

Costs and Security in Clouds

Yao Chen and Radu Sion

Abstract Cloud computing has emerged as an important paradigm for deploying services and applications for both enterprises and end-users. In this chapter, we explore two important aspects of cloud computing – *costs* and *security*. We aim to answer two questions: (1) Is cloud computing a cost effective endeavor? (2) How much security can we afford in the cloud while maintaining the cost benefits of outsourcing?

To answer these questions, we start by looking at the economics of computing in general and clouds in particular. Specifically, we derive the end-to-end cost of a CPU cycle in various environments and show that its cost lies between 0.5 picocents in efficient clouds and nearly 27 picocents for small enterprises (1 picocent = $\$1 \times 10^{-14}$), values validated against current cloud pricing. We show that cloud computing makes sense only in scenarios when the clients distance can be offset by a minimal application computation footprint. We then explore the cost of common cryptography primitives as well as the viability of their deployment for cloud security purposes. It turns out that securing outsourced data and computation against untrusted clouds is often costlier than the associated savings, with outsourcing mechanisms up to several orders of magnitudes costlier than their non-outsourced locally run alternatives.

1 Introduction

As computing becomes embedded in the very fabric of our society, the exponential growth and advances in cheap, high-speed communication allow for unprecedented levels of global information exchange and interaction. As a result, new market forces

Y. Chen (✉) • R. Sion

Network Security and Applied Cryptography Lab, Stony Brook University,
Stony Brook, NY, USA

e-mail: yaochen@cs.stonybrook.edu; sion@cs.stonybrook.edu

emerge that propel toward a fundamental, cost-efficient paradigm shift in the way computing is deployed and delivered: computing outsourcing.

Computing outsourcing provides great elasticity and scalability of resources. It minimizes client-side management overheads and benefit from a service provider's global expertise consolidation and bulk pricing, and helps users avoid the capital expense in acquiring computing resources. The past decades' traditional outsourcing paradigms have usually involved established service providers such as IBM that manage or host clients' machines in dedicated data centers. More recently, first storage and then computation outsourcing has been commoditized through the emergence of globally-sized enterprises such as Google, Yahoo, Amazon, and Sun which started offering increasingly complex storage and computation outsourcing "cloud" services. CPU cycles have become consumer merchandise.

So far, the end-to-end viability of cloud computing has mostly not been explored. Is a remotely hosted computing cycle in a cloud indeed cheaper than performing it locally *when considering the end-to-end bottom-line*? It seems the markets have spoken and the increasing number of service providers can be viewed as testimony that this indeed is the case. Yet by what margins? And what are the features of suitable applications for cloud deployment? As the migration from in-house data centers to the clouds is non-trivial and fraught with potentially large costs, asking these questions is essential.

In this chapter, to understand the viability of clouds, we provide a cost model for computing in different environments and derive the dollar cost of primitives such as CPU cycles, storage and network transfers. Using the model, we then evaluate cloud outsourcing end-to-end and derive a threshold principle defining when outsourcing indeed is economically viable, i.e., when computing-related savings outweigh the costs of networking. We then evaluate the footprints and types of applications most suited for cloud deployment.

Despite the associated buzz, clouds have been somewhat less successful in attracting medium to large size corporations. Such clients often fall under strict regulatory compliance requirements for manipulating information or simply are reluctant to place sensitive data and computation logic under the control of a remote, third-party provider, without practical assurances of privacy and confidentiality in which the provider is un-trusted. Significant challenges lie in the path of successful large-scale adoption.

To address this, existing secure outsourcing research addressed several issues including guaranteeing integrity, confidentiality and privacy of outsourced data to secure querying on outsourced encrypted database. Such assurances will likely require strong cryptography as part of elaborate intra- and client-cloud protocols. Yet, strong crypto is expensive. Thus, it is important to ask: how much cryptography can we afford in the cloud while maintaining the cost benefits of outsourcing?

Some believe the answer is simply *none*. For example, in an interview [56] Whitfield Diffie argued that **"current techniques would more than undo the economy [of] outsourcing and show little sign of becoming practical."**

Here we set out to find out whether this holds and if so, by what margins. One way to look at this is in terms of CPU cycles. For each desired un-secured client CPU

cycle, *how many additional cloud cycles can we spend on cryptography*, before its outsourcing becomes too expensive? We end up gaining the insight that today's secure data outsourcing primitives are often orders of magnitude more expensive than local execution, mainly due to the fact that we do not know how to process complex functions on encrypted data efficiently enough. And outsourcing simple operations – such as existing research in querying encrypted data, keyword searches, selections, projections, and simple aggregates – is simply not profitable. Thus, while traditional security mechanisms allow the elegant handling of inter-client and outside adversaries, today it is still too costly to secure against cloud insiders with cryptography.

2 Cost Models

To reach the granularity of compute cycles we explore the cost of running computing at different levels. We chose environments of increasing size: home, small enterprises, mid-size enterprises and large size data centers. The boundaries between these setups are often dynamic and the main reason we're using them is to help differentiate a set of key parameters.¹

2.1 Levels

Home Users (H). We include this scenario as a baseline for a simple home setup containing several computers. This could correspond to individuals with spare time to maintain a small set of computers, or a very small home-based enterprise with no staffing overheads. It is important to consider this scenario as it represents a potentially large slice of the outsourcing market, especially through application such as mail, document, media and personal blog/web hosting. Also this niche is important as it features a set of peculiarities, including access to residential energy pricing, negligible cooling, rental and management costs (as we will not factor individuals' time in).

Small Enterprises (S). We consider here any scenario involving an infrastructure of up to 1,000 servers run in-house in a commercial enterprise. The cost structure will start to feature most of the usual suspects, including commercial energy and network pricing, cooling, space leases, staffing etc. Small enterprises however can not afford custom hardware, efficient power-distribution, and cooling or dedicated buildings among others. More importantly, in addition to power distribution

¹We note it is not the subject of our work to explore in-depth data center infrastructures. A plethora of online sources discuss issues related to data centers, often focusing on power and overall efficiency (most notably James Hamilton's blog [27]).

inefficiencies, due to their nature, small enterprises cannot be run at high utilization as they would be usually under the incidence of business cycles and its associated peak loads.

Mid-size Enterprises (M). We consider here setups of up to 10,000 servers, run by a corporation, often in its own dedicated data center(s). Mid-size enterprises might have some clout and access to better service deals for network service as well as more efficient cooling and power distribution. They are not fully global, yet could feature several centers across one or two time zones, allowing increased independence from local load cycles as well as the ability to handle daily peaks better by shifting loads across timezones. All the above results ultimately in increased utilization (20–25 % est.) and overall efficiency.

Large Enterprises/Clouds (L). Clouds and large enterprises run over 10,000 servers, cross multiple time-zones, often literally at a global level, with large data centers distributed across all continents and often in tens to hundreds of countries. For example Google has built a 30-acre site in Dalles, Oregon, next to a hydroelectric dam providing cheap power. The site is composed of 34,000sqft buildings [33]. Especially in cloud setups, high speed networks allow global-wide distribution and integration of load from thousands of individual points of load. This in turn flattens the 24-h overall load curve and allows for efficient peak handling and comparably high utilization factors (50–60 % est. [28]). Cloud providers run the most efficient infrastructures, and often are at the forefront of innovation. Moreover, clouds have access to bulk-pricing for network service from large ISPs, often one order of magnitude cheaper than mid-size enterprises.

2.2 Factors

We now consider the cost factors that come into play across all of the above levels. These can be divided into a set of inter-dependent vectors, including: hardware (servers, networking gear), building (floor space leasing), energy (running hardware and cooling), service (administration, staffing, software maintenance), and network service. Other breakdown layouts of these factors are possible.

Server Hardware. Hardware costs include servers, racks, power equipment, network equipment, cooling equipment etc. We will discuss network equipment later. Naturally, there are different choices for data centers to increase capacity. *Up-scaling* – the purchase of a smaller number of more expensive off-the-shelf multi-blade servers – is often considered in mid-size enterprises, and features lower software and infrastructure cost advantages. *Scaling out* – deploying massive numbers of low-cost, almost “expendable” custom-designed and often in-house built multi-CPU server boards – is a strategy available to large, cloud-size providers such as Google and Amazon. The advantages of this approach are low hardware costs, low inter-failure correlation and high overall efficiency factors. Sometimes these

two approaches can be combined; e.g., servers embedded with 4–8 CPUs can be considered as scale-out architecture of scale-up nodes [25]. We note that these costs drop with time, likely even by the time this goes to print. For example, while many of the current documented mid-size deployments use single or multi-CPU System-X blade servers at around \$1–2,000 each [32], large data centers deploy custom setups at about \$3,000 for 4 CPUs, near-future developments could yield important changes.² We will be conservative and empirically assume home PC prices of around \$750/CPU, small and mid-size enterprise costs of around \$1,000/CPU (for 2 CPU blades) and cloud-level costs of no more than \$500/CPU.

Energy. Energy in data centers does not only include power, computing and networking hardware but the entire support infrastructure, including cooling, physical security, and overall facilities. With the increasing density of today's rack structure, temperature rises more rapidly than in old server rooms [7]. For example, any additional 40 W/sqft can lead to a rise of 25 °F in 10 min. A simple rough way to infer power costs is by estimating the Power Usage Efficiency (PUE) of the data center. The PUE is a metric to evaluate the energy efficiency of a data center [24] ($\text{PUE} = \text{Total Power Usage} / \text{IT Equipment Power Usage}$). PUE ranges from 1.13 to 1.21 for big providers as claimed by Google, Facebook and 1.22 for efficient data center containers, to over 2 for typical data centers [44, 51]. We will assume 1.2–1.5 PUE for large enterprises, 1.6–2 PUE for mid-size enterprises and 2–2.5 for small enterprises [44]. Costs of electricity are relatively uniform and documented [23].

Service. Evaluating the staffing requirements for data centers is an extremely complex endeavor as it involves a number of components such as software development and management, hardware repair, maintenance of cooling, building, network and power services.

Analytical approaches are challenged by the sparsity of available relevant supporting data sets. We deployed a set of commonly accepted rule of thumb values that have been empirically developed and validate well [29]: the server to administrator ratio varies from 2:1 up to experimental 2,500:1 values due to different degrees of automation and data management. In deployment, small to mid-size data centers feature a ratio of 100–140:1 whereas cloud level centers can go up to 1,000:1 [23, 28].

Network Hardware. To allow for analysis of network intensive protocols, we chose to separate network transport service costs from the other factors of impact in the bottom line for CPU cycle. Specifically, while the internal network infrastructure costs will be factored in the data center costs, network service will not. We will estimate separately the cost of transferring a bit reliably to/from the data center intermediated by outside ISPs' networks. Internal network infrastructure costs can be estimated by evaluating the number of required switches and routers. The design

²In one documented instance, e.g., Amazon is working with Rackable Systems to deliver an under \$700 AMD-based 6 CPU board dubbed CEMS (Cooperative Expendable Micro-Slice Servers) V3.

of scalable large economy network topology with high inter-node bandwidth for data centers is an ever ongoing research problem [45]. We base our results on some of the latest state of the art research, deploying fat tree interconnect structures. Fat trees have been shown to offer significantly lower overall hardware costs with good overall connectivity factors. For example inter-connecting a 27,648 node cluster with Ethernet switching can be done for under \$8.64 million [45], assuming \$3,000 48-port GigE switches at the edge, aggregation and core layers.

Floor Space. Floor space costs vary wildly, by location and use. While office space can be had for up to tens of dollars/sqft/month in Manhattan, data center space can be had at much lower rates, being as low as \$0.1/sqft/month [15, 16, 48]. While small to mid-size enterprises usually have data centers near their location (thus sometimes incurring office-level pricing), large companies such as Google and Microsoft tend to build data centers on owned land, in less populated place where the per sqft price can be brought down much lower, often amortized to zero over time.

We also note that floor surface is directly related to power consumption and cooling with designs supporting anywhere from 40 to 250 W/sqft [21]. Thus, the overall power requirements (driven by CPUs) impact directly the required floor space.

3 Cost Primitives

Armed with knowledge of the above factors, we now estimate the cost of basic computing primitives.

3.1 CPU Cycles

We start by evaluating the amortized dollar cost of a CPU cycle in Eq. (1). See notations in Table 1 and various setups' parameters in Table 2.

Table 1 Notations for Eq. (1)

Symbol	Definition
N_s, N_w	Number of servers, switches
α	administrator: server ratio
β	W/sqft
λ_s, λ_w	Server, switch price
λ_p, λ_f	Personnel, floor cost per second
λ_e	Electricity price/(W·s)
μ	CPU utilization
ν	CPU frequency
τ_s, τ_w	Servers, switches lifespan (5 years)
w_p, w_i	Server power at peak, idle

Table 2 Sample key parameters

Parameters	Home	Small	Medium	Large
CPU utilization	5–8 %	10–12 %	15–20 %	40–56 %
server:admin ratio	N.A.	100–140	140–200	800–1k
Space (sqft/month)	N.A.	\$0.5	\$0.5	\$0.25
PUE	N.A.	2–2.5	1.6–2	1.2–1.5

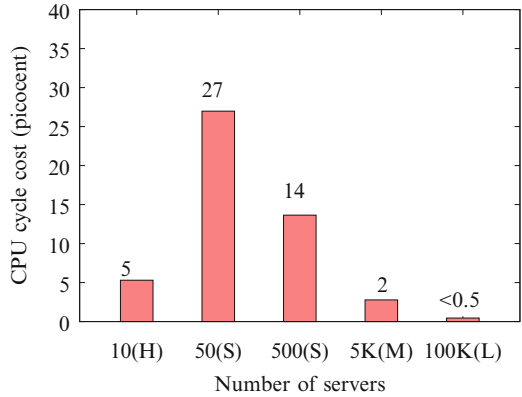
Table 3 Current pricings of a CPU cycle from major cloud providers

Provider	Picocents
Amazon EC2	0.93–2.36
Google AppEngine	Up to 2.31
Microsoft Azure	Up to 1.96

$$\begin{aligned}
 \text{CycleCost} &= \frac{\text{Server} + \text{Energy} + \text{Service} + \text{Network} + \text{Floor}}{\text{Total Cycles}} \\
 &= \frac{\lambda_s \cdot N_s / \tau_s + (w_p \cdot \mu + w_i \cdot (1 - \mu)) \cdot \text{PUE} \cdot \lambda_e + \frac{N_s}{\alpha} \cdot \lambda_p + \lambda_w \cdot N_w / \tau_w + \lambda_f \cdot \frac{(w_p \cdot \mu + w_i \cdot (1 - \mu)) \cdot \text{PUE}}{\beta}}{\mu \cdot v \cdot N_s}
 \end{aligned} \tag{1}$$

The results are depicted in Fig. 1, costs ranging from 0.45 picocents/cycle in very large cloud settings all the way to (S), the costliest environment, where a cycle costs up to 27 picocents ($1 \text{ US picocent} = \1×10^{-14}).

Fig. 1 CPU cycle costs



We validate our results by exploring the pricing of the main cloud providers (Table 3). The prices lie surprisingly close to each other and to our estimates, ranging from 0.93 to 2.36 picocents/cycle. The difference in cost is due to the fact that these points include not only CPUs but also intra-cloud networking, instance-specific disk storage and cloud providers' profit.

Table 4 Summarized network service costs [28, 49]

	H, S		M	L
Monthly	\$44.90	\$200	\$95	\$13
Bandwidth (d / u)	15/5 Mbps	per Mbps	per Mbps	
Dedicated	No	Yes	Yes	Yes
Picocent/bit	115/345	>7,000	3,665	500

Table 5 Per bit transfer costs

Settings	Cost (picocent)
(H, S) \rightarrow Cloud	900
(M) \rightarrow Cloud	4,500

3.2 Network Service

Published numbers place network service costs for large data centers at around \$13/Mbps/month and for mid-size setups at \$95/Mbps/month [28] for *guaranteed* bandwidth. Home user and small enterprise pricing usually benefits from economies of scale and numbers are readily available, e.g., Optimum Online provides 15/5 Mbps internet connection for small business starting at \$44.9/month. We note however that the quoted bandwidth is not guaranteed and refers only to the hop connecting the client to the provider. However, if home users or small enterprises were to order *guaranteed* network service, the price is much higher (around \$200/Mbps/month as quoted to us by network providers.). In this work, we mainly consider *non-guaranteed* network services for home users and small enterprises. We summarize these costs in Table 4.

The end-to-end cost of network transfer includes the cost on both communicating parties and the CPU overheads of transferring a bit from one application layer to another (a minimum about 20 CPU cycles per 32 bit data). Moreover, for reliable networking (e.g., TCP/IP) we need to also factor in the additional traffic and spent CPU cycles (e.g., SYN, SYN/ACK, ACK, for connection establishment, ACKs for sent data, window management, routing, re-transmissions, etc.). If we assume a 1 % TCP re-transmission rate, 1 ACK packet for every two data packets, it costs more than 900 picocents to transfer 1 bit reliably in the $S \rightarrow L$ scenario. We summarize the per bit transfer cost in other scenarios in Table 5.

Moreover, if the applications are not optimized to fully utilize payloads these costs could be much higher, e.g., if only a 32 bit value payload is sent, it would incur upwards of 10,000 picocents per bit.

3.3 Storage

Simply storing bits on disks has become truly cheap. Increased hardware reliability (with mean time between failures rated routinely above a million hours even for consumer markets) and economies of scale resulted in extreme drops in the costs of

disks. Table 6 shows the costs of ownership and operation of a representative sample (by no means exhaustive) set of commonly available consumer-level disks (numbers were obtained in November 2009 from numerous online sources, including the disk vendors' sites, price search engines and independent online hardware discussion sites). Costs incorporate energy and amortized acquisition components. Energy is dominating at 60–70 % of the total cost. We note that actual observed MTBF are often up to about 3.4 times lower than advertised [53]. We considered this in computing the values in Table 6.

In terms of amortized acquisition costs, the Seagate Barracuda provides the best price/hardware/MTBF ratio at 7.67 picocents/bit/year. We observe that hardware constitutes only a small percentage of the overall costs, e.g., for the Maxtor, the amortized hardware acquisition being only 12.16 % of the overall ownership cost. And it holds across all considered (H,S,M,L) levels due to the fact that the existence of a critical mass of disk consumer level buyers results in economies of scale pricing available for everybody.

This leads to the insight that, if storage power and maintenance has been already factored in, then, for most scenarios direct storage hardware costs are very small and *can be mostly ignored when evaluating network and CPU intensive protocols*. Naturally this does not hold if the main costs include long-term data at rest with little or no computation and networking. But, as soon as data gets transferred or processed, direct storage costs become negligible.

4 To or Not To

The insights gained above in the costs of computation, network and storage enable us to explore the viability of the outsourcing endeavor.

We start by noting that it is easy to find scenarios for which it does *not* make sense to outsource to clouds from a strict cost-centric perspective. For example, the CPU cycle costs in Fig. 1 immediately show that it is not profitable to outsource personal workloads (H) to small (S) enterprises (we denote this $H \rightarrow S$) as it would naturally incur additional network bandwidth and CPU cycle costs are much higher for (S).

Yet, what about the other options, $\{H \rightarrow M, H \rightarrow L, S \rightarrow M, S \rightarrow L, M \rightarrow L\}$?

The answer in each of these cases is highly dependent on the type of applications outsourced. Basically, there are three main services the cloud provides: storage, networking and computation. The costs of these three primitives behave differently across computing environments of different scale, thus their outsourcing costs are different. Often the relation between these primitives in an application determines its outsourcing saving. In the following, we explore applications of different types in two outsourcing scenarios (single-client outsourcing and multi-client outsourcing).

Table 6 Magnetic disk storage costs

Disk	Cap. (GB)	Price (USD)	Adj. MTBF (million hours)	Amort. acq. (picocent/bit/year)	Power (W)	Power cost (picocent/bit/year)	Total cost (picocent/bit/year)	Aqc. %
Maxtor Diamond Max	500	53	0.35	32.89	10.85	237.62	270.50	12.16
Hitachi Deskstar 7k500	500	67	0.29	49.89	12.30	269.37	319.26	15.63
Hitachi Ultrastar A7K1000	1,024	153	0.35	46.36	11.50	122.97	169.33	27.38
WD Caviar GP Low Power	1,024	103	0.29	37.45	5.75	61.49	98.93	37.85
Seagate Barracuda 7200.10	750	63	0.35	26.06	10.95	159.87	185.93	14.02

4.1 Single-Client Model

One of the simplest computation outsourcing scenarios involves clients shifting their *own* CPU-intensive applications onto clouds, to save costs. Later *these same clients* (or delegates thereof) will access these cloud-hosted applications for their own use. An example of this are *large corporations considering migrating in-house data centers to clouds*.

Naturally, this is feasible when the savings outweigh the outsourcing overhead costs. In general, outsourcing a computation load from environment a to environment b is economically justified when

$$\begin{aligned} \text{Savings} &= \text{Cycles} \times c_a - \text{Cycles} \times c_b - \text{Trans}_{a \rightarrow b} \geq 0 \\ \Leftrightarrow \text{Cycles} &\geq \frac{\text{Trans}_{a \rightarrow b}}{c_a - c_b} \end{aligned} \quad (2)$$

where Cycles is the number of CPU cycles needed per bit data, and c_x denotes the CPU cycle cost for environment $X \in \{H, S, M, L\}$. We call this the *first minimal CPU-intensive requirement criterion* (we will also call this the “first outsourcing criterion”):

First outsourcing criterion:

For an application accessed mainly by clients in environment a , outsourcing it from a to another environment b is economically justified iff. its computation load exceeds $\frac{\text{Trans}_{a \rightarrow b}}{c_a - c_b}$ compute cycles per transferred input bit.

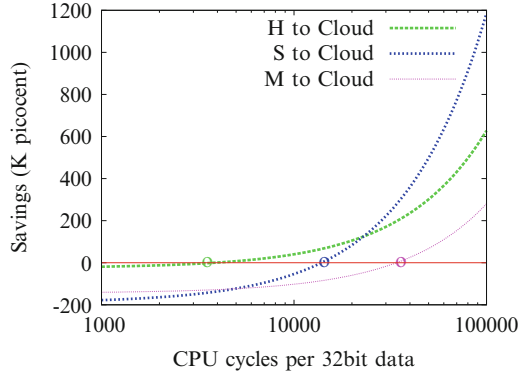
To illustrate, consider a 32 bit item in the $S \rightarrow L$ case. We know from Sect. 3.2, that the cost of reliably transferring 32 bits can be anywhere 28,000 and 320,000 picocents depending on the nature of the connection and whether connection establishment costs are amortized across multiple sends. For consistency, we disregard for now any application-specific costs, such as the existence of results and their transfer costs. As a lower bound, we get

$$\text{Cycles} \geq \frac{\text{Trans}_{S \rightarrow L}}{c_S - c_L} \in (1,000, 12,000).$$

In other words, *if the task at hand requires anywhere less than 1,000 CPU cycles (in the most optimized possible case) per 32 bits of input data, it is not profitable to outsource from a home setting to a large cloud.*

Moreover, 1,000 turns out to also be a lower bound across all outsourcing options as can be seen in Fig. 2. For $H \rightarrow L$, we have anywhere between $\text{Cycles} > 6,400$

Fig. 2 Cost savings of outsourcing per 32 bit data from $S \rightarrow L$, $H \rightarrow L$, $M \rightarrow L$ with increasing application computation load. The lower bounds on the numbers of CPU cycles needed to justify cloud outsourcing are 1,000, 6,400, and 96,100 respectively



and $Cycles > 71,000$. For $M \rightarrow L$, due to the much higher network costs of (M), 32 bit transfers can cost anywhere between 144,000 and 1,615,000 picocents, which results in anywhere between $Cycles > 96,100$ and $Cycles > 1,070,000$.

Applications which are well suited in such CPU-intensive outsourcing include highly scientific computations [52], which usually consume large amounts of CPU. We note that recently Mathworks seems to have tapped this niche, by adding a parallel toolbox in Matlab which enables users to do parallel computing on the Amazon Elastic Compute Cloud [3].

We note that the above *minimal CPU-intensive requirement* criterion specifically refers to network costs that cannot be amortized over multiple transactions, hence the wording “per *transferred* input bit”. Yet, often applications involve significant amounts of already cloud-hosted data inputs, and in such cases, the criterion simply refers to any data that is transferred to/from the cloud.

Simple Storage. Overall, the CPU-intensive requirement of the criterion suggests that purely storage-centric applications are not good candidates for unified-client outsourcing in the cloud. This indeed seems to hold for simple storage outsourcing in which a single data customer places data remotely for future access. For the $S \rightarrow L$ scenario, the amortized cost of storing a bit reliably either locally *or remotely* is under 9 picocents/month (including power). Network transfer however, is at least 900 picocents per accessed bit, a cost that is not amortized and two orders of magnitude higher than storing the data.

Thus, from a pure technological cost-centric point of view, it is simply not effective to store data remotely. Depending on the application network footprint, *outsourced storage costs (incl. network transfer cost) can be upwards of 2+ orders of magnitude higher than local storage*. It’s worth noticing that cloud providers also allow users to mail a portable storage device and upload the data to the cloud over their local network [2]. Yet, as we discussed in Sect. 3.3, simple storage without data processing has become truly cheap even for end users. Using clouds as remote storage is not cost efficient.

Searchable Storage and Databases. Scenarios where outsourcing of data becomes viable include any data processing mechanisms that allow the amortization of networked data transfer over multiple queries to the data set.

Consider for example a searchable outsourced database of size n which allows queries of certain search selectivity s (search results are of size $n * s * S_r$, where S_r is the size of a single result) to be submitted. In this case, the intuition dictates that outsourcing is profitable for a CPU-intensive search process (e.g., for a large database size) and a high selectivity (very low s). For illustration, if searching involves a binary index ($O(\log n)$ CPU cycles), and a comparison takes $C_{compare} = 3$ cycles, we have

$$Savings = \log n \times C_{compare} \times (c_a - c_b)$$

$$Cost_{trans} = nsS_rTrans_{a \rightarrow b},$$

and, for cost viability, we want

$$\begin{aligned} \log n \times C_{compare} \times (c_a - c_b) &\geq nsS_rTrans_{a \rightarrow b} \\ \Leftrightarrow s &\leq \frac{\log n \times C_{compare} \times (c_a - c_b)}{nS_rTrans_{a \rightarrow b}} \end{aligned}$$

In the $S \rightarrow L$ scenario, for a database of $n = 10^9$ keywords and $S_r = 32$ bits, this results in $s \leq 8.3 \times 10^{-11}$. And s will be even lower when database size grows.

4.2 Multi-Client Model

Yet, paradoxically, despite the above conclusion, storage outsourcing seems to be thriving. Just recently, Smugmug, a paid digital photo sharing website, announced \$1M savings a year by outsourcing storage to Amazon S3 [1].

This can be explained as follows. The core storage costs coupled with the lack of an intense-enough CPU load, indeed do not justify outsourcing for a unified client scenario. Yet, web-based enterprises such as Smugmug, by their very nature provide services to third party clients and thus also require mechanisms to handle their clients' remote access, e.g., through often CPU-intensive web interfaces supported by web servers running on actual CPUs. This can increase the per-bit CPU footprint significantly. Moreover, network service pricing for mid-size enterprises can be up to one order of magnitude higher than for clouds, as can be seen in Table 4 – and in effect, clouds can afford to also operate as an efficient content distribution (CDN) service.

Overall, the case for cloud feasibility becomes more complicated in multi-client scenarios. The outsourcing criterion needs to be updated as a function also of the different network service deals of the two environments. Then, outsourcing is economically tenable when

$$Cycles \times c_a - Cycles \times c_b + (Trans_{c \rightarrow a} - Trans_{c \rightarrow b}) \geq 0 \quad (3)$$

where c is the environment from which the majority of client accesses are coming to the outsourced application (Fig. 3). Then, the outsourcing criterion can be rewritten into a more complete (“second outsourcing criterion”) form as follows:

Second outsourcing criterion:

For an application that resides in environment a , whose accesses come mainly from clients in environment c , outsourcing it from a to another environment b is economically justified iff.

its computation load exceeds $\frac{Trans_{c \rightarrow b} - Trans_{c \rightarrow a}}{c_a - c_b}$ compute cycles per transferred input bit – for $c_a \geq c_b$ and $Trans_{c \rightarrow a} \leq Trans_{c \rightarrow b}$, or,
 its computation footprint is lower than $\frac{Trans_{c \rightarrow a} - Trans_{c \rightarrow b}}{c_b - c_a}$ compute cycles per transferred input bit – for $c_a \leq c_b$ and $Trans_{c \rightarrow a} \geq Trans_{c \rightarrow b}$

We can better understand Eq. (3) by detailing the following four cases:

- (i) $c_a \geq c_b$ and $Trans_{c \rightarrow a} \geq Trans_{c \rightarrow b}$, in this case, savings are constantly positive, yielding no *CPU intensive requirement*;
- (ii) $c_a \leq c_b$ and $Trans_{c \rightarrow a} \leq Trans_{c \rightarrow b}$, no savings can be achieved (constantly negative);
- (iii) $c_a \geq c_b$ and $Trans_{c \rightarrow a} \leq Trans_{c \rightarrow b}$, then $Cycles \geq \frac{Trans_{c \rightarrow b} - Trans_{c \rightarrow a}}{c_a - c_b}$
- (iv) $c_a \leq c_b$ and $Trans_{c \rightarrow a} \geq Trans_{c \rightarrow b}$, in this case, $Cycles \leq \frac{Trans_{c \rightarrow a} - Trans_{c \rightarrow b}}{c_b - c_a}$, this unusual case corresponds to an *upper bound* on the amount of computation an application can have before outsourcing becomes counter-productive;

We show in Fig. 3 the cost savings of $S, M \rightarrow L$ with different third party clients and applications at different CPU intensive levels. The CPU intensive requirements are much lower than in the single-client model. Note, given today’s cost points, $M \rightarrow L$ is always profitable and falls into case (i). This may also explain the success of Smugmug outsourcing to Amazon S3. Moreover, if S requires guaranteed network service for the application (see numbers in Table 4), $S \rightarrow L$ also falls into case (i).

For completeness, the equation also covers cases when outsourcing occurs from larger to smaller scale environments, as in (iv). One illustrative instance of this is a large enterprise placing smaller data centers strategically closer to targeted clients. Although CPU cycles will cost more in these smaller data centers, this kind of outsourcing can effectively take advantage of its associated network proximity.

This illustrates another point of feasibility for clouds: content distribution for applications with numerous (often geographically dispersed) clients. This is not only profitable because of the better network service deals that clouds get from major ISPs, but also due to their on-demand scalability promise etc., which is outside of the scope of this chapter.

For multi-client applications such as content distribution or data processing, it is important to consider also intra-cloud communication as well as the actual

Fig. 3 Illustration of the cost savings of outsourcing per 32 bit of data from $a \in \{S, M\}$ to $b = L$ with $c \in \{S, M\}$ – with increasing computation load – according to Eq. (3) (corresponding to the second outsourcing criterion). For $a = S, c = S$, the CPU intensive requirement is 410 cycles per 32 bit

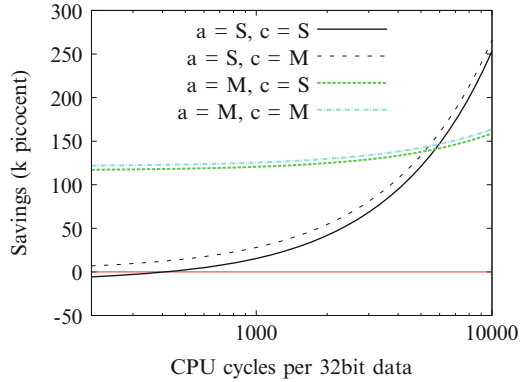


Table 7 Inter- and intra-cloud network transfer pricing (picocent)

	Amazon	Microsoft	Google
Data-in	1,164	1,164	1,164
Data-out	1,979	1,746	1,396
First 10 TB/month			
Next 40 TB/month	1,513	1,746	1,396
Next 100 TB/month	1,280	1,746	1,396
Next 150 TB/month	1,164	1,746	1,396
Intra-cloud/same region	0	0	0
Intra-cloud/inter-region	116	N/A	N/A

profit-including pricing of bit transfers in/out of clouds. For example, at the time of this writing, clouds charge 1,164 picocents per incoming bit, roughly double than what they are paying to ISPs. Table 7 illustrates these pricing points.

5 Cryptography

So far we know that a CPU cycle will set us back 0.45–27 picocents, transferring a bit costs at least 900 picocents, and storing it costs under 100 picocents/year. We now explore the costs of basic crypto and modular arithmetic. All values are in picocents. Note that CPU cycles needed in cryptographic operations often vary with optimization algorithms and types of hardware used (e.g., specialized secure CPUs and crypto accelerators with hardware RSA engines [4] are cheaper per cycle than general-purpose CPUs).

Symmetric Key Crypto. We first evaluate the per-bit costs of AES-128, AES-192, AES-256 and illustrate in Table 8. The evaluation is based on results from the ECRYPT Benchmarking of Cryptographic Systems (eBACS) [9].

Table 8 AES-128, AES-192, AES-256 costs (per byte) on 64-byte input

	AES-128	AES-192	AES-256
S	1.42E + 03	1.48E + 03	1.52E + 03
L	2.37E + 01	2.47E + 01	2.53E + 01

Table 9 Cost of RSA encryption/decryption on 59-byte messages (picocents)

	1,024 bit		2,048 bit	
	Encrypt	Decrypt	Encrypt	Decrypt
S	3.74E + 06	1.03E + 08	8.99E + 06	6.44E + 08
L	6.24E + 04	1.72E + 06	1.50E + 05	1.07E + 07

Table 10 DSA on 59-byte messages. The 1,024-bit DSA uses 148-byte secret key and 128-byte public key. The 2,048-bit DSA uses 276-byte secret key and 256-byte public key

	1,024 bit		2,048 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

Table 11 Costs of ECDSA signatures on 59-byte messages (curve over a field of size 2^{163} , 2^{409} , 2^{571} respectively) (picocents)

	ECDSA-163		ECDSA-409	
	KG/SGN	Verify	KG/SGN	Verify
S	1.36E + 08	2.65E + 08	9.60E + 08	1.91E + 09
L	2.27E + 06	4.41E + 06	1.60E + 07	3.19E + 07
	ECDSA-571			
	KG/SGN	Verify		
S	2.09E + 09	4.18E + 09		
L	3.48E + 07	6.96E + 07		

RSA. Numerous algorithms aim to improve the speed of RSA, mainly by reducing the time to do modular multiplications. In Table 9, we illustrate the costs of RSA encryption/decryption using benchmark results from [9].

PK Signatures. We illustrate costs of DSA, and ECDSA signatures based on NIST elliptic curves [9] in Tables 10 and 11.

Cryptographic Hashes. We also show per byte cost of MD5 and SHA1 with varied input sizes in Table 12.

Table 12 Per-byte cost of MD5 and SHA1 (with 64- and 4,096-byte input)

	MD5		SHA1	
	4,096	64	4,096	64
S	1.52E + 02	3.75E + 02	2.14E + 02	6.44E + 02
L	2.53E + 00	6.25E + 00	3.56E + 00	1.07E + 01

6 Secure Outsourcing

Thus armed with an understanding of computation, storage, network and crypto costs, we now ask whether securing cloud computing against insiders is a viable endeavor.

We start by exploring what security means in this context. Naturally, the traditional usual suspects need to be handled in any outsourcing environment: (mutual) authentication, logic certification, inter-client isolation, network security as well as general physical security. Yet, all of these issues are addressed extensively in existing infrastructures and are not the subject of this work.

Similarly, for conciseness, within this scope, we will isolate the analysis from the additional costs of software patching, peak provisioning for reliability, network defenses etc.

6.1 Trust

We are concerned cloud clients being often reluctant to place sensitive data and logic onto remote servers without guarantees of compliance to their security policies [19, 35]. This is especially important in view of recent sub-poenas and other security incidents involving cloud-hosted data [13, 14, 42]. The viability of the cloud computing paradigm thus hinges directly on the issue of clients' trust and of major concern are cloud insiders. Yet how "trusted" are today's clouds from this perspective? We identify a set of scenarios.

Trusted clouds. In a *trusted* cloud, in the absence of unpredictable failures, clients are served correctly, in accordance to an agreed upon service contract and the cloud provider's policies. No insiders act maliciously.

Untrusted clouds. For *untrusted* clouds, we distinguish several cases depending on the types of illicit incentives existing for the cloud and the client policies with which these will directly conflict. We call a cloud *data-curious* if insiders thereof have incentives to violate confidentiality policies (mainly) for (sensitive) client data. Similarly, in an *access-curious* cloud, insiders will aim to infer client access patterns to data or reverse-engineer and understand outsourced computation logic. A *malicious* cloud will focus mainly on (data and computation) integrity policies and alter data or perform incorrect computation.

Reasonable cloud insiders are likely to factor in the potential illicit gains (the incentives to violate the policy), the penalty for getting caught, as well as the probability of detection. Thus for most practical scenarios, insiders will engage in such behavior only if they can get away undetected with high probability, e.g., when no (cryptographic?) safeguards are in place to enable the detection.

6.2 *Secure Outsourcing*

Yet, millions of users embrace free web apps in **an untrusted provider model**. This shows that today's (mostly personal) cloud clients are willing to trade their privacy for (free) service. This is not necessarily a bad thing, especially at this critical-mass building stage, yet raises questions of clouds' viability for commercial, regulatory-compliant deployment, involving sensitive data and logic. And, from a bottom-line cost-perspective, is it worth even trying? This is what we aim to understand here.

In the following **we will assess whether clouds are economically tenable if their users do not trust them and therefore must employ cryptography and other mechanisms to protect their data**. A number of experimental systems and research efforts address the problem of outsourcing *data* to *untrusted service providers*, including issues ranging from searching in remote encrypted data to guaranteeing integrity and confidentiality to querying of outsourced data. In favor of cloud computing, we will set our analysis in the most favorable $S \rightarrow L$ scenario, which yields most CPU cycle savings.

6.3 *The Case for Basic Outsourcing*

Before we tackle cloud security, let us look at the simplest computation outsourcing scenario (where clients outsource data to the cloud, expect the cloud to process it, and send the results back). In Chap. 1, we show that, to make (basic, unsecured) outsourcing cost effective, the cost savings (mainly from cheaper CPU cycles) need to outweigh the cloud's distance from clients. In $S \rightarrow L$, outsourced tasks should perform at least 1,000 CPU cycles per every 32 bit data, otherwise it is not worth outsourcing them.

6.4 *Encrypted Data Storage with Integrity*

With an understanding of the basic boundary condition defining the viability of outsourcing we now turn our attention to one of the most basic outsourcing scenarios in which a single data client places data remotely for simple storage purposes. In the $S \rightarrow L$ scenario, the amortized cost of storing a bit reliably either locally *or remotely*

is under 9 picocents/month (including power). Network transfer however, is of at least 900 picocents per accessed bit, a cost that is not amortized and two orders of magnitude higher.

From a technological cost-centric point of view it is simply not effective to store data remotely: **outsourced storage costs can be upwards of 2+ orders of magnitude higher than local storage** for the $S \rightarrow L$ scenario *even in the absence of security assurances*.

Cost of Security. Yet, outsourced storage providers exist and thrive. This is likely due to factors outside of our scope, such as the convenience of being able to have access to the data from everywhere or collaborative application scenarios in which multiple data users share single data stores (multi-client settings). Notwithstanding the reason, since consumers have decided it is worth paying for outsourced storage, the next question we ask is, how much more would security cost in this context? We first survey some of the existing work.

Several existing systems encrypt data before storing it on potentially data-curious servers [10, 12, 43]. File systems such as I³FS [34], GFS [22], and Checksummed NCryptfs [54] perform online real-time integrity verification.

It can be seen that two main assurances are of concern here: integrity and confidentiality. The cheapest integrity constructs deployed in most of the above revolve around the use of hash-based MACs. As discussed above, SHA-1 based keyed MAC constructs with 4,096-byte blocks would cost around 4 picocent/byte on the server and 200 picocents/byte on the client side, leading to a total cost of about 25 picocents/bit. This is at least four times lower than the cost of storing the bit for a year and at least one order of magnitude lower than the costs incurred by transferring the same bit (at 900+ picocents/bit). Thus, **for outsourced storage, integrity assurance overheads are negligible**.

For publicly verifiable constructs, crypto-hash chains can help amortize their costs over multiple blocks. In the extreme case, a single signature could authenticate an entire file system, at the expense of increased I/O overheads for verification. Usually, a chain only includes a set of blocks.

For an average of twenty 4,096 byte blocks³ secured by a single hash-chain signed using 1,024-bit RSA, would yield an amortized cost approximately 1 M picocents per 4,096-byte block (30+ picocents/bit) for client read verification and 180+ picocents/bit for write/signatures. This is up to **8 times more expensive than the MAC based case**.

³Douceur et al. [20], show that file sizes can be modeled using a log-normal distribution. E.g, for $\mu^e = 8.46$, $\sigma^e = 2.4$ and 20,000 files, the median file size would be 4 KB, mean 80 KB, along with a small number of files with sizes exceeding 1 GB [5, 20].

6.5 *Searches on Encrypted Data*

Confidentiality alone can be achieved by encrypting the outsourced content before outsourcing to potentially access-curious servers. Once encrypted however, it cannot be easily processed by servers.

One of the first processing primitives that has been explored allows clients to search directly in remote encrypted data [6, 8, 17]. In these efforts, clients either linearly process the data using symmetric key encryption mechanisms, or, more often, outsource additional secure (meta)data mostly of size linear in the order of the original data set. This meta-data aids the server in searching through the encrypted data set while revealing as little as possible.

But is remote searching worth it vs. local storage? We concluded above that simply using a cloud as a remote file server is extremely non-profitable, up to several orders of magnitude. Could the searching application possibly make a difference? This would hold if either (i) the task of searching would be extremely CPU intensive allowing the cloud savings to kick in and offset the large losses due to network transfer, or (ii) the search is extremely selective and its results are a very small subset of the outsourced data set – thus amortizing the initial transfer cost over multiple searches.

We note that existing work does not support any complex search predicates outside of simple keyword matching search. Thus the only hope there is that the search-related CPU load (e.g., string comparison) will be enough cheaper in the cloud to offset the initial and result transfer costs.

Keyword searching can be done in asymptotically constant time, given enough storage or logarithmic if B-trees are used. While the client could maintain indexes and only deploy the cloud as a file server, we already discovered that this is not going to be profitable. Thus if we are to have any chance to benefit here, the index structures need to also be stored on the server.

In this case, the search cost includes the CPU cycle costs in reading the B-tree and performing binary searches within B-tree nodes. As an example, consider 32 bit search keys (e.g., as they can be read in one cycle from RAM), and a 1 TB database. One to three CPU cycles are needed to initiate the disk DMA per reading, and each comparison in the binary search requires another 1–3 cycles (for executing a comparison conditional jump operation). A B-tree with 16 KB nodes will have approximately a 1,000 fanout and a height of 4–5, so performing a search on this B-tree index requires about 100–300 CPU cycles. Thus in this simple remote search, $S \rightarrow L$ outsourcing would result in CPU-related savings of around 2,500–8,000 picocents per access. Transferring 32 bits from $S \rightarrow L$ costs upwards of 900 picocents. Outsourced searching becomes thus more expensive for any results upwards of 36 bytes per query.

6.6 Insights into Secure Query Processing

By now we start to suspect that similar insights hold also for outsourced query processing. This is because we now know that (i) the tasks to be outsourced should be CPU-intensive enough to offset the network overhead – in other words, outsourcing peanut counting will never be profitable, and (ii) existing confidentiality (e.g., homomorphisms) and integrity (e.g., hash trees, aggregated signatures, hash chains) mechanisms can “secure” only very simple basic arithmetic (addition, multiplication) or data retrieval (selection, projection) which would cost under a few of cycles per word if done in an unsecured manner. In other words, *we do not know yet how to secure anything more complex than peanut counting*. And outsourcing of peanut counting is counter productive in the first place. Ergo our suspicion.

We start by surveying existing mechanisms. Hacigumus et al. [26] propose a method to execute SQL queries over partly obfuscated outsourced data to protect data **confidentiality** against a data-curious server. The main functionality relies on (i) partly obfuscating the outsourced data by dividing it into a set of partitions, (ii) query rewriting of original queries into querying referencing partitions instead of individual tuples, and (iii) client-side pruning of (necessarily coarse grained) results. The information leaked to the server is balancing a trade-off between client-side and server-side processing, as a function of the data segment size. Hore et al. [30] explores optimal bucket sizes for certain range queries.

Ge et al. [55] discuss executing aggregation queries with confidentiality on an untrusted server. Unfortunately, due to the use of extremely expensive homomorphisms this scheme leads to large processing times for any reasonably security parameter settings (e.g., for 1,024 bit fields, 12+ days *per query* are required).

Other researchers have explored the issue of **correctness** in settings with potentially malicious servers. In a publisher-subscriber model, Devanbu et al. deployed Merkle trees to authenticate data published at a third party’s site [18], and then explored a general model for authenticating data structures [39, 40]. In [46, 47] as well as in [37], mechanisms for efficient integrity and origin authentication for selection predicate query results are introduced. Different signature schemes (DSA, RSA, Merkle trees [41] and BGLS [11]) are explored as potential alternatives for data authentication primitives. In [36, 50] *verification objects* VO are deployed to authenticate data retrieval in “edge computing” In [31, 38] Merkle tree and cryptographic hashing constructs are deployed to authenticate range query results.

To summarize, existing secure outsourced query mechanisms deploy (i) partitioning-based schemes and symmetric key encryption for (“statistical” only) confidentiality, (ii) homomorphisms for oblivious aggregation (SUM, COUNT) queries (simply too slow to be practical), (iii) hash trees/chains and (iv) signature chaining and aggregation to ensure correctness of selection/range queries and projection operators. SUM, COUNT, and projection usually behave linearly in the database size. Selection and range queries may be performed in constant time, logarithmic time or linear time depending on the queried attribute (e.g., whether it is a primary key) and the type of index used.

For illustration purposes, w.l.o.g., consider a scenario most favorable to outsourcing, i.e., assuming the operations behave linearly and are extremely selective, only incurring two 32-bit data transfers between the client and the cloud (one for the instruction and one for the result). Informally, to offset the network cost of $900 \times 32 \times 2 = 57,600$ picocents, only traversing a database of size at least 10^5 will generate enough CPU cycle cost savings. Thus it seems that with very selective queries (returning very little data) over large enough databases, outsourcing can break even.

Cost of Security. In the absence of security constructs, we were able to build a scenario for which outsourcing is viable. But what about a general scenario? What are the overheads of security there? It is important to understand whether the cost savings will be enough to offset them. While detailing individual secure query protocols is out of scope here, it is possible to reason generally and gain an insight into the associated order of magnitudes.

Existing integrity mechanisms deploy hash trees, hash chains and signatures to secure simple selection, projection or range queries. Security overheads would then include *at least* the (client-side) hash tree proof re-construction ($O(\log n)$ crypto-hashes) and subsequent signature verification of the tree's root. The hash tree proofs are often used to authenticate range boundaries. The returned element set is then authenticated often through either a hash chain (in the case of range joins, at least 30 picocents per byte) or aggregated signature constructs (e.g., roughly 60,000 picocents each, for selects or projections). This involves either modular arithmetic or crypto-hashing of the order of the result data set. For illustration purposes, we will again favor the case for outsourcing, and assume only crypto-hashing and a linear operation are applied.

Consider a database that has $n = 10^9$ tuples of 64 bits each. In that case (binary) hash tree nodes need to be at least 240 bits ($80 + 160$ bits = 2 pointers + hash value) long. If we assume 3 CPU cycles are needed per data item, the boundary condition results in selectivity $s \leq 0.00037$ before outsourcing starts to make economical sense. In a more typical scenario of $s = 0.001$ (queries are returning 0.1 % of the tuples), a per-query loss of over 0.3 US cents will be incurred.

The above holds only for the $S \rightarrow L$ scenario in which hash trees are deployed. In the case of signature aggregation [38, 47], the break-even selectivity would be even lower due to the higher computation overheads.

7 Conclusions

In this chapter, we mused on the dollar cost and security in cloud computing. We started by giving a cost model for computation, storage and networking in different environments. We saw that CPU cycles cost no less than 0.45 picocents, a bit cannot be transferred without paying at least 900 picocents, and stored a year without a pocket setback of at least 100 picocents. We validated the cost model with today's pricing points of clouds.

We determine two “outsourcing criteria”, defining the boundary condition of cloud migration viability. The “first outsourcing criterion” considers unified client applications and postulates that, from a technological cost-centric perspective, outsourcing them *is profitable for computation intensive tasks, specifically, when its (mostly computation-related) cost savings are sufficient to offset client-cloud network distances*. This happens today for unified client applications requiring *no less than 1,000 CPU cycles per each 32 bits of client-cloud transferred input*.

In the case of applications with third-party clients, the feasibility equation changes dramatically. The “second outsourcing criterion” postulates that, for today’s pricing points, for mid-size enterprises, it always makes sense to outsource to cloud. For small enterprises, to make outsourcing profitable, the CPU intensive requirement is much lower than in the single-client model (410 CPU cycles per 32 bit data) or even no CPU intensive requirement if they require guaranteed network service. This is mainly because of the dominating costs of networking, and the fact that in the single-client model, the comparison baseline would not include any networking costs (as the data would be accessed locally).

We also explored whether cryptography can be deployed to secure cloud computing against insiders. We estimated common cryptography costs (AES, MD5, SHA-1, RSA, DSA, and ECDSA) and finally explored outsourcing of data and computation to untrusted clouds. We showed that deploying the cloud as a simple remote encrypted file system is extremely unfeasible if considering only core technology costs. We also concluded that existing secure outsourced data query mechanisms are mostly cost-unfeasible because **today’s cryptography simply lacks the expressive power to efficiently support outsourcing** to untrusted clouds. Hope is not lost however. We found borderline cases where outsourcing of simple range queries can break even when compared with local execution. These scenarios involve large amounts of outsourced data (e.g., 10^9 tuples) and extremely selective queries which return only an infinitesimal fraction of the original data (e.g., 0.00037 %).

References

1. Amazon s3: Show me the money. <http://blogs.smugmug.com/don/2006/11/10/amazon-s3-show-me-the-money/>.
2. Aws import/export. <http://aws.amazon.com/importexport/>.
3. Parallel computing with matlab on amazon elastic compute cloud (ec2). <http://www.mathworks.com/programs/techkits/ec2-paper.html>.
4. IBM 4764 PCI-X Cryptographic Coprocessor. Online at <http://www-03.ibm.com/security/cryptocards/pcixcc/overview.shtml>, 2007.
5. Nitin Agrawal, William J. Bolosky, John R. Douceur, and Jacob R. Lorch. A five-year study of file-system metadata. In *Proceedings of the 5th USENIX conference on File and Storage Technologies (FAST 07)*, Berkeley, CA, USA, 2007. USENIX Association.
6. Georgios Amanatidis, Alexandra Boldyreva, and Adam O’Neill. Provably-secure schemes for basic query support in outsourced databases. In Steve Barker and Gail-Joon Ahn, editors, *DBSec*, volume 4602 of *Lecture Notes in Computer Science*, pages 14–30. Springer, 2007.

7. AMD. Power and cooling in the data centers. Power and cooling in the data centers, Online at http://enterprise.amd.com/Downloads/34146A_PC_WP_en.pdf.
8. Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 535–552. Springer, 2007.
9. Daniel J. Bernstein and Tanja Lange (editors). ebacs: Ecrypt benchmarking of cryptographic systems. Online at <http://bench.cr.yp.to> accessed 30 Jan. 2009.
10. M. Blaze. A Cryptographic File System for Unix. In *Proceedings of the first ACM Conference on Computer and Communications Security*, pages 9–16, Fairfax, VA, 1993. ACM.
11. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *EuroCrypt*, 2003.
12. G. Cattaneo, L. Catuogno, A. Del Sorbo, and P. Persiano. The Design and Implementation of a Transparent Cryptographic Filesystem for UNIX. In *Proceedings of the Annual USENIX Technical Conference, FREENIX Track*, pages 245–252, Boston, MA, June 2001.
13. CNN. Feds seek Google records in porn probe. Online at <http://www.cnn.com>, January 2006.
14. CNN. YouTube ordered to reveal its viewers. Online at <http://www.cnn.com>, July 2008.
15. Katherine Conrad. Data centers hot once again in the bay area. Online at http://findarticles.com/p/articles/mi_qn4176/is_20070401/ai_n18782997.
16. Qwest Communications Corp. Space furniture rental. Online at http://www.qwest.com/about/policy/docs/qcc/documents/WO-sfr-Amd52_111006.pdf.
17. Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security*, pages 79–88, New York, NY, USA, 2006. ACM.
18. Premkumar T. Devanbu, Michael Gertz, Chip Martel, and Stuart G. Stubblebine. Authentic third-party data publication. In *IFIP Workshop on Database Security*, pages 101–112, 2000.
19. Donna Bogatin. Google Apps data risks: Security vs. privacy. Online at <http://blogs.zdnet.com/micro-markets/?p=1021>, February 2007.
20. John R. Douceur and William J. Bolosky. A large-scale study of file-system contents. In *Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 59–70. ACM New York, NY, USA, 1999.
21. Janice Fetzer. Internet data centers: end user & developer requirements. Online at <http://www.utilityeda.com/Summer2006/Mares.pdf>.
22. S. Ghemawat, H. Gobioff, and S. T. Leung. The Google File System. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, pages 29–43, Bolton Landing, NY, October 2003. ACM SIGOPS.
23. Albert Greenberg, James Hamilton, David A. Maltz, and Parveen Patel. The cost of a cloud: Research problems in data center networks. In *SIGCOM Computer Communications Review*, 2009.
24. The Green Grid. Green grid metrics: Describing data center power efficiency. Online at http://www.thegreengrid.org/gg_content/Green_Grid_Metrics_WP.pdf.
25. The Clipper Group. Scale-up and scale-out architectures-ibm provides choice with the xseries. Technical report, 2005.
26. H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra. Executing sql over encrypted data in the database-service-provider model. In *Proceedings of the ACM SIGMOD international conference on Management of data*, pages 216–227. ACM Press, 2002.
27. James Hamilton. Perspectives Blog. Online at <http://mvdirona.com/jrh/work/>.
28. James Hamilton. Internet-scale service efficiency. Large Scale Distributed Systems & Middleware (LADIS 2008), 2008.
29. James Hamilton. On designing and deploying internet-scale services. Technical report, Windows Live Services Platform, Microsoft, 2008.
30. B. Hore, S. Mehrotra, and G. Tsudik. A privacy-preserving index for range queries. In *Proceedings of ACM SIGMOD*, 2004.

31. HweeHwa Pang and Arpit Jain and Krithi Ramamritham and Kian-Lee Tan. Verifying Completeness of Relational Query Results in Data Publishing. In *Proceedings of ACM SIGMOD*, 2005.
32. IBM. IBM blade servers. Online at <http://www-03.ibm.com/systems/bladecenter/hardware/servers/>.
33. Saul Hansell John Markoff. Hiding in plain sight, google seeks more power. Online at <http://www.nytimes.com/2006/06/14/technology/14search.html>.
34. A. Kashyap, S. Patil, G. Sivathanu, and E. Zadok. I3FS: An In-Kernel Integrity Checker and Intrusion Detection File System. In *Proceedings of the 18th USENIX Large Installation System Administration Conference (LISA 2004)*, pages 69–79, Atlanta, GA, November 2004. USENIX Association.
35. Larry Dignan. Will you trust Google with your data? Online at <http://blogs.zdnet.com/BTL/?p=4544>, February 2007.
36. M. Atallah and C. YounSun and A. Kundu. Efficient Data Authentication in an Environment of Untrusted Third-Party Distributors. In *24th International Conference on Data Engineering ICDE*, pages 696–704, 2008.
37. Maithili Narasimha and Gene Tsudik. DSAC: integrity for outsourced databases with signature aggregation and chaining. Technical report, 2005.
38. Maithili Narasimha and Gene Tsudik. Authentication of Outsourced Databases using Signature Aggregation and Chaining. In *Proceedings of DASFAA*, 2006.
39. C. Martel, G. Nuckolls, P. Devanbu, M. Gertz, A. Kwong, and S. Stubblebine. A general model for authenticated data structures, 2001.
40. Charles Martel, Glen Nuckolls, Premkumar Devanbu, Michael Gertz, April Kwong, and Stuart G. Stubblebine. A general model for authenticated data structures. *Algorithmica*, 39(1):21–41, 2004.
41. R. Merkle. Protocols for public key cryptosystems. In *IEEE Symposium on Research in Security and Privacy*, 1980.
42. Jeralyn Merritt. What google searches and data mining mean for you. Online at <http://www.talkleft.com/story/2006/01/25/692/74066>.
43. Microsoft Research. Encrypting File System for Windows 2000. Technical report, Microsoft Corporation, July 1999. www.microsoft.com/windows2000/techinfo/howitworks/security/encrypt.asp.
44. Rich Miller. Microsoft: Pue of 1.22 for data center containers. Online at <http://www.datacenterknowledge.com/archives/2008/10/20/microsoft-pue-of-122-for-data-center-containers/>.
45. Al-Fares Mohammad, Loukissas Alexander, and Vahdat Amin. A scalable, commodity data center network architecture. *SIGCOMM Comput. Commun. Rev.*, 38(4):63–74, 2008.
46. E. Mykletun, M. Narasimha, and G. Tsudik. Authentication and integrity in outsourced databases. In *Proceedings of Network and Distributed System Security (NDSS)*, 2004.
47. E. Mykletun, M. Narasimha, and G. Tsudik. Signature bouquets: Immutability for aggregated/condensed signatures. In *Computer Security - ESORICS 2004*, volume 3193 of *Lecture Notes in Computer Science*, pages 160–176. Springer, 2004.
48. Department of Administration. Records management fact sheet 13. Online at http://www.doa.state.wi.us/facts_view.asp?factid=68&locid=2.
49. Optimum. Optimum online plans. Online at <http://www.buyoptimum.com>.
50. HweeHwa Pang and Kian-Lee Tan. Authenticating query results in edge computing. In *ICDE '04: Proceedings of the 20th International Conference on Data Engineering*, page 560, Washington, DC, USA, 2004. IEEE Computer Society.
51. U.S. Environmental Protection Agency ENERGY STAR Program. Report to congress on server and data center energy efficiency public law 109–431, 2007.
52. J. J. Rehr, J. P. Gardner, M. Prange, L. Svec, and F. Vila. Scientific computing in the cloud, 2008.

53. Bianca Schroeder and Garth A. Gibson. Disk failures in the real world: what does an mttf of 1,000,000 hours mean to you? In *FAST '07: Proceedings of the 5th USENIX conference on File and Storage Technologies*, Berkeley, CA, USA, 2007.
54. G. Sivathanu, C. P. Wright, and E. Zadok. Enhancing File System Integrity Through Checksums. Technical Report FSL-04-04, Computer Science Department, Stony Brook University, May 2004. www.fsl.cs.sunysb.edu/docs/nc-checksum-tr/nc-checksum.pdf.
55. Tingjian Ge and Stan Zdonik. Answering aggregation queries in a secure system model. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 519–530. VLDB Endowment, 2007.
56. Whitfield Diffie. How Secure Is Cloud Computing? Online at <http://www.technologyreview.com/computing/23951/>, November 2009.

Secure Cloud Computing

Jajodia, S.; Kant, K.; Samarati, P.; Singhal, A.; Cohan, V.;
Wang, C. (Eds.)

2014, XII, 343 p. 95 illus., Hardcover

ISBN: 978-1-4614-9277-1