Moving Privacy-Sensitive Services from Public Clouds to Decentralized Private Clouds

Martin Henze, Jens Hiller, Oliver Hohlfeld, Klaus Wehrle
Communication and Distributed Systems, RWTH Aachen University, Germany
{henze, hiller, hohlfeld, wehrle}@comsys.rwth-aachen.de

***Abstract*—Today's public cloud services suffer from fundamental privacy issues, e.g., as demonstrated by the global surveillance disclosures. The lack of privacy in cloud computing stems from its inherent centrality. State-of-the-art approaches that increase privacy for cloud services either operate cloud-like services on user's devices or encrypt data prior to upload to the cloud. However, these techniques jeopardize advantages of the cloud such as elasticity of processing resources. In contrast, we propose decentralized private clouds to allow users to protect their privacy and still benefit from the advantages of cloud computing. Our approach utilizes idle resources of friends and family to realize a trusted, decentralized system in which cloud services can be operated securely and privacy-preserving. We discuss our approach and substantiate its feasibility with initial experiments.**

I. Introduction

Recent privacy incidents such as the global surveillance disclosures [1] demonstrate fundamental issues of today's public cloud services [2]. The root cause of these is the inherent centrality of cloud computing with only a few companies dominating the market. Resulting privacy concerns, missing trust, or legal restrictions on data locality and data ownership make users and companies search for alternatives, especially for US-based cloud providers [3], [4]. Further, missing transparency on data location and purpose of data usage lets users perceive a loss of control over their data which results in resentment to cloud services [5]–[7]. Still, cloud services provide very desirable features that cannot be neglected. Specifically, they offer high availability, easy accessibility, extreme scalability, and simple deployment. Most notably, cloud services provide a high ease of use due to their integration into many devices and applications (e.g., smart phones and web browsers).

State-of-the art solutions to overcome the fundamental privacy concerns with public cloud services can be classified into two categories. First, approaches that shift cloud services to devices controlled by an individual user (e.g., ownCloud, arkOS, or Seafile) typically trade-in availability and scalability for increased privacy. This is mainly due to using only a single or very few devices, often hosted at the user's home and connected via only one residential access line to the Internet. Second, when considering solely the secure storage of data in the cloud, these privacy concerns can partly be countered using encryption (e.g., using BoxCryptor). However, in such a setting data is merely stored in the cloud. Decryption and any processing have to happen on the client device without the scalable resources of the cloud. Still, even in this restricted scenario, the cloud provider can derive valuable meta information, e.g., time and location of data access. Hence, state-of-the-art solutions (partly) preserve privacy and data control at the cost of diminishing the benefits of cloud computing.

In order to realize privacy-sensitive cloud services *and* keep the advantages of cloud computing, we propose an architecture called decentralized private clouds (DPCs). We motivate our approach based on two core observations: *(i)* moving away from the centrality of cloud computing is essential to achieve strong privacy and *(ii)* users posses unused processing resources in their home networks (e.g., routers or network attached storage and set-top boxes) that become increasingly more powerful (e.g., modern home routers have multiple CPU cores and 1 GB of RAM). Hence, we propose to move privacy-sensitive services from public clouds to an individual DPC for each user which consists solely of trusted infrastructure contributed by close friends and family. This allows us to break up the inherent centrality of cloud computing but still leverage decentralized resources to realize its prominent features.

To turn this idea into a technical system, we have to address the following challenges: *(i)* coping with resource constraints (processing, storage, network) of devices available in home networks, *(ii)* achieving the advantages of cloud computing in a highly decentralized cloud over heterogeneous devices, *(iii)* extending trust from individual device owners to the whole DPC, and *(iv)* achieving deployability, most importantly by easing the migration from public cloud services.

In this paper, we discuss how to solve these technical challenges for DPCs spanning over resource constrained devices in home networks and substantiate the feasibility of our proposed approach through initial experiments. With this, we show a promising way for realizing privacy *and* preserving cloud advantages at the same time. Our proposed approach not only constitutes interesting and demanding technical challenges, but also raises exciting legal and economic questions.

II. Problem Analysis and Threat Model

In this paper, we tackle the challenge of realizing fully privacy-preserving cloud services. The main concern regarding privacy for cloud services is the loss of control over data when it is outsourced to the cloud [5], [6]. Besides general security threats such as hacking or denial of service attacks, this loss of control over data is mainly due to three threats. We lay out these threats to privacy for cloud services in our threat model:

Provider access: First, the cloud provider (or one of its employees) might be interested in the data and unauthorizedly access it [8]. Such incidents rarely become public, but in 2012 Dropbox had to acknowledge that an account of one of its employees had been hacked and used to steal customer data [9]. As users are pretty much aware of this threat [3], it is one of the major adoption barriers for cloud computing for both private and corporate users [2], [10].
Governmental access: Second, certain countries access and intercept data within their legislation for safety, security, economic, or scientific purposes [1], [6]. Users have been made aware of the real and imminent threat to their privacy through governmental access to their data in the cloud by the recent global surveillance disclosures [1]. As a consequence, according to a study conducted by the Cloud Security Alliance, about 10% of non-US based companies canceled contracts with US-based cloud providers already in 2013 [11] and the Information Technology & Innovation Foundation estimates the costs of the global surveillance disclosures for US-based cloud providers to 22 to 35 billion US dollar [12].
Indirect access: Finally, it is common for cloud service providers to subcontract (unnoticeably for the user) several cloud providers [6], e.g., to mitigate load peaks. Hence, the previous two threats amplify significantly, as the user does not only have to trust one cloud provider (and the responsible jurisdiction), but a potentially unknown number of additional cloud providers and the jurisdictions they operate in [13].

When analyzing these serious threats to privacy, we identify two root causes that lead to these threats to privacy:
Centrality: The cloud computing market is inherently centralized and only a few companies dominate the market. Hence, their customers' data becomes a valuable "target" for both law enforcement agencies as well as malicious employees. At the same time, users have only very limited choice in selecting a cloud provider or service operator in which they fully trust.
Non-transparency: From the outside, cloud services act as black boxes and little is known or made public about how they are realized in detail. Hence, users of such cloud services do not even know whom they have to trust with their data, e.g., due to governmental or indirect access as discussed above.

III. Decentralized Private Clouds

In order to overcome the severe threats to privacy identified in our threat model, it is thus inevitable to break up their two root causes: centrality and non-transparency. We will do so by introducing decentralized private clouds that run only on devices an individual user explicitly trusts. Our approach allows for a new calibration of the trade-off between privacy and advantages of cloud computing such as availability and accessibility. This stands in stark contrast to today's approaches for preserving privacy for cloud services which come at the cost of diminishing many benefits of the cloud computing paradigm as we detail in the following. When shifting cloud services to devices controlled by the individual users (e.g., using ownCloud, arkOS, or Seafile), often only one device is available to execute these cloud services, which severely

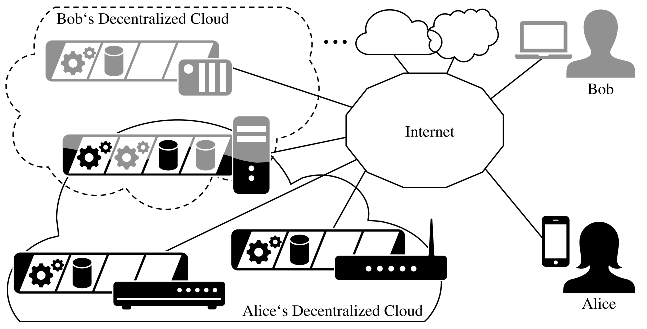

Fig. 1. In our decentralized private clouds architecture, each user builds her decentralized cloud over trusted devices. Here, she can use processing (⚙) and storage (🛢) resources to realize Internet-accessible privacy-sensitive services.

jeopardizes availability and scalability. This issue further exacerbates for private "clouds" hosted at the users' homes as home networks are typically connected via a single residential access line and thus are a single point of failure. Similarly, when still utilizing public cloud services but encrypting data prior to upload (e.g., using BoxCryptor), the inability of clouds to efficiently process encrypted data requires application logic and decryption on the client. This diminishes many advantages of the cloud with respect to processing and accessibility. Additionally, cloud providers can still obtain valuable meta information, e.g., time and location of data access. Hence, current approaches put the user in the dilemma of having to choose between her privacy and the advantages of cloud computing. To overcome this dilemma, we propose to build for each user an individual decentralized private cloud (DPC) over trusted resources. While this approach is certainly not suited for all kinds of cloud applications, it offers the user an additional choice for certain applications which target a small user group and have strong requirements for privacy (especially to protect against corporations and governments). In the following, we will show how our decentralized architecture can be realized and used to improve a user's privacy.

A. High-level Overview

We present a high-level overview of our proposed approach in Figure 1 and exemplarily focus our discussion on the view point of the user Alice. Before Alice can start using her DPC, she first has to gain access to infrastructure that she trusts and can provide her with the required processing and storage resources. For this purpose, we propose to leverage the idle resources on devices of close friends and family. These devices range from less powerful, embedded devices (e.g., Raspberry Pis or NAS and set-top boxes) to more powerful devices such as off-the-shelf desktop computers. Typically, these devices are located within home networks and connected to the Internet using residential access lines.

Once Alice has built-up her DPC, she can begin to run cloud services on it. We specifically target cloud services that are inherently susceptible for privacy threats. Here, our focus lies on services for a small, closed target audience (e.g., only Alice herself or selected friends). These applications can range from a calendar service offering synchronization, scheduling, and notifications up to a fully-fledged document storage service able to store several GB of data and offering functionality such as file sharing, full-text search, image editing, or multimedia streaming (note that cloud services which can be accessed by anyone are in our opinion better off with public cloud services as their information is publicly available anyways). In order to run a cloud service, Alice selects a service from a service marketplace, similar to those available for smart phones (of course, Alice could also develop her own service). The cloud service is then deployed on one or multiple of the devices in Alice's DPC. Should the cloud service require persistent storage of data, this data is distributed on the available devices.

As with public cloud services, Alice can access her services independently of her location via the Internet at any time. She does not have to care about the actual device a specific service is running on or which device stores her data. Notably, no modifications are required on the client side, hence, Alice can continue to use her web browser or other applications (e.g., an app on her smart phone) as with today's cloud services.

In our approach, each user has her own DPC spanning over resources she trusts, e.g., provided by friends and family. However, as we utilize resources based on social relationships, the DPCs of different users are likely to overlap (gray/black device in Figure 1). Here, Alice and Bob trust the same device and hence can both utilize its resources. Importantly, this does not imply that Alice and Bob have to trust each other.

In the following, we first discuss the main challenges of realizing DPCs for individual users in more detail. Based on this, we discuss how to address these challenges in the three core operations (building-up, operating, and securing) of our approach. For this, we arrange our presentation according to the typical usage pattern of our exemplary user Alice. Along the way, we demonstrate the feasibility of our approach through initial experiments where applicable.

B. Challenges

In order to realize decentralized private clouds (DPCs) on home devices, we identify the following four main challenges:
(C1) Resource Constraints: As we consider devices in home networks, we have to cope with limited storage and processing resources as well as restricting network conditions. Additionally, we utilize a wide range of different devices and hence have to account for a heterogeneity in these resources. When considering *storage resources*, a home router might provide up to few GB of storage space, while a NAS box can supply up to a few TB of disk space. Since these resources need to be shared with other users, restrictions and quotas apply (e.g., 100 MB of storage per-user VM in the Seattle testbed [14]). A similar situation holds true for *processing resources*. Cloud services, formerly executed on powerful server CPUs of public clouds, have to be operated on comparatively limited CPUs provided by desktop PCs or even embedded devices. Furthermore, DPCs face *network conditions* of residential access links which provide limited availability and capacity. Specifically, devices connected via residential access links might not be always connected to the Internet and the available bandwidths are often orders of magnitude smaller than those of data centers. To further exaggerate this issue, home networks often suffer from an asymmetry in bandwidth, i.e., a higher ratio of downlink than uplink.
(C2) Cloud Advantages: Preserving advantages of cloud computing in a DPC is a challenging task. Specifically, a user should not even notice that she is not using traditional cloud services. First of all, the *availability* of public clouds has to be achieved using decentralized devices with residential access links as sole connection to the Internet (C1). Similar to availability, the *accessibility* should not be harmed compared to public clouds. Most importantly, services in DPCs should be accessible from any device and anywhere, just as public cloud services. Hence, it should not be necessary to implement application logic or decryption on these clients. Additionally, users should be able to transparently access their services without having to care about where it is operated. From another perspective, DPCs should provide *scalability* with respect to a service's varying processing and storage demands.
(C3) Extending trust: Our DPC approach builds on social trust. However, we have to provide measures to extend this initial trust in persons to the whole system. First of all, DPCs span over the untrustworthy Internet (cf. Figure 1) and hence are susceptible to several attack vectors. Secondly, not only do users have to trust the devices their services run on, but also the device owners need to trust users to not abuse device resources. Thirdly, we have to account for multi-tenancy in resource usage. Specifically, two users that do not trust each other might end up utilizing resources on the same device (cf. §III-A). Finally, no untrusted party should have access to private information (including meta information).
(C4) Deployability: With DPCs, we aim for a drop-in replacement of today's public cloud services. Hence, *deployability of DPCs* becomes an important challenge to facilitate the seamless migration from public clouds to DPCs. Most importantly, a sufficient amount of different cloud services has to be available in order to replace today's public cloud services. Furthermore, we have to provide simple *deployment of services*. As for public cloud services, users need to be able to conveniently deploy DPC services themselves, without requiring interaction with other parties.

C. Building-up a Decentralized Private Cloud

Before Alice can start to use services in her decentralized private cloud (DPC), she first has to acquire the necessary resources as well as select and deploy services.

Acquiring Decentralized Resources. Initially, Alice has to acquire decentralized resources on which she can build her DPC. In order to amplify privacy in contrast to public clouds, she must trust the operators of these resources to respect her privacy (C3). We derive this trust relation from existing social trust (e.g., close friends or family). This trust has to hold in both ways, i.e., resource operators have to trust Alice to not misuse their resources (C3). Existing social trust also addresses the question of incentives for contributing resources [15], as resource operators can trust Alice to also provide them access to her resources (we further deepen this discussion in §V).

Acquiring Cloud Services. After Alice has acquired the necessary decentralized resources for her DPC, she wants to run services in her cloud. To achieve a seamless migration from public clouds to Alice's private cloud (C4), we envision support for existing services developed for AppScale, an open source implementation of the widely-used Google App Engine model. A wide range of cloud services that base on AppScale / Google App Engine is readily available. Additionally, many developers are familiar with the employed programming model, which guarantees a steady development of new services. Alice obtains cloud services from a service marketplace, similar to those for smart phones. Each service provides a service description and users can rate services, which allows Alice to take an informed decision. Additionally, we require source code availability, such that a service's functionality can be audited by the marketplace operator or trusted third parties [16]. To verify that AppScale can indeed be utilized to build a DPC, we performed initial experiments. Our experiments show that we can mimic the socket and storage API of AppScale and even provide extensions, e.g., transparent transport security (C3). Most notably, existing AppScale / Google App Engine services can be run in a DPC with only minor modifications (at most 8 lines of code).

Deploying Cloud Services. Once Alice has decided for a cloud service, she has to deploy it in her DPC in order to use it. In order to achieve this without additional client software (C2) and without the need for technical expertise (C4), each DPC initially comes with a special service, the *ControlCenter*. This ControlCenter provides a web interface for service management, e.g., deploying services in the cloud. It will, based on the current load, identify the device of Alice's DPC that fulfills the service's resource requirements best and deploy the service on this device. Still, resource constraints can pose a problem for services with high resource demands (C1), e.g., bandwidth intensive services may face network limitations. To address this issue, we classify services and devices according to their resource demands and availability, respectively. This enables the ControlCenter to place services on devices with sufficient resources, e.g., an application with high uplink demand on a device with good connectivity.

D. Operating a Decentralized Private Cloud

Now, Alice has successfully deployed services in her decentralized private cloud (DPC). Stepping away from Alice's perspective, we now primarily focus on how to realize the advantages of cloud computing (C2) with constrained resources (C1). As for public cloud services, these measures are mostly hidden from users but important to justify trust into the system.

Accessing Cloud Services. Alice needs to be able to access her cloud services (similar to public clouds), without having to know on which specific device they run (C2). To achieve this, we assign each service a DNS host name using Dynamic DNS under which this service can be accessed. The corresponding Dynamic DNS entry is updated whenever the service is migrated to another device or the IP address of the device running the service changes. Devices in home networks typically have only one public IP address but typically should run more than one service. Hence there is a need for demultiplexing incoming connections to individual services. As we specifically target privacy-sensitive services and communication traverses the untrustworthy Internet (C3), it is reasonable to assume that all communication will be protected using TLS. This allows us to utilize the Server Name Indication extension of TLS transport security for demultiplexing between services. Our initial experiments regarding the overhead of this approach on the TLS session establishment show a negligible overhead of 0.19 ms on a resource constrained Raspberry Pi (C1).

Service Reliability. Currently running services in a DPC may abort due to device or network failures. As in public clouds, such failures must be transparent to the user, i.e., services must recover from failures without user interaction (C2). To this end, we extend the ControlCenter in each DPC (cf. §III-C) to also monitor the status of each service based on the TLS heartbeat extension. A detected service malfunction then triggers a recovery of the service on another device. However, a sensible selection of the heartbeat frequency is crucial as it configures the trade-off between detection delay and bandwidth consumed for monitoring. Our measurements show that monitoring one application with a frequency of 1 heartbeat/s results in light network traffic overhead of 0.29 KB/s (incoming and outgoing). Hence, we can detect service failures in a DPC within at most 1 s and only a modest bandwidth overhead. An additional grace period before service recovery can avoid unnecessary overhead in case of temporary failures. Notably, also the ControlCenter can fail, hence, we operate multiple instances that monitor each other.

Data Reliability. While the above approach allows us to recover services in case of errors, this does not hold for the data stored by these services. For a transparent recovery, a service requires access to previously stored data after recovery (C2). Hence, we decouple the storage location from the processing location and provide redundant storage using a distributed hash table (DHT) running on the devices of the DPC. This allows cloud services to store and later retrieve data independently from their processing location and, hence, also after a recovery. We further increase reliability by storing data more than once to create redundancy. Additionally, using a DHT enables us to address the resource heterogeneity of devices (C1). Specifically, we dynamically adjust the range of the DHT for which a device is responsible and assign devices with lots of storage resources more than one range. This allows balancing of the storage load according to available resources.

Amplifying Data Redundancy. However, this reliability comes at the cost of additional storage space needed for creating redundancy (C1). To further increase the redundancy and hence reliability of stored data, we propose to utilize the virtually infinite storage resources of public clouds [17]. However, when using public cloud storage, we have to ensure data confidentiality and prevent unauthorized modifications (C3). Thus, we transparently apply encryption and integrity protection to data before storing it in the DHT [16]. Our performed measurements to quantify the overhead for this show a linear overhead in the size of persisted data, e.g., protecting a 100 KB data item requires only 17.47 ms on a Raspberry Pi. Nevertheless, we have to take care that untrusted parties do not learn meta information such as time and location of data access (C3). Hence, instead of using Alice's public cloud account (e.g., Dropbox or Google Drive) for storing the data, we extend the storage resources of devices by using the public cloud accounts of the owners of these devices. By this we successfully hide the origin of data stored in public clouds.

Scalability. More advanced or frequently used cloud services may require more processing resources than even powerful devices in a DPC can provide. In this case, we follow the scale-out approach and distribute one cloud service over multiple devices. This becomes especially feasible if the processing load is induced by user requests and hence request level parallelization can be employed to split a cloud service into (mostly) independent components that require only little synchronization. In contrast, if a service requires to operate on large amounts of data, we can employ a paradigm similar to MapReduce to perform operations on data as close to its storage location as possible. With respect to increasing storage demands, our DHT approach for providing reliable storage of data (see above) is inherently scalable. Alice has to simply acquire more storage resources, e.g., by adding more devices or additionally utilizing public cloud storage (see above).

E. Securing a Decentralized Private Cloud

In addition to social trust in resource owners (cf. §III-C), we have to base privacy on additional security measures (C3).

Secure Communication. Communication of devices in Alice's cloud traverses the untrusted Internet (C3) and hence must be protected. To this end, we rely on mutually-authenticated TLS channels for the communication between devices. Here, Alice relies on or operates a certificate authority to issue certificates for the access to her DPC. Hence, only devices that are authorized (and hence trusted) by Alice can participate in her DPC. Specifically, Alice deploys certificates that grant access to her DPC to all devices she trusts. Another class of certificates is used by Alice and her ControlCenter to authenticate at these devices, e.g., to deploy services.

Efficient Tenant Separation. Different services, possibly from different DPCs and hence users, can run on one device in parallel (C3, Figure 1). Hence, in order to ensure Alice's privacy, we require a strict separation of different services that run on the same device. Additionally, the device owner may wish to reserve a certain amount of resources for own local services. Thus, the owner of a device in Alice's DPC wants to ensure that Alice's services use only granted resources (C3). To address these requirements, we employ virtualization to sandbox services. First, virtualization allows us to protect a service against other services running on the same device [16]. Second, we can use virtualization to closely restrict access to resources, e.g., prevent direct, unrestricted access of services to the Internet or file system, and enforce the usage of dedicated APIs to access resources. However, one of the inevitable challenges of the devices in Alice's DPC are the potentially limited processing resources, especially when considering resource-constrained, cheap devices (C1). To account for this, we employ lightweight virtualization mechanisms to not pose additional processing overhead on the devices. More specifically, we use LXC containers, an operating-system-level virtualization, to realize the AppScale compatible PaaS environment, where only the platform APIs can be accessed (cf. §III-C), and thus avoid the overhead of full virtual machines. We performed measurements to verify that even a resource constrained Raspberry Pi (C1) is able to launch more than 30 basic cloud services (delivering a simple website) isolated in LXC containers in parallel.

Beyond Social Trust. Attacks on the devices of Alice's DPC can subvert the trust founded on social relationships. To protect against maliciously altered devices, we can optionally employ the trusted platform module (TPM) of certain modern computers. A TPM enables hardware-based security by providing cryptographic operations such as key generation, encryption, signature generation, and cryptographic hash computation. In our scenario of DPCs, we leverage TPMs to remotely attest the integrity of devices. This enables Alice to check if a specific device executes only trusted software components [18]. This allows to employ a trusted operating system to prevent the access to data during runtime in memory.

IV. Related Work

In order to break up the inherent centrality of cloud computing and thus increase privacy, related work proposes to split up the storage of data over different cloud providers. RAIN [19] aims at splitting data into very small segments which are distributed among a multitude of storage providers. In contrast, MetaStorage [17] allows to distribute data on a per file basis over several existing cloud offers and has even been extended to preserve compliance [20]. While these approaches target traditional cloud infrastructures, FriendBox [15] builds up a storage cloud over resources contributed by friends. Note, however, that this approach trades in most of the advantages of cloud computing to achieve privacy. Specifically and in contrast to our work, users cannot benefit from scalability and accessibility, as the client has to realize any application logic and decryption or reassembling of data.

From a different motivation than ours, several approaches aim to utilize idle resources of home network devices to provide cloud-like services. For example, Seattle [14], a community cloud built over commodity devices, aims at providing a learning platform. Caton et al. [21] extract trust levels from social networks to extend Seattle with trust-based resource allocation. CuteCloud [22] employs virtual machines to manage idle resources in a community cloud. Similarly, CWC [23] strives to build a cloud on processing resources of charging smart phones. Finally, ParaDrop [24] aims at realizing edge computing by offloading processing tasks from the cloud back to home gateways. Although these approaches, in contrast to our work, do not aim at preserving privacy when using arbitrary cloud services, they provide valuable input for parts of our challenges, especially with respect to realizing cloud characteristics on resource constrained devices.

Finally, several approaches aim at creating personal containers or data boxes in order to safeguard access to personal data [25]–[27]. Their core idea is to store all personal data of a user and selectively make this data available for specific purposes. DPCs as proposed in this paper could provide a solid and secure foundation for realizing such approaches.

V. Discussion and Outlook

In this paper, we propose to build decentralized private clouds (DPCs) over commodity hardware to provide the basis for moving privacy-sensitive services out of public clouds. Such an approach comes with several challenges, ranging from resource constraints over preserving the advantages of public clouds to extending trust and deployability. To overcome these challenges, we present our approach of DPCs, especially focusing on privacy-sensitive services. Our initial experiments indicate the general feasibility of our approach. Although we focus on home devices as the most challenging deployment scenario, DPCs can also be deployed on more powerful infrastructures, e.g., in corporate settings or federated clouds. This would enable enterprises to benefit from a cloud infrastructure even if legislation or customers' concerns render the utilization of traditional cloud computing challenging.

Besides the technical challenges discussed and approached in this paper, the concept of DPCs also constitutes exciting legal and economic questions. First and foremost, the question arises how law can be enforced in such a decentralized setting. In our opinion, DPCs show great potential in realizing a trade-off between valid interests involved with criminal prosecution and the people's fear of mass surveillance, especially through foreign intelligence agencies. Individual devices in a DPC can still be seized or wire-tapped if need arises, however the inherent decentrality renders the unduly monitoring of all users virtually impossible. Another interesting legal question concerns the liability of the device owner, especially if a cloud service is misused for cyber crimes such as sending SPAM emails or hacking. From a technical perspective, we aim to counter these threats by our trust model (cf. §III-C) and restricting access to resources (cf. §III-E). When focusing on economic questions, the main concern is the motivation or compensation for providing resources for others. Here, we see two promising complementary approaches. As we build on existing social trusts, users have good reason to rely on the concept of quid pro quo. Still, should users encounter an imbalance in resource-usage and want to be compensated for this, we propose to use micro-payment schemes such as Bitcoin to reimburse resource providers.

To further evaluate and hence substantiate the feasibility of our approach, we are currently developing a full prototype of DPCs. We plan to use this prototype to perform detailed measurements of the involved processing overheads. Additionally, we are working on a simulation model to study the impact of network characteristics (e.g., bandwidth and latency) on DPCs.

To conclude, with this paper we present an approach for retaining privacy when using cloud services by moving them from public clouds to decentralized private clouds. Thereby we break up the inherent centrality and non-transparency of cloud computing without the need to give up its advantages.

Acknowledgments

This work has received funding from the European Union's Horizon 2020 research and innovation program 2014-2018 under grant agreement No. 644866 (SSICLOPS) and the Excellence Initiative of the German federal and state governments. It reflects only the authors' views and the European Commission is not responsible for any use that may be made of the information it contains.

References

[1] B. Gellman, "Edward Snowden, after months of NSA revelations, says his mission's accomplished," *The Washington Post*, December 24, 2013.
[2] M. Theoharidou *et al.*, "Privacy risk, security, accountability in the cloud," in *IEEE CloudCom*, 2013.
[3] I. Ion *et al.*, "Home is safer than the cloud!: Privacy concerns for consumer cloud storage," in *SOUPS*, 2011.
[4] S. Pearson and A. Benameur, "Privacy, security and trust issues arising from cloud computing," in *IEEE CloudCom*, 2010.
[5] M. Henze *et al.*, "The cloud needs cross-layer data handling annotations," in *IEEE SPW*, 2013.
[6] T.-M. Pasquier and J. Powles, "Expressing and enforcing location requirements in the cloud using information flow control," in *CLaw Workshop*, 2015.
[7] M. Henze *et al.*, "User-driven Privacy Enforcement for Cloud-based Services in the Internet of Things," in *FiCloud*, 2014.
[8] M. Paul *et al.*, "A possible solution for privacy preserving cloud data storage," in *CLaw Workshop*, 2015.
[9] A. Agarwal, "Security update and new features," July 31, 2012. [Online]. Available: http://blog.dropbox.com/security-update-new-features/
[10] M. Henze *et al.*, "A comprehensive approach to privacy in the cloud-based Internet of Things," *Future Generation Computer Systems*, 2015.
[11] Cloud Security Alliance, "CSA survey results: Government access to information," 2013.
[12] D. Castro, "How much will PRISM cost the U.S. cloud computing industry?" The Information Technology & Innovation Foundation, 2013.
[13] M. Henze *et al.*, "Towards Data Handling Requirements-aware Cloud Computing," in *IEEE CloudCom*, 2013.
[14] J. Cappos *et al.*, "Seattle: A platform for educational cloud computing," in *ACM SIGCSE*, 2009.
[15] R. Gracia-Tinedo *et al.*, "FriendBox: A hybrid F2F personal storage application," in *IEEE CLOUD*, 2012.
[16] M. Henze *et al.*, "Maintaining User Control While Storing and Processing Sensor Data in the Cloud," *IJGHPC*, vol. 5, no. 4, 2013.
[17] D. Bermbach *et al.*, "MetaStorage: A federated cloud storage system to manage consistency-latency tradeoffs," in *IEEE CLOUD*, 2011.
[18] J. M. McCune *et al.*, "Flicker: An execution infrastructure for TCB minimization," in *ACM SIGOPS/EuroSys*, 2008.
[19] M. G. Jaatun *et al.*, "The design of a redundant array of independent net-storages for improved confidentiality in cloud computing," *Journal of Cloud Computing*, vol. 1, no. 1, 2012.
[20] T. Wüchner *et al.*, "Compliance-preserving cloud storage federation based on data-driven usage control," in *IEEE CloudCom*, 2013.
[21] S. Caton *et al.*, "A social compute cloud: Allocating and sharing infrastructure resources via social networks," *IEEE T SERV COMPUT*, vol. 7, no. 3, 2014.
[22] D. Che *et al.*, "CuteCloud: Putting "credit union" cloud computing into practice," in *ACM RACS*, 2012.
[23] M. Y. Arslan *et al.*, "Computing while charging: Building a distributed computing infrastructure using smartphones," in *ACM CoNEXT*, 2012.
[24] D. Willis *et al.*, "ParaDrop: a multi-tenant platform to dynamically install third party services on wireless gateways," in *ACM MobiArch*, 2014.
[25] A. Chaudhry *et al.*, "Personal Data: Thinking Inside the Box," *Aarhus Series on Human Centered Computing*, vol. 1, no. 1, 2015.
[26] R. Mortier *et al.*, "The Personal Container, or Your Life in Bits," in *Digital Futures*, 2010.
[27] Y.-A. de Montjoye *et al.*, "openPDS: Protecting the Privacy of Metadata through SafeAnswers," *PLoS ONE*, vol. 9, no. 7, 2014.