# Cloud and Mobile Security Seminar

Spring 2013

Lecture 5: Cryptography-based solutions: Limitations and Challenges

# Cryptographic Primitives Reminder

- Traditional encryption (RSA, AES)
  - Limitations: no service-side computation, limited access control
- Fully-homomorphic encryption (FHE)
  - User can perform any server-side function F on encrypted data
  - Service learns nothing about data or results
- RDBMS-enabled encryption (CryptDB)
  - User can delegate the search function to the service
  - Service learns nothing else about the data
- Attribute-based encryption (not covered, but cool)
  - Allows flexible, role-based access control with encryption

Q: What's still challenging when you use crypto to secure untrusted clouds?

# Crypto Challenges / Limitations

- Key management
- Usability
- Performance
- Applicability
- Cost

# Outline

- Limitation 1: Crypto is not applicable to everything (Yu)
  - **On the impossibility of cryptography alone for privacy-preserving cloud computing.** M. van Dijk and A. Juels. In *Proc. of HotSec*, 2010.
- Limitation 2: Crypto is heavyweight / expensive (Tingting)
  - **On the (im)practicality of securing untrusted computing clouds with cryptography.** Y. Chen and R. Sion. Technical report, State University of New York, 2010.
  - **[On securing untrusted clouds with cryptography.** Y. Chen and R. Sion. In *Proc. of WPES*, 2010.]
- Implications of these two limitations (Roxana)
  - When to cloud and when not to cloud?

# On the Impossibility of Cryptography Alone for Privacy-Preserving Cloud Computing

Marten van Dijk, Ari Juels
RSA Laboratories

***Presented By:***
***Yu Wan***

# Overview

- Background
- Summary
- Cloud Application Class Hierarchy
  - Private Single-Client Computing
  - Private Multi-Client Computing
  - Private Stateful Multi-Client Computing
- How to get Cloud Privacy?
- Conclusion
- Discussions

# Background

- **Marten van Dijk**:
- Collaborated with Craig Gentry on the paper:
- Fully homomorphic encryption over the integers, *Dec. 2009*
- *Research scientist at the MIT CS and AI Laboratory*
- On the Impossibility of Cryptography Alone for Privacy-Preserving Cloud Computing, Aug. 2010
- *RSA Laboratories*

- **Ari Juels:**
- Chief Scientist(2010), RSA Director(2007) of RSA Laboratories

# Summary

- Client's lack of direct resource control raises concern about potential data privacy violations.
- ->Cryptography

  Most powerful: *Fully Homomorphic Encryption*, and it is recently realized as a fully functional construct.
- *Argument*: even with FHE, cryptography alone cannot enforce privacy demanded.

- Defines a hierarchy of three natural classes of private cloud applications, and prove that cryptography alone cannot implement all these classes.

# Benefits of Cloud Computing

- Cloud providers reduce per-unit resource cost, and allow clients to scale resource consumption up or down.

- Flexible and portable

- Could provide greater reliability than local computers:

by using redundant sites and backup storages.

- *Question:* How can clients trust that the cloud provider?

# The Goal

- *Identify Challenges:*

Running applications over client data while not able to learn any information itself

Releasing output values to clients according to an access-control policy

- Provide a *negative message* towards the desired privacy-preserving model, the Holy Grail.

# Hierarchy of Natural Classes of Private Cloud Applications

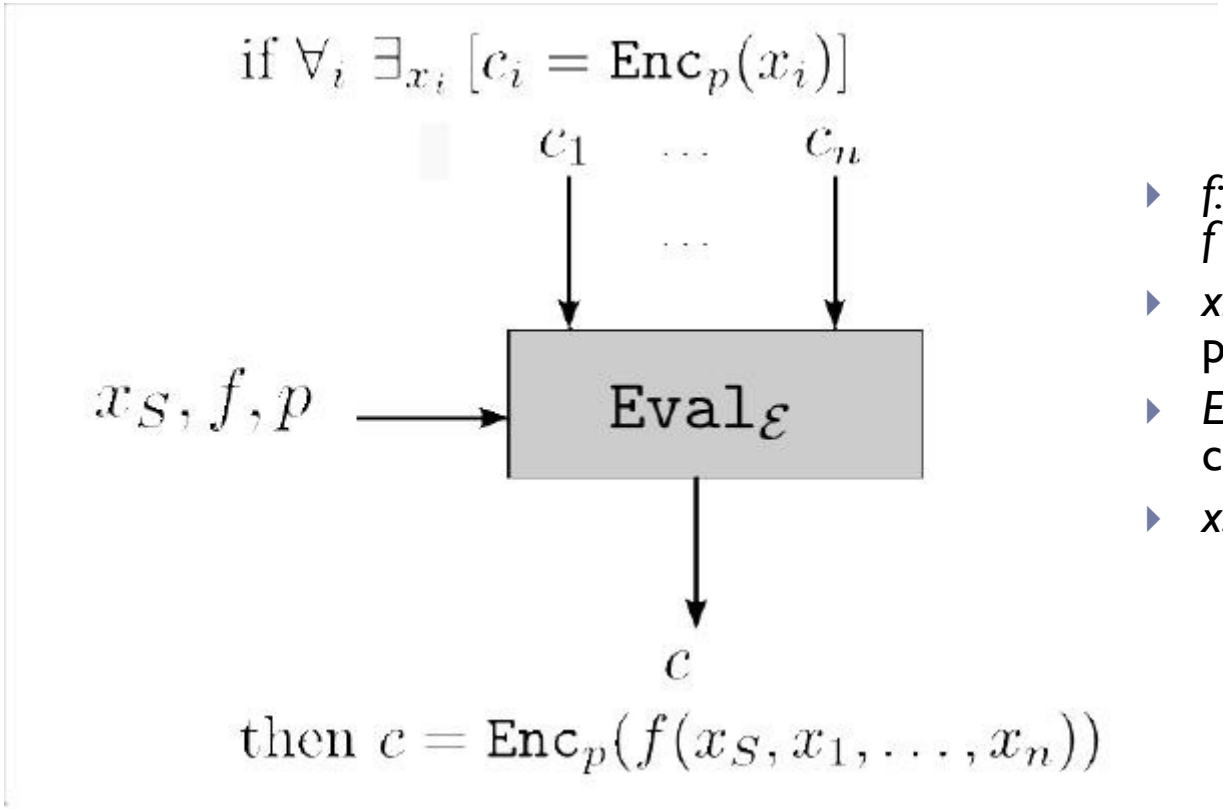
- **Private Single-Client Computing**
- Access control policy:
- only a given client that owns the data may learn any output

- Example:
- Privacy preserving tax-preparation program, such as NetSuite

Input data: $x_i$, financial statements of client $C_i$

Output: A prepared tax return

# Private Single-Client Computing




- *f*: client C asks S to evaluate *f* over x
- $x_i$ encrypted under C's public key
- *Eval*ε: Cloud S evaluates ciphertext *ci*
- $x_s$: other data supplied by S

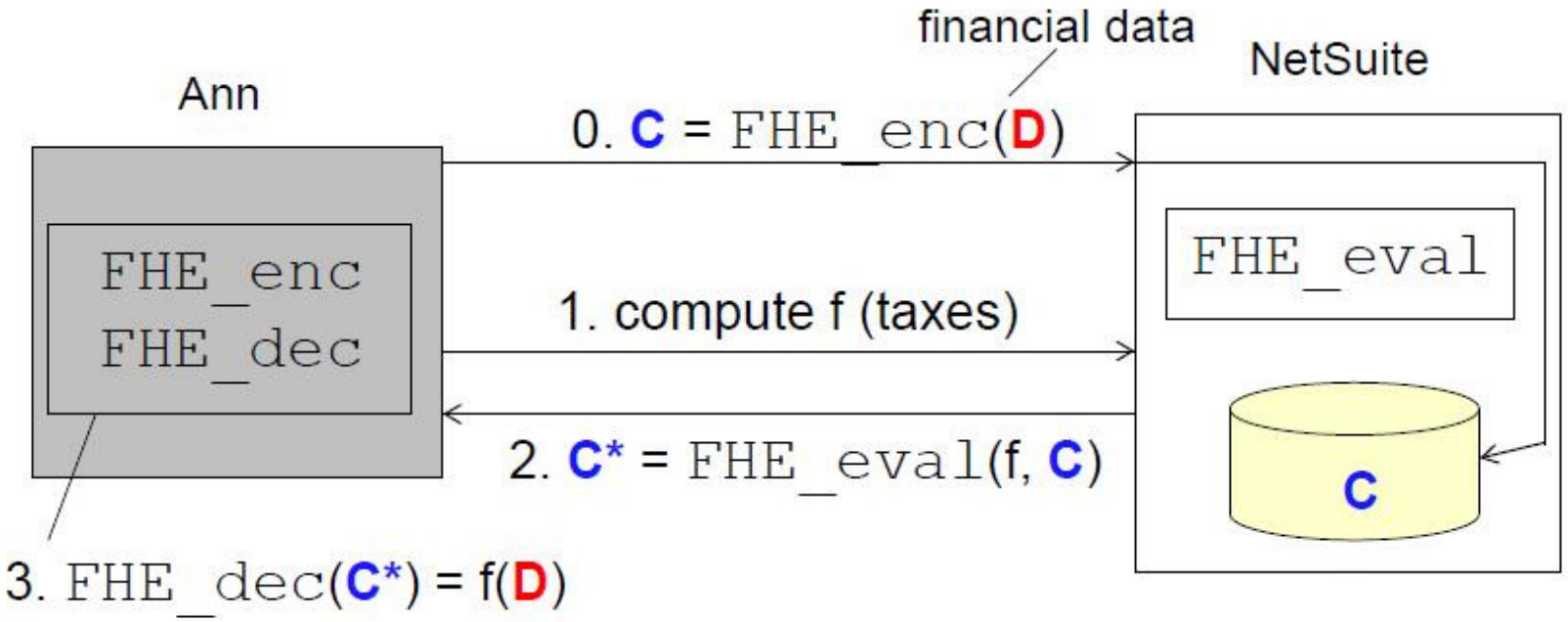
It is **possible** to construct a semantically secure (against chosen plaintext attacks) encryption scheme ε with *Eval*ε that satisfies this property.

*C. Gentry - first FHE scheme*

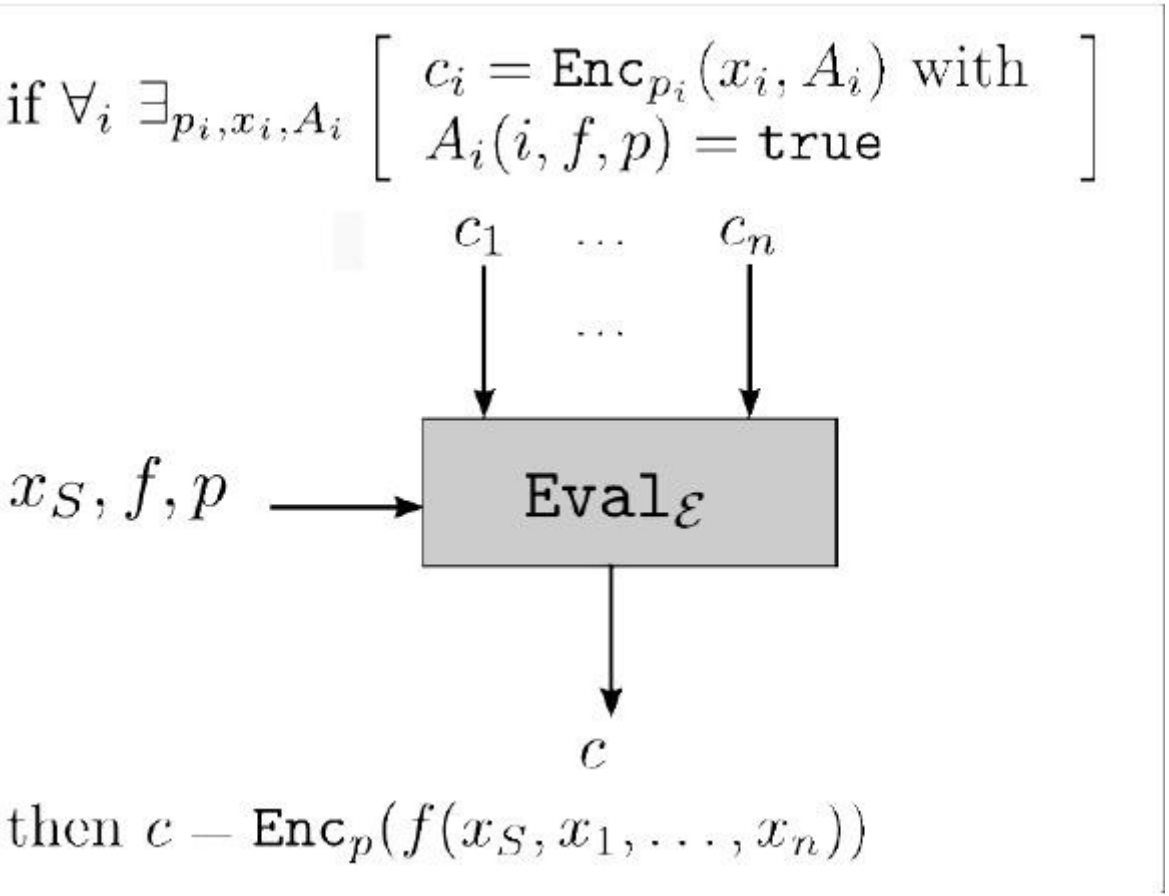
# Private Single-Client Computing-**NetSuite**

Netsuite's `FHE_eval`:

- Expresses tax program (f) with a boolean circuit (f ')
- Evaluate that circuit against the encrypted data C



# Private Multi-Client Computing




- ***Social Networking System***
- *Ai:* access control policy
- When when all *Ai* on all *ci* are met, *Evalε returns the c*

- *Compute data from multiple clients*: need new primitives beyond FHE
- Access-controlled ciphertexts(functional privacy) and re-encryption

# Private Multi-Client Computing
## *Special Case*

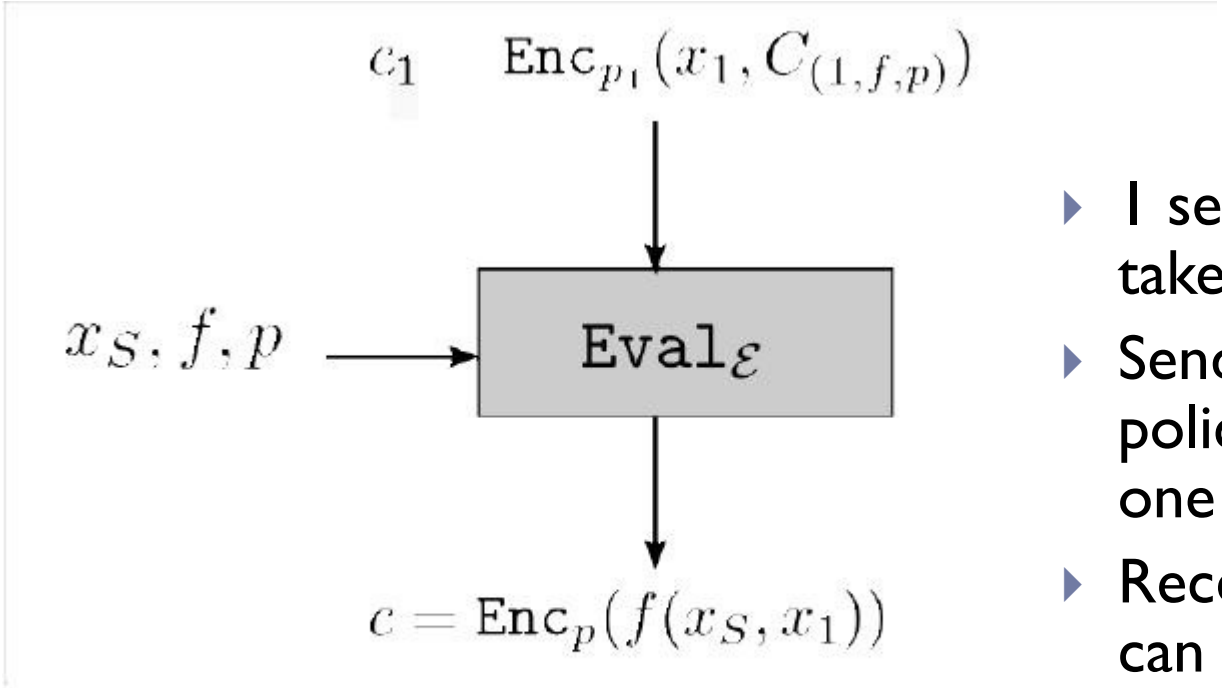



Figure 3: 2-Player Setting

- 1 sender, 1 receiver, *f* takes only $x_s$ and $x_1$.
- Sender, Access-control policy C(1,f,p),allows only one *f* and one key$_{output}$ *p*
- Receiver, knows key *s* to *p*, can decrypt result f($x_s$ ,$x_1$) for any $x_s$. Oracle access to function $x_s$ -> *f*($x_s$,$x_1$)

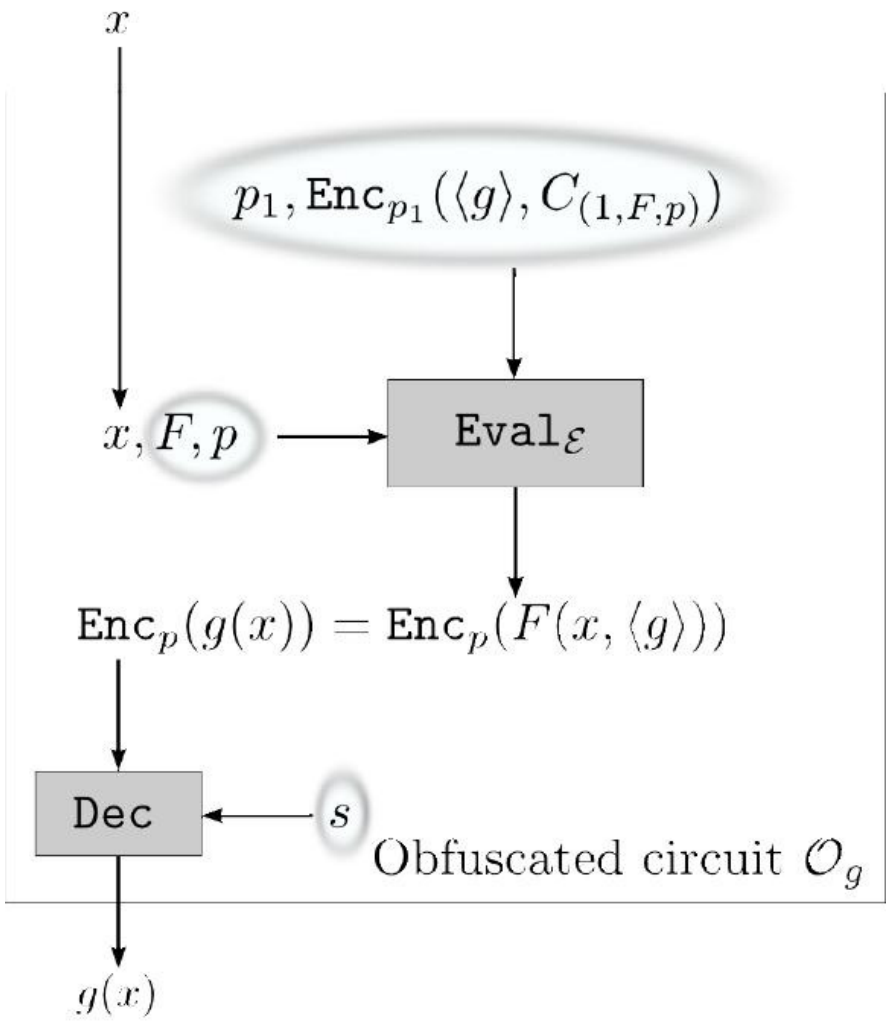
- Proof: Private multi-client computing is in general unachievable using cryptography.

# Private Multi-Client Computing
## *Special Case*

$$Pr[\mathcal{A}(p_1, f, p, c_1, s) = \pi(x_1)] \leq Pr[\mathcal{S}^{\{x_S \to f(x_S, x_1)\}}(1^\lambda) = \pi(x_1)] + \alpha(\lambda).$$

- *Definition 1 formalize:*
- Adversary learns about $x_1$ only what the access control dictates
- An adversary learns no more about $x_1$ than a simulator can learn using oracle access to $x_S$ -> $f(x_S, x_1)$
- Functional privacy ≠ semantic security

# Private Multi-Client Computing
## *Reducing 2-player scheme*




- *O* takes any *g* and outputs $O_g$
- Circuit size, running time are poly in $\lambda$, $\lambda$=|g|
- Evaluate given program *g* over input *x* in the domain of encryption under key $p_1$.
- *F*, meta-circuit, runs any circuit *g* on any input, homomorphically computed function.

# Private Multi-Client Computing
## *Reducing 2-player scheme*

$$BB = (p_1, F, p, c_1 = \mathtt{Enc}_{p_1}(\langle g \rangle, C_{(1,F,p)}), s)$$

- BB, execution environment, all the data needed to realize it.
- **Running 2-player scheme on BB:** takes input *x* and outputs *g(x)*-Obfuscated circuit that execute *g*
- **Definition I** -> Execution of BB obfuscates any poly-size program *g*, NOT achievable

*Note:*

- *General* program obfuscation is impossible, but there are positive results for *specific* form of obfuscation

# Private Stateful Multi-Client Computing

- Basically private multi-client applications where the access-control policy depends on the *history application execution* by S.
- Trustworthy computation environment is necessary
- Relationship between private multi-client computing and stateful private multi-client computing:
- *Do they have identical trusted execution requirements?*

# Private Stateful Multi-Client Computing

*Example:*

- Healthcare-research system -> client: patient or facility

- Facility as a client is permitted to learn aggregate statistics over the full set of records. The stat should never be sufficient to reveal indentifying data.

# How to get Cloud Privacy?

- Practical options for trustworthy computations:

  Trusted computing

- Client distribute data across a collection of service providers

-> limited-capability distributed trust model

# Some Reflections

- Very specific, not written in an easily comprehensible way.
- The proofs and derived conclusions are not explained in detail, especially for Private Stateful Multi-Client Computing.
- Maybe not all cases are considered.
- If proving techniques are altered, would it provide different results.

# Discussions

- If it is not realizable with cryptography(software) alone, what other methods could we use?
- What does the proof mean for us? How important is the proof?


THANK
YOU

# Discussion

- The fact that the overall principle is not achievable doesn't mean we should stop trying
- Nobody (including the authors) knows what the impossibility result means in terms of what's doable vs. what's not
- But people build special-case homomorphic encryption
  - Support specific functionality, but are much more efficient
  - E.g., CryptDB
- It seems that single-user systems are potentially the most clearly-applicable to FHE

# Outline

- Limitation 1: Crypto is not applicable to everything (Yu)
  - **On the impossibility of cryptography alone for privacy-preserving cloud computing.** M. van Dijk and A. Juels. In *Proc. of HotSec*, 2010.
- Limitation 2: Crypto is heavyweight / expensive (Tingting)
  - **On the (im)practicality of securing untrusted computing clouds with cryptography.** Y. Chen and R. Sion. Technical report, State University of New York, 2010.
  - **[On securing untrusted clouds with cryptography.** Y. Chen and R. Sion. In *Proc. of WPES*, 2010.]
- Implications of these two limitations (Roxana)
  - When to cloud and when not to cloud?

# Cloud Economics

- The question: when does it make sense economically to move from private DC to cloud?
  1. Without crypto (i.e., when you trust cloud)
  2. With crypto (i.e., when you don't trust cloud)

- We next discuss this question for each case
  - We'll use Radu Sion's study

- Be advised that many of the results are back-of-the-envelope and were revised in newer versions of the paper/talk
  - But I believe they teach us how to think about cloud economics

# On the (Im)Practicality of Securing Untrusted Computing Clouds with Cryptography

By Yao Chen and Radu Sion

Presented by **Tingting Ai**

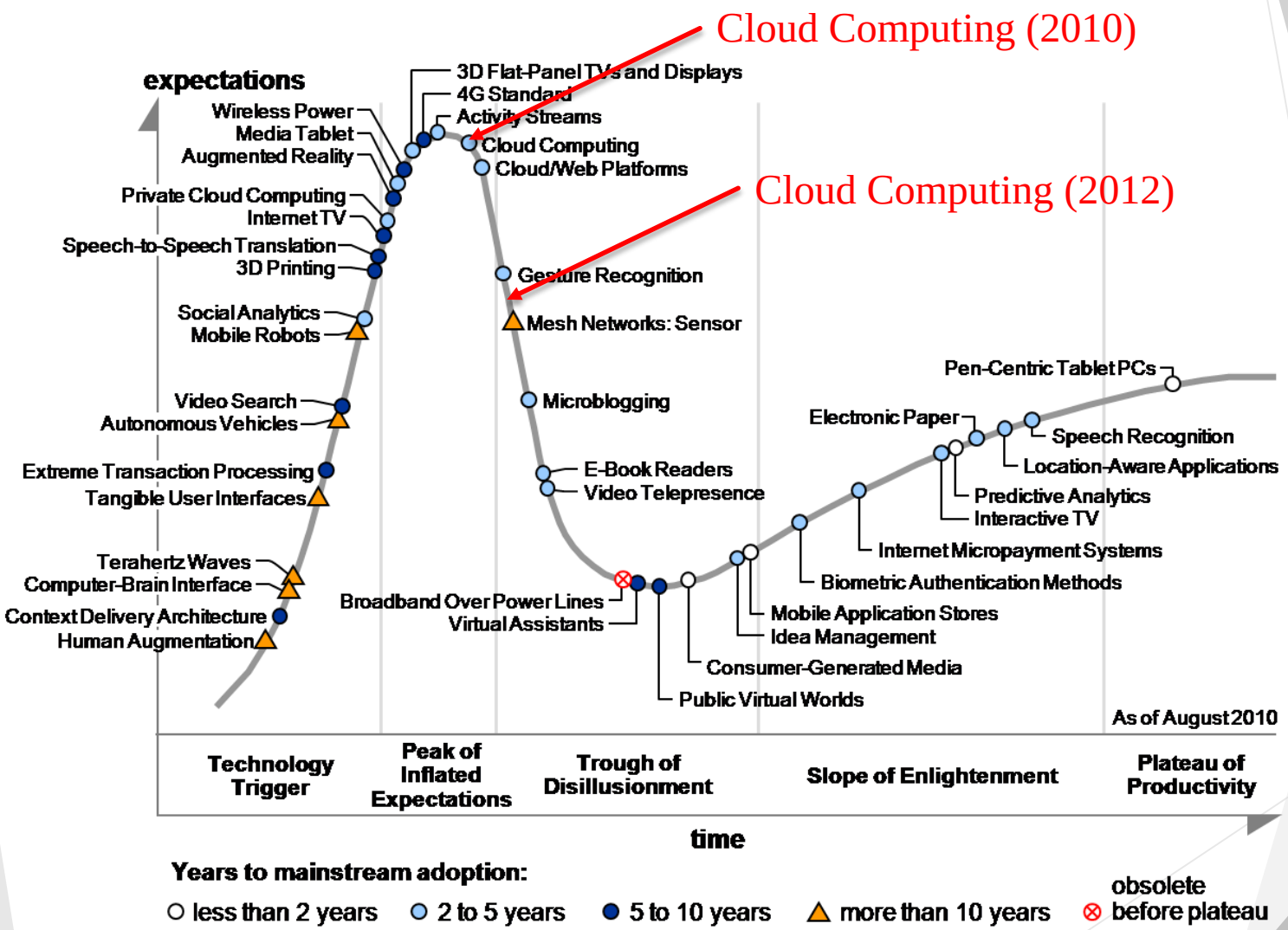
# Agenda

- About the Paper
- Introduction
- Technical Overview
  - Cost Models
  - Cryptography
  - Secure Outsourcing
- Merits & Downsides
- Questions & Discussion

# About the Paper

- Draft version of the paper:
- **On Securing Untrusted Clouds with Cryptography**
  - WPES '10 Proceedings of the 9th annual ACM workshop on Privacy in the electronic society (Oct. 2010), pp.109-114.
  - Cited by 17
  - Total Downloads 712
- **Tranquilizer** to the hype of Cloud Computing (Gartner's Hype Cycle )

# Gartner's Hype Cycle [1]




[1] Source: Managing the Hype Cycle. Online at http://www.digitaltonto.com/2011/managing-the-hype-cycle/

[ Source: http://www.businesscloud9.com/content/gartner-cloud-washing-hype-gives-way-buyer-realism/11345

# Introduction

- Goal of Cloud Computing -
- "Current techniques would more than undo the economy gained by the outsourcing and show little sign of becoming practical."

  -- By Whitfiled Diffie [2]

- Existing secure outsourcing research – addressed integrity and confidentiality.
- **End-to-end viability** of outsourcing?

[2] Source: How Secure Is Cloud Computing? Online at http://www.technologyreview.com/computing/23951/,November 2009.

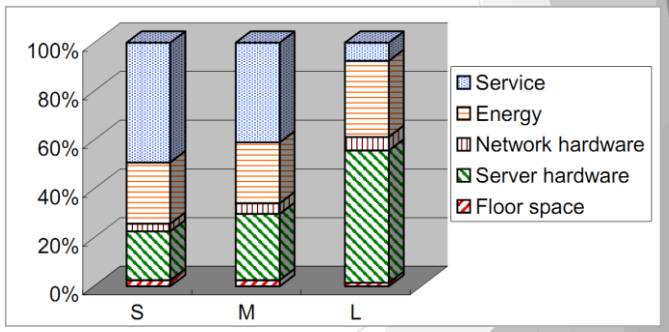
# End-to-end Viability

- Is computing cycle in the cloud cheaper when considering the end-to-end bottom-line?
  - Market's response?
- Is secure processing on behalf of clients possible in untrusted clouds?
  - Integrity, confidentiality and privacy – strong cryptography.
- How much cryptography can we afford in the cloud while maintaining the cost benefits of outsourcing?
  - How many CPU cycles?

# Technical Overview - Cost Models

| Parameters | Home Users | Small Enterprises | Mid-size Enterprises | Large Enterprises |
|---|---|---|---|---|
| CPU utilization | 5-8% | 10-12% | 15-20% | 40-56% |
| Server:Admin ratio | N.A. | 100-140 | 140-200 | 800-1000 |
| Space (sqft/month) | N.A. | $0.5 | $0.5 | $0.25 |
| PUE [2] | N.A. | 2-2.5 | 1.6-2 | 1.2-1.5 |
| # Servers | several | <1,000 | <10,000 | >10,000 |

- **Server Hardware**: scale up, scale out
- **Energy**: PUE
- **Service**: server to admin ratio
- **Network Hardware**: infrastructure (fat tree)
- **Floor space**: S/M – office-level pricing; L – own land

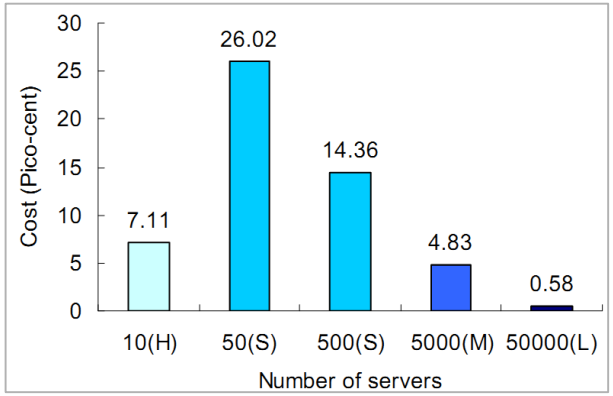



[2] Power Usage Efficiency (PUE): PUE = Total Power Usage / IT Equipment Power Usage

# Technical Overview - Costs

- Equation [3]:

$$CycleCost = \frac{Server + Energy + Service + Netowrk + Floor}{Total\ Cycles}$$




| Provider | Picocents |
|---|---|
| Amazon EC2 | 0.93 - 2.36 |
| Google AppEngine | up to 2.31 |
| Microsoft Azure | up to 1.96 |

- **Storage**: Simply storing bits on disks has become truly cheap.
  - The best price/hardware/MTBF ratio from the sample set is at 26.06 pico-cents/bit/year

[3] Source: Y. Chen, R. Sion, On securing untrusted clouds with cryptography. WPES '10 Proceedings of the 9th annual ACM workshop on Privacy in the electronic society

# Technical Overview - Costs (Cont.)

- **Network Service**:
  - Different pricing levels
  - Costs incurred by both communicating parties
  - CPU overheads for transfer between application layers
  - Additional traffic for reliable networking

| | **H, S** | **M** | **L** |
|---|---|---|---|
| monthly | $44.90 | $95 | $13 |
| bandwidth (d/u) | 30/5 Mbps | per 1Mbps | per 1Mbps |
| dedicated | No | Yes | Yes |
| picocent/bit | 58/346 | 3665 | 500 |

| Per bit transfer cost | |
|---|---|
| **H → cloud** | 800 |
| **S → cloud** | 6,000 |
| **M → cloud** | 4,500 |

* 15/5 Mbps in formal paper, and 15/1 Mbps provided by Time Warner Cable http://www.timewarnercable.com/en/residential-home/internet/plans.html

# Technical Overview - Cryptography

- **Costs on cryptography:**
  - Symmetric Key Crypto: AES, DES, TDES
  - Modular Multiplication: $t_{mul}(|N|) \approx (|N|/d)^2 \times t_d$
  - Modular Exponentiation: $t_{exp}(|N|) \approx |N| t_{sq}(|N|)$
  - Key Size: RSA and NIST key management guideline.
  - RSA: public key encryption $O(k^2)$, private key decryption $O(k^3)$, key generation $O(k^4)$
  - PK Signatures: DSA, RSA and ECC-based signature.
  - Cryptographic Hashes: MD5 and SHA1

(Cost of RSA)

| | 512 bit | | 1024 bit | | 2048 bit | |
|---|---|---|---|---|---|---|
| | Encrypt | Decrypt | Encrypt | Decrypt | Encrypt | Decrypt |
| H | 3.23E+6 | 4.36E+5 | 2.52E+7 | 1.72E+6 | 2.00E+8 | 6.84E+6 |
| S | 6.53E+6 | 8.82E+5 | 5.10E+7 | 3.48E+6 | 4.04E+8 | 1.38E+7 |
| M | 2.20E+6 | 2.96E+5 | 1.71E+7 | 1.17E+6 | 1.35E+8 | 4.65E+6 |
| L | 2.64E+5 | 3.56E+4 | 2.06E+6 | 1.40E+5 | 1.63E+7 | 5.58E+5 |

# Outsourcing – Trust models

- **Trusted clouds**: no unpredictable failures, correct service, service contract and security policies. No malicious insiders.
- **Untrusted clouds**:
  - Data-curious: violating confidentiality polices
  - Access-curious: access patterns, reverse-engineering
  - Malicious: violating integrity policies
- Existing Providers:
  - Virtually all of the above operate in the trusted model – written contracts.
  - Reasons: advertisement-driven revenue and business model; direct cost (free services, e.g. email).

# AWS Customer Agreement [4]

***Last updated March 15, 2012***


- " The service offerings are provided "as is." We and our affiliates and licensors make no representations or warranties of any kind, […] including any warranty that the service offerings or third party content will be uninterrupted, error free or free of harmful components, or that any content, […], will be secure or not otherwise lost or damaged.

- We […] will not be liable to you for any […] damages […]. Further, neither we nor any of our affiliates or licensors will be responsible for any compensation, reimbursement, or damages arising in connection with: (a) your inability to use the services, […] or (d) any unauthorized access to, alteration of, or the deletion, destruction, damage, loss or failure to store any of your content or other data. "

[4] Source: AWS Customer Agreement. Online at http://aws.amazon.com/agreement/

# Secure Outsourcing – Is it worth trying?

- Personal clients: untrusted provider model
- **1. Basic outsourcing**:

$$Savings = Cycles \times c_a - Cycles \times c_b - Trans_{a \to b} \geq 0 \Leftrightarrow Cycles \geq \frac{Trans_{a \to b}}{c_a - c_b}$$

  - Minimal CPU-intensive requirement principle: tasks should be at least 3,800 CPU cycles per every 32 bit of outsourced data.
- **2. Encrypted Data Storage With Integrity**
  - Not efficient: **2+ orders** higher than local storage (no security).
  - Cheapest: hash-based MACs. 10 picocents/bit.
  - Publicly verifiable constructs: crypto-has chains.

$$Cost_{chashchains} = \frac{C_h \times S_{block} \times N_{block} + C_{sig}}{N_{block}}$$

  - 1-45 times more expensive than the MAC based case.

# Secure Outsourcing – Case Studies

- **3. Searches on Encrypted Data**
  - Querying encrypted data with confidentiality: linearly process, or outsource additional secure (meta)data.
  - Profitable when searching is extremely (i) CPU intensive, (ii) selective (amortizing initial transfer cost over multiple searches)
  - Enough storage or B-tree, store index structures on server:

$$Cost_{search} = c_s \times h_{btree}(\gamma \log B + cycles_r)$$

    - 12+ orders of magnitude higher costs without security
  - Security costs: search token (encryption), decrypt received data.

$$Cost_{encrypted\_search} \geq O(ns) \times C_{decrypt} + C_{encrypt} + kC_{crypto\_eval}$$

    - Encryption + crypto evaluation = 1.5 times more than minimum search token transfer costs

# Secure Outsourcing – Case Studies (Cont.)

- **4. Oblivious Data Access**
  - Private Information Retrieval (PIR) schemes: allow a user to retrieve an item from a server in possession of a database without revealing which item she is retrieving
  - Time in theory $\Omega(n) \rightarrow$ poly-logarithmic $\rightarrow O(log(n))$, on real hardware – impractical
  - Basic idea: retrieve M(x, y) in matrix M of size $\sqrt{n} \times \sqrt{n}$ (n bits)

$$C_{cPIR} = C^{s}_{mult} \times \frac{n}{2} + C^{c}_{mult} \times (\sqrt{n} - 1) + 2C_{exp} + 2\sqrt{n} \times Trans_{c \leftrightarrow s}$$

  - **2+ orders** more expensive than transferring the entire database to client.

# Single-server computational PIR (Optional)

- Matrix M of size $\sqrt{n} \times \sqrt{n}$. To retrieve bit M(x, y) privately, the client:
- (i) chooses two random prime numbers p and q of similar bit length, and sends N = pq to the server; (ii) generates $\sqrt{n}$ numbers $s_1, s_2, \ldots, s_{\sqrt{n}}$, such that $s_x$ is a quadratic non-residue (QNR) and the rest are quadratic residues (QR) in $Z_N^*$;
- (iii) sends $s_1, s_2, \ldots, s_{\sqrt{n}}$ to the server.
- For each "column" j $\in (1,\sqrt{n})$ in the $\sqrt{n} \times \sqrt{n}$ matrix, the server:
- (iv) computes the $r_j = \prod_{0<i<\sqrt{n}} q_{ij}$ where $q_{ij} = s_i^2$ if M(i, j) = 1 and $q_{ij} = s_i$ otherwise.
- (v) sends $r_1, \ldots, r_{\sqrt{n}}$ to the client. The client then checks if $r_y$ is a QR in $Z_N^*$ which implies M(x, y) = 1, else M(x, y) = 0

# PIR - Example

- 3 ×3 matrix, get M(3,2), i.e. x=3, y=2
- $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Select random prime number 17 (modulo 17)
  - QR: 1, 2, 4, 8, 9, 13, 15, 16
  - NQR: 3, 5, 6, 7, 10, 11, 12, 14
- $s_1 = 8$, $s_2 = 19$, $s_3 = 5$ ($s_x$ is NQR)
- $r_j = \prod_{0<i<\sqrt{n}} q_{ij}$ where $q_{ij} = s_i^2$ if M(i, j) = 1 and $q_{ij} = s_i$ otherwise.
- $r_1 = 115520$, $r_2 = 760$, $r_3 = 30400$
- Check if $r_2$ is QNR, yes => M(2,1) = 0

# Secure Outsourcing – Case Studies (Cont.)

- **5. Secure Query Processing**
  - Existing secure query mechanisms:
    - Partitioning-based (confidentiality):
    - Homomorphism (correctness): 12+ days per query ?!!!
    - Hash trees/chains, and Signature (authenticate data)
  - Favorable case: linear-time and extremely selective operation. Require database size >=$10^5$
  - Crypto-hashing and linear operation:

$$Savings = kn \times (c_c - c_s), \quad Cost_{hashtree} = C_{verify} + nsC_{hash} \log n,$$
$$Cost_{trans} = nsBTrans_{s \to c}, \quad Savings \geq Cost_{hashtree} + Cost_{trans}$$

$$s \leq \frac{kn(c_c - c_s) - C_{sign_c}}{n(BTrans_{s \to c} + C_{hash_c} \log n)}$$

  - Selectivity factor: s ≤ 0.0001 (signature aggregations: s ≤ $10^{-6}$)

# Summary

- Costs for computing primitives.
- Cryptography costs: TEDS, AES, MD5, etc.
- Conclusion: profitable for computation intensive tasks, requiring several thousand CPU cycles per 32-bit transferred input data.
- Outsourcing costs:
  - Simply storing data: 2+ orders > local storage
  - Oblivious data access protocols: 2+ orders > trivial data transfer
  - Secure querying: mostly cost-unfeasible – no efficient cryptography
- Borderline case: large outsourced database & extremely selective queries (e.g. 0.00001%)

# Merits

- Practical and critical view (tranquilizer to the hype)
- Novel approach to evaluate costs
- Costs for computing primitives
- Covered various crypto-schemes
- Minimal CPU-intensive requirement (Borderline cases)
- Thorough background research (list of references)

# Downsides

- Biased view – other credits of cloud
- Price changes rapidly with time – price difference in the formal paper
- In-depth crypto knowledge and calculation – too hard for general readers, and distract readers from main point (economy)
- Cost evaluation in limited scenarios (H→Cloud)
- Narrow economical analysis

# Thoughts & Discussions

- What the cloud providers can do to reduce the security costs of clients?
- With data outsourcing, can small enterprises switch to the operation mode of home users, thereby achieve more savings?
- Is “cost per bit” reliable? Alternatives?
- Other factors in economic analysis.
- Practicality for different level of computing environments.

# Thank you!

Q: When does it make sense economically to move from private DC to cloud:

- Single-user scenario
- Multi-user scenario

Give example applications.

# Summary of Cloud Economics without Crypto

- Cloud CPU is cheaper
- Cloud storage is the same
- Cloud network costs depend on scenario
  - Single-user scenario
  - Multi-user scenario

# Summary of Cloud Economics without Crypto

- Single-user scenario: Cloud only makes sense if you do lots of computation and transferring very little data

$$\text{Profit} = \text{Cycles} * (c_{DC} - c_{cloud}) - \text{Bits} * (\text{Trans}_{DC \rightarrow cloud} + \text{Network}_{cloud})$$

10s pcents/cycle

~5,500 pcents/bit

- Multi-user scenario: Cloud should usually make sense
  - You save both on transfer and on computation costs

$$\text{Profit} = \text{Cycles} * (c_{DC} - c_{cloud}) - \text{Bits} * \text{Network}_{cloud}$$

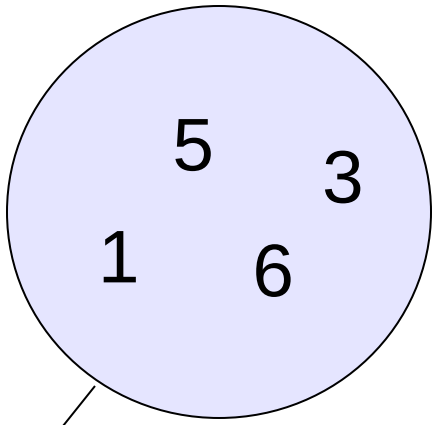
~500 pcents/bit

*Numbers are based on Sion paper and shouldn't be treated as hard truths.*

# So, When Does Moving to Cloud Make Sense?

(Still without crypto)

1. Image processing
2. Single-user cloud FS
3. Backup, archival storage
4. Corporate CRM, Goog Apps
5. Web hosting
6. Data sharing (e.g., scientific)
7. Others?

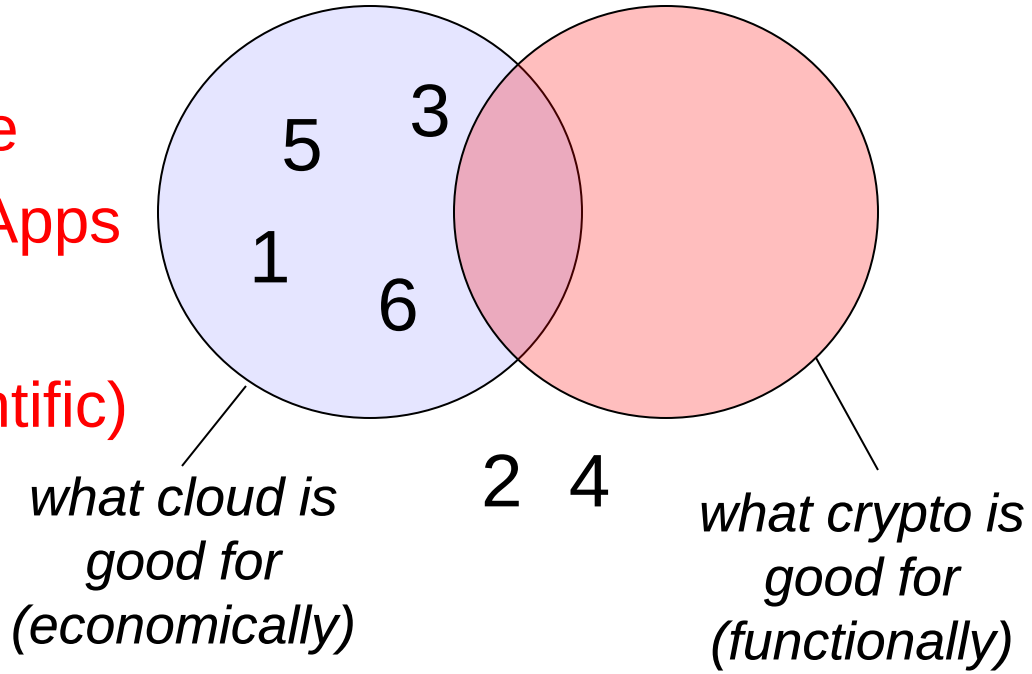



*what cloud is good for (economically)*

2 4

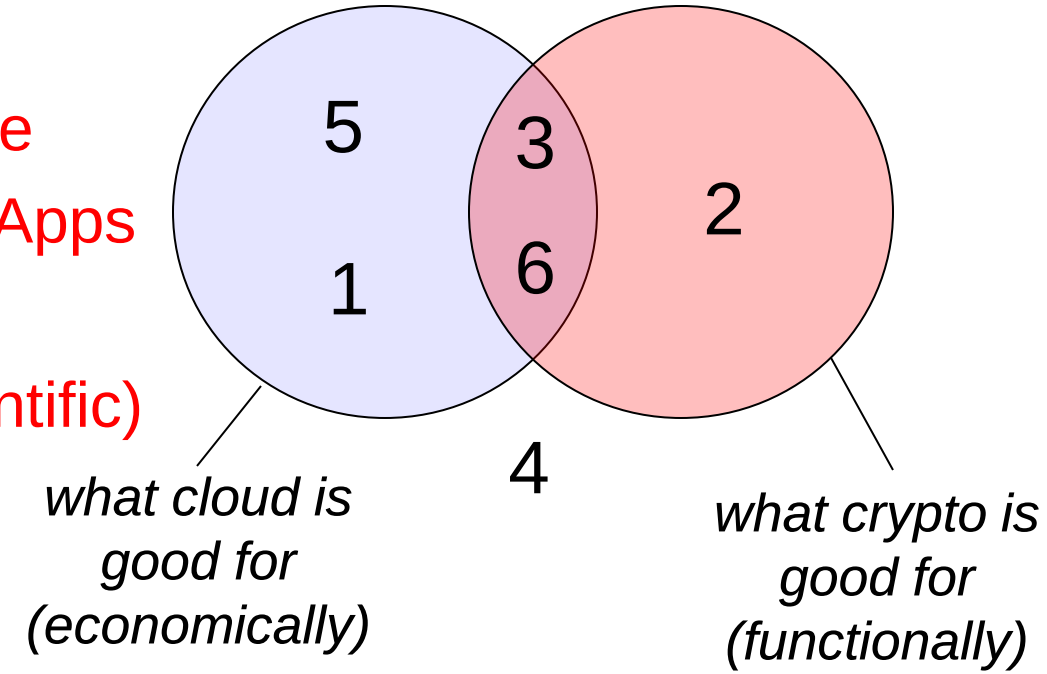
# What Is Crypto Good For?

1. Image processing
2. Single-user cloud FS
3. Backup, archival storage
4. Corporate CRM, Goog Apps
5. Web hosting
6. Data sharing (e.g., scientific)
7. Others?



# What Is Crypto Good For?

1. Image processing
2. Single-user cloud FS
3. Backup, archival storage
4. Corporate CRM, Goog Apps
5. Web hosting
6. Data sharing (e.g., scientific)
7. Others?




- Sion paper: Crypto costs for single-user cloud storage (2)
- Our plan: Use their data to extrapolate for 3 and 6
  - This will be a discussion – I don't actually know answers!

# Crypto Costs: Confidentiality

- Symmetric encryption
  - 25 pcents/bit -- negligible

| | AES<br>128 bits | DES<br>64 bits | TDES<br>64 bits |
|---|---|---|---|
| H | 13 | 37 | 103 |
| S | 25 | 76 | 208 |
| M | 8 | 26 | 70 |
| L | 1 | 3 | 8 |

- Public-key encryption (RSA):
  - 4.04E+8 pcents/bit
    – expensive!

| | 1024 bit | | 2048 bit | |
|---|---|---|---|---|
| | Encrypt | Decrypt | Encrypt | Decrypt |
| H | 2.52E+7 | 1.72E+6 | 2.00E+8 | 6.84E+6 |
| S | 5.10E+7 | 3.48E+6 | 4.04E+8 | 1.38E+7 |
| M | 1.71E+7 | 1.17E+6 | 1.35E+8 | 4.65E+6 |
| L | 2.06E+6 | 1.40E+5 | 1.63E+7 | 5.58E+5 |

- So, is the move to the cloud still worth it for:
  - Backup
  - Data sharing

*Numbers are in pcents per bit.*

# Crypto Costs: Integrity

- Symmetric-key integrity (message authentication codes)
  - They calculated that for MAC: 25 pcents/bit total

- Public-key signatures (DSA)

| | 1024 bit | | 2048 bit | |
|---|---|---|---|---|
| | Sign | Verify | Sign | Verify |
| S | 5.73E+07 | 6.94E+07 | 1.89E+08 | 2.30E+08 |
| L | 9.55E+05 | 1.16E+06 | 3.15E+06 | 3.84E+06 |

- So, is the move to the cloud still worth it for:
  - Backup
  - Data sharing

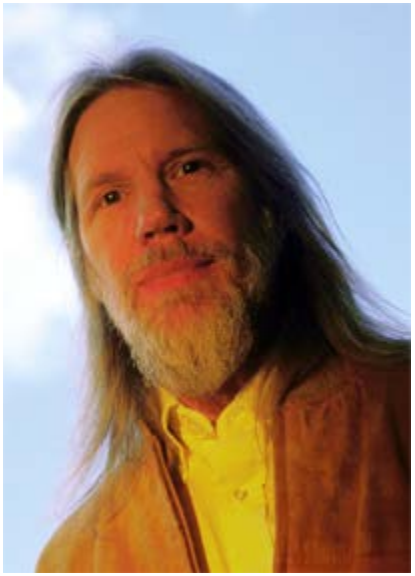
# Crypto Costs: Search/Queries on (Encrypted) Data

- First, non-encrypted search/query: is it worth it?
  - Only if your query is incredibly selective and performing the search is computationally intensive
  - They conclude that search is typically not sufficiently intensive and selected, and leads to 12+ order of magnitude higher costs
- Example rough estimate for encrypted DB queries:
  - If selectivity s < 0.037%, then you start getting profits from computation despite encryption [Sion, et.al., WEPS '10]
- Might be OK for user-facing results
  - Since users can only in-take a fixed number of results

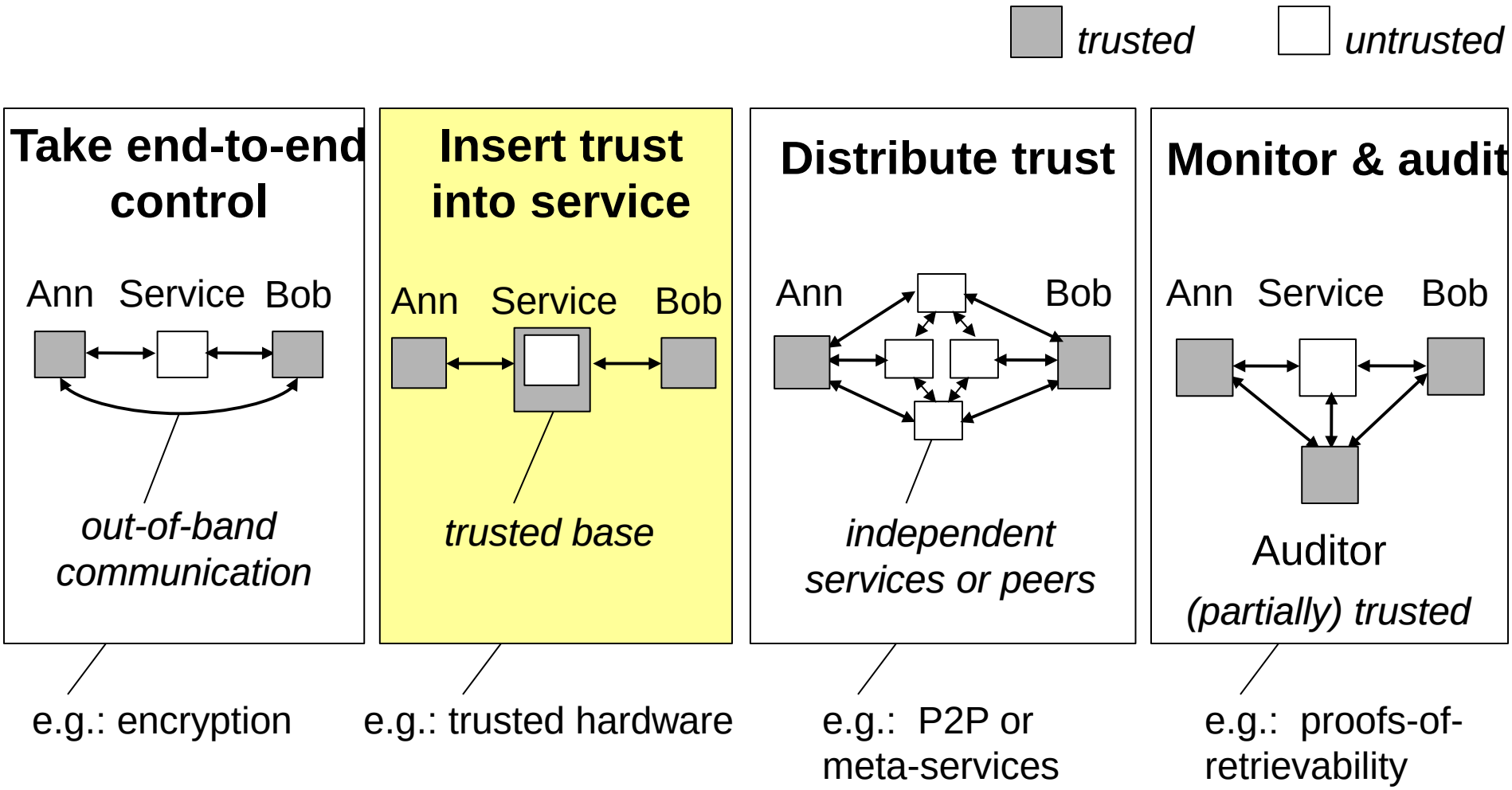
# Paper's Conclusion

- *"[…] current [crypto] techniques would more than undo the economy gained by the outsourcing and show little sign of becoming practical […]"*

Whitfield Diffie (of Diffie/Hellman key exchange)



# Other Ways of Dealing with Untrusted Services



# Next Time: Trusted Hardware

- Trusted hardware overview
  - **Bootstrapping trust in commodity computers.** B. Parno, J. M. McCune, and A. Perrig. In *Proceedings of IEEE Symposium on Security and Privacy (Oakland)*, 2010.
- Applying trusted hardware to clouds
  - **Private virtual infrastructure for cloud computing.** F. Krautheim. In *Proceedings of the USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, 2009.