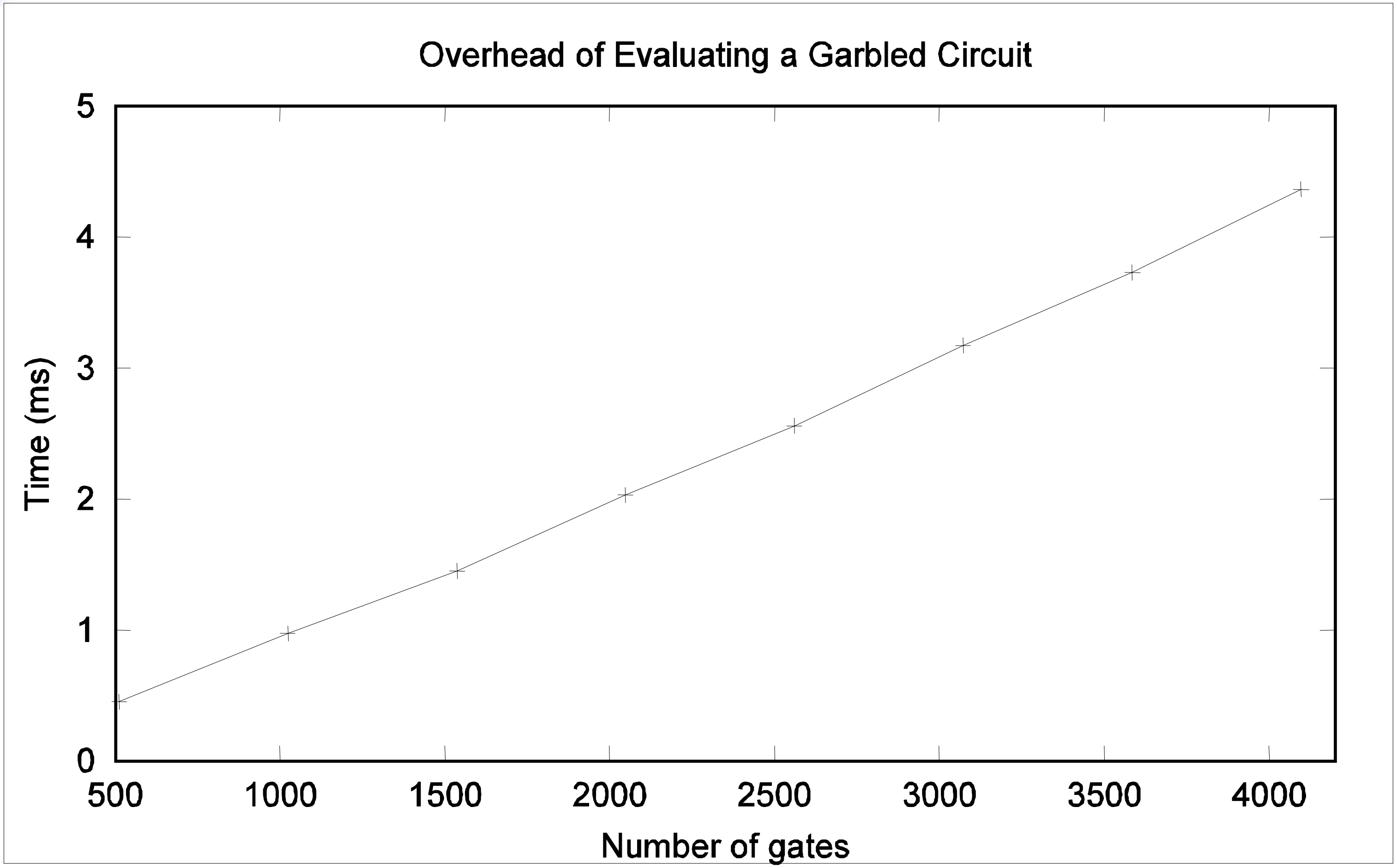
- Proxy servers can't cheat host with incorrect G(b)

→ Detection of Cheating

VDOT Design

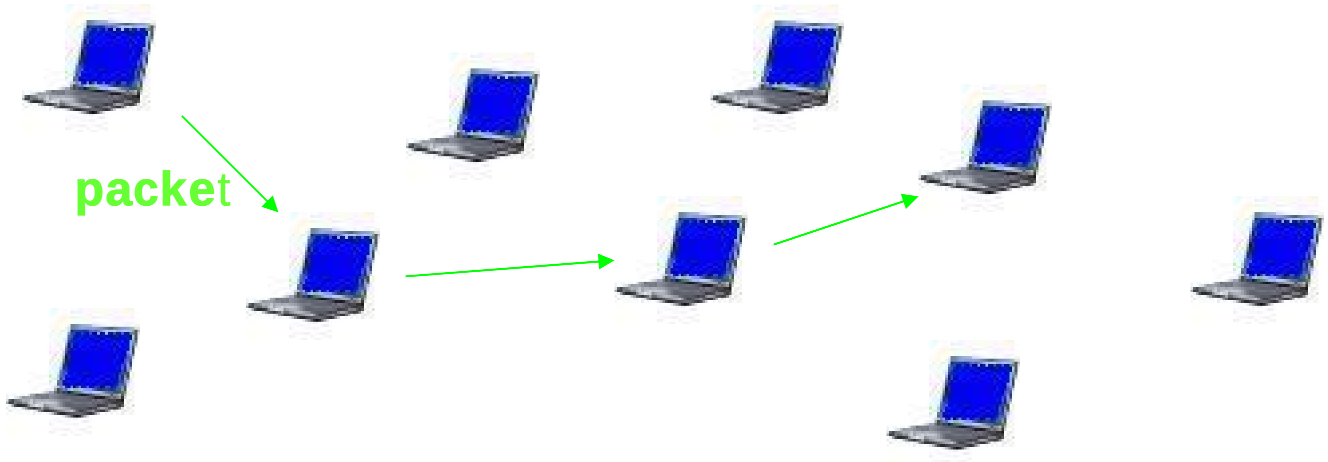
- Choose a distributed variant of *Bellare-Micali OT* [BM89] as basis of design.
- Add detection of cheating by employing the special algebraic structure of keys in Feldman VSS [Fel87].

Performance: Overhead of Garbled Circuits

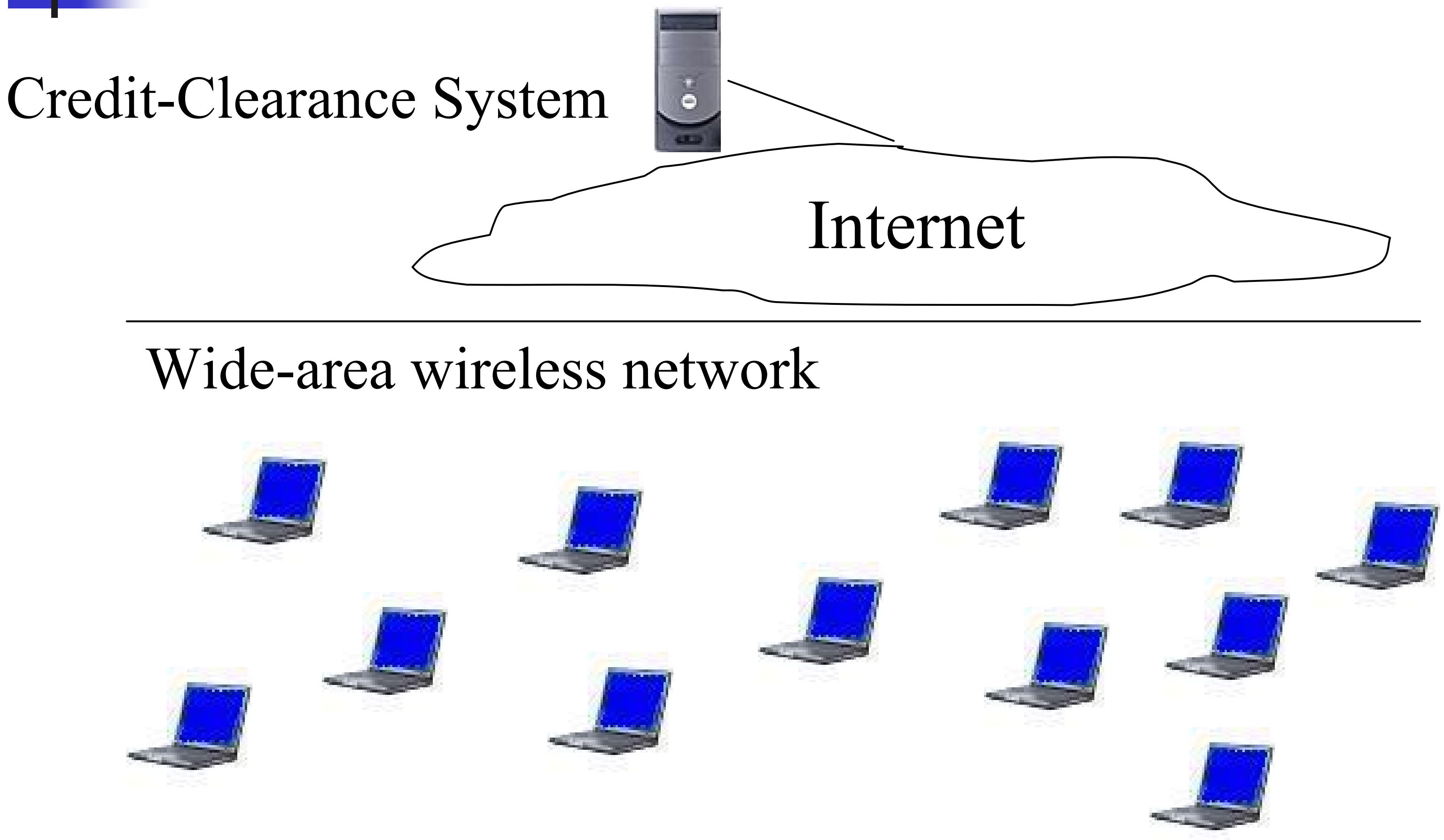


Component 5: Mobile Ad Hoc Network [ZCY03]

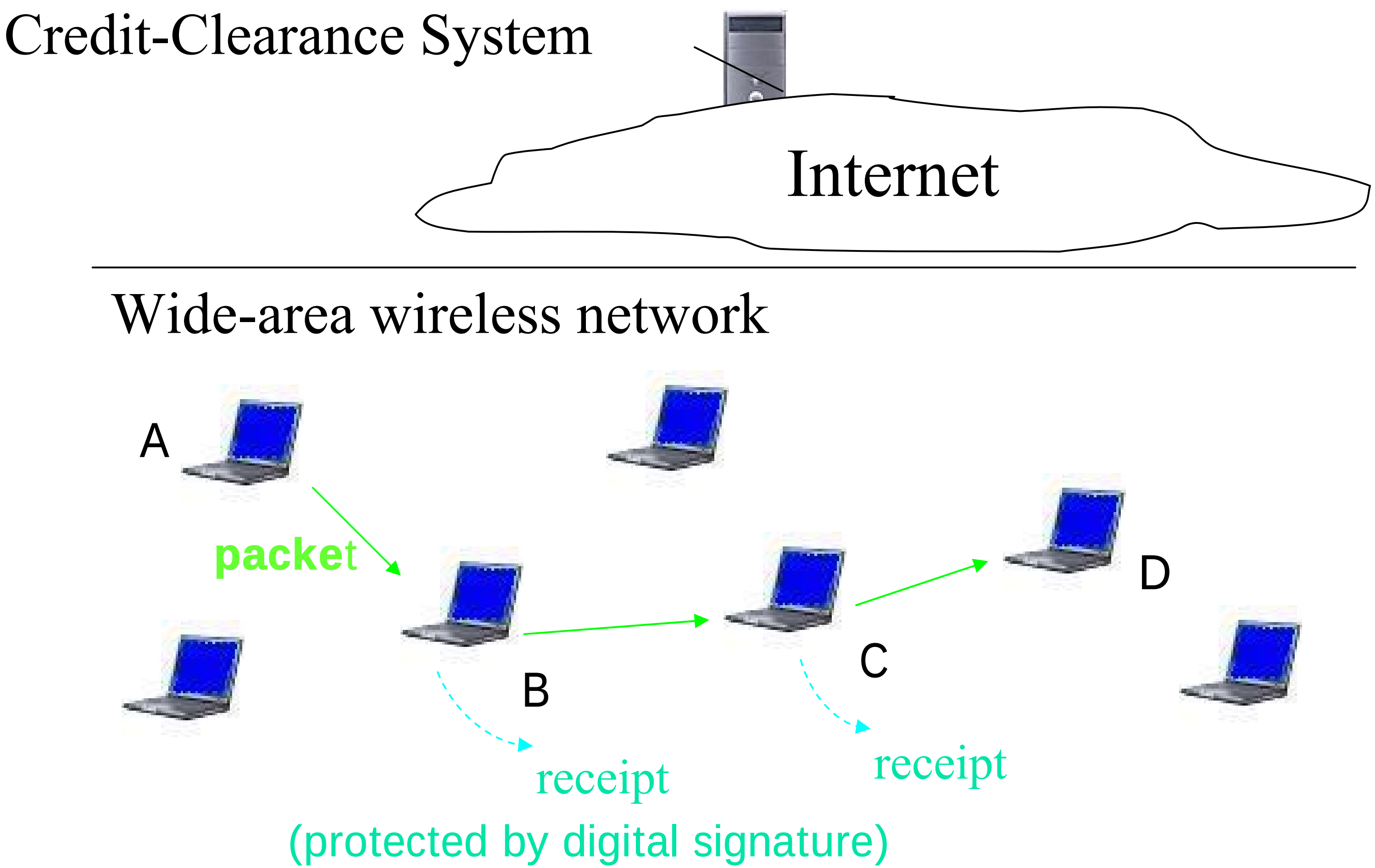
- Wireless multi-hop networks are formed by mobile nodes, with no pre-existing infrastructure.
- Nodes depend on other nodes to relay packets.
- A node may have no incentive to forward others' packets.



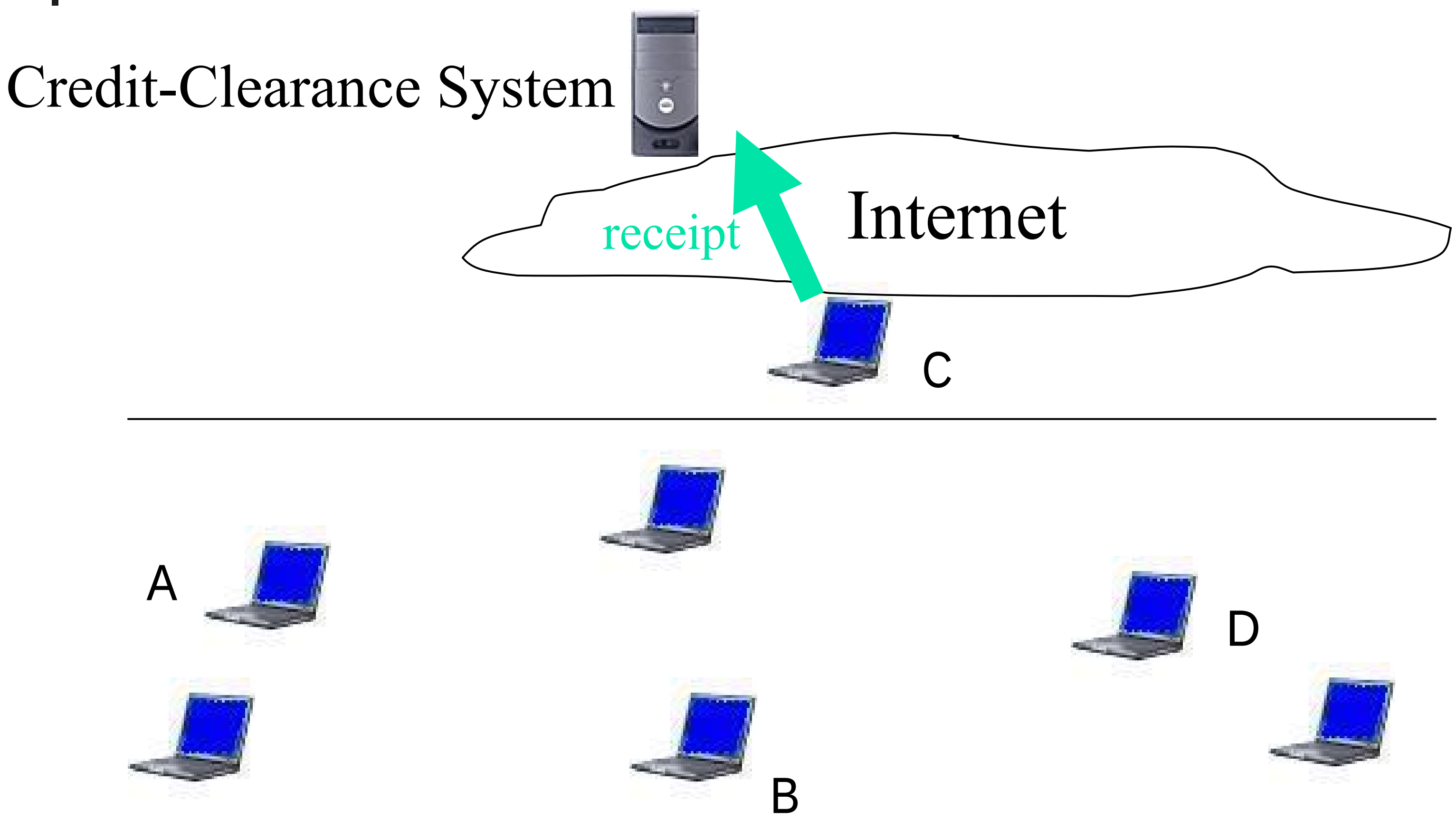
Sprite: System Architecture



Big Picture: Saving Receipts



Big Picture: Getting Payment



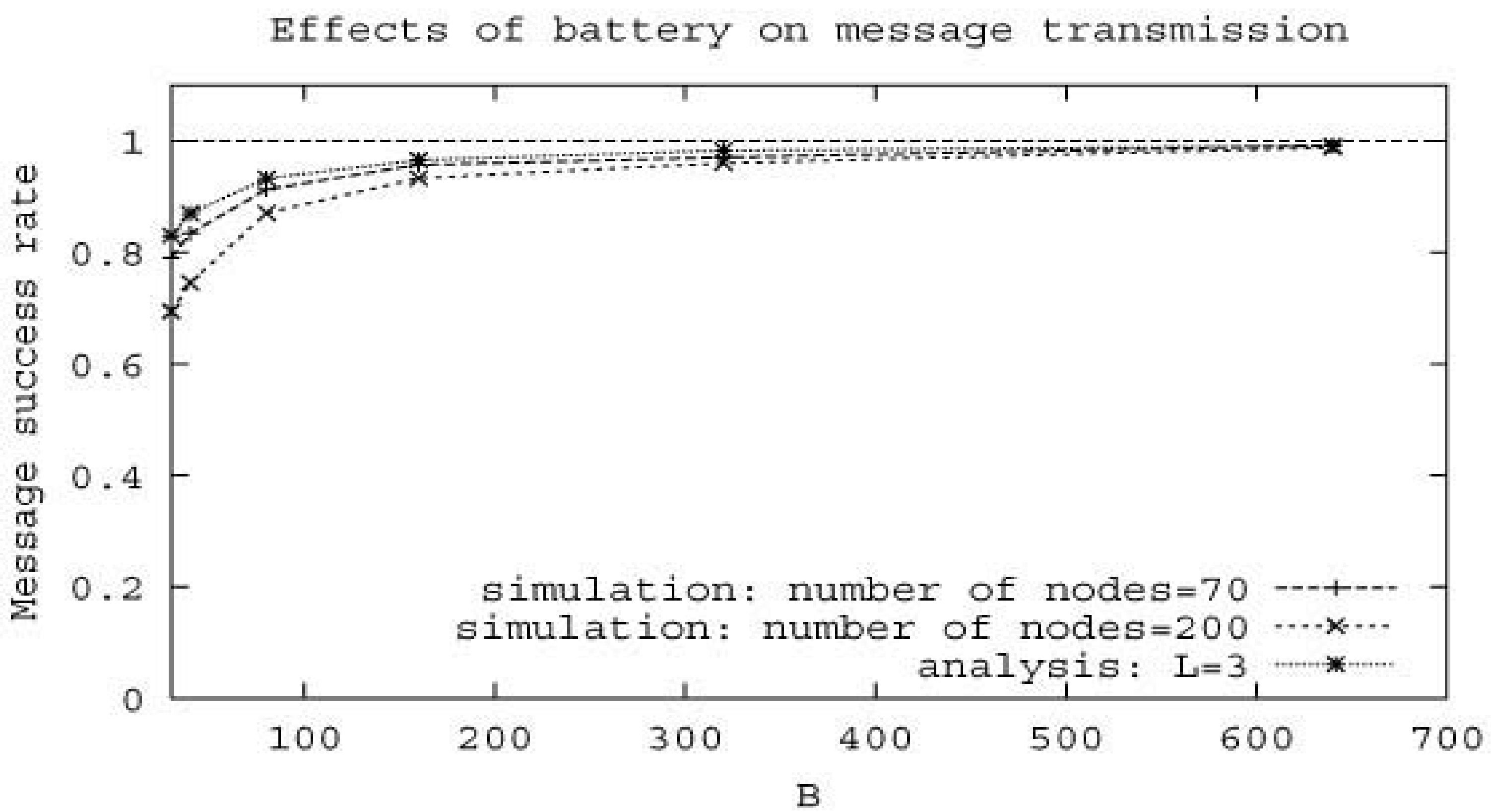
We Design a Cheat-Proof Payment Scheme

- Cheating cannot increase a player's welfare.
- In case of collusion, cheating cannot increase the sum of colluding players' welfares.

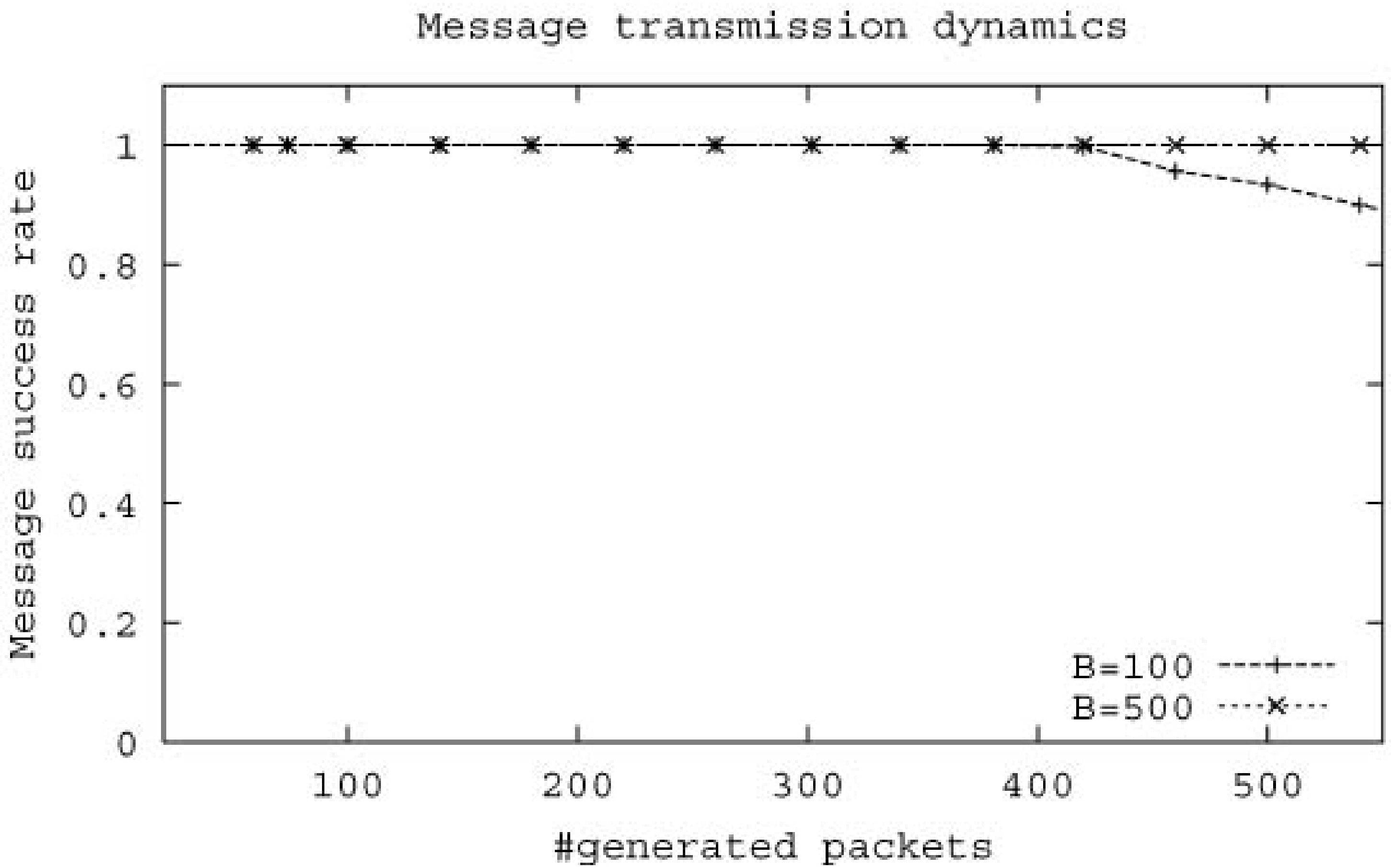
Evaluation: Overhead

Signing Alg.	Send (ms)	Forward (ms)	Header (bytes)	Receipt (bytes)
RSA 1024	10.4	0.3	128	180
ECNR 168	7.3	13.2	42	94
ECNR 168 w/ precomp	3.7	6.1	42	94

Effects of Battery on Performance



Dynamics of Message-Success Rate



Summary of Our Results on Mobile Ad Hoc Networks

- We designed a simple scheme to stimulate cooperation.
- Our system is provably secure against (colluding) cheating behaviors.
- Evaluations have shown that the system has good performance.

THANK YOU