SHARDIS: A Privacy-Enhanced Discovery Service for RFID-Based Product Information

Benjamin Fabian, *Member, IEEE*, Tatiana Ermakova, and Cristian Müller

***Abstract*—The EPCglobal Network is an emerging global information architecture for supporting Radio-Frequency Identification (RFID) in supply chains. Discovery services for the EPCglobal Network are distributed services that serve the following pivotal lookup function: Given an identifier for a real-world object, e.g., an Electronic Product Code (EPC) stored on an RFID tag, they return a list of Internet addresses of services that offer additional information about the object. Since a client's information interests in the EPCglobal Network can be used to create inventory lists and profiles of his physical surroundings, as well as be used for business intelligence on the flow of goods in corporate applications, protecting client privacy becomes crucial. In particular, privacy mechanisms should by design be integrated into discovery services where the client's information interests could be analyzed by many potential adversaries. This paper introduces SHARDIS, a privacy-enhanced discovery service for RFID information based on the peer-to-peer paradigm. The idea is to enhance confidentiality of the client's query against profiling by cryptographically hashing the search EPC and by splitting and distributing the service addresses of interest. Furthermore, a probabilistic analysis of the privacy benefits of SHARDIS is presented. SHARDIS was implemented using the global research platform PlanetLab. Several performance experiments show its practical feasibility for many application areas.**

***Index Terms*—Discovery service, privacy, RFID, supply chain.**

I. Introduction

RADIO-FREQUENCY IDENTIFICATION (RFID) is about to be deployed in supply chains worldwide within the next few years. The main incentive for RFID deployment is to improve efficiency and transparency of flows of goods by enabling their real-time tracking, analysis, and control. On the other hand, RFID is a core technology for *Ubiquitous Computing* and *Ambient Intelligence*, where smart, context-aware environments identify all objects they contain and adapt to explicit or anticipated user needs. In addition to the anticipated ubiquity of RFID tags and readers, there is another important factor that facilitates these applications: The standardization of a global numbering scheme for physical objects, the Electronic Product Code (EPC) that is stored on RFID tags. The EPC standard actually comprises an entire family of data structures. Its SGTIN-96 variant (see Fig. 1), for example, includes a company prefix that is a unique identifier of the item manufacturer who can assign item reference numbers to classes of objects he produces. Within the same class, similar objects can be distinguished by their serial number—this is a fundamental extension of the conventional barcode.

Header	Filter Value	Partition	Company Prefix (EPC Manager)	Item Reference (Object Class)	Serial Number
8 Bits	3 Bits	3 Bits	20-40 Bits	4-24 Bits	38 Bits
			Total Length: 44 Bits		

Fig. 1. Electronic product code (EPC), SGTIN-96.

The intention of influential companies in several industries forming the international consortium EPCglobal [1] is to add RFID tags to as many objects that leave production as possible, and even to permanently integrate tags into clothes and devices. These tags store EPCs, which are unique keys to retrieve additional information from a large distributed network of databases around the globe, the EPCglobal Network (EPCN) [2]. According to their vision, data about items should not be stored on the RFID tags, but retrieved from the EPCN via the Internet by using the EPC as a search key. This aims to facilitate information exchange about goods in global supply chain networks, leading to an expected larger increase of transparency and efficiency than would be provided by isolated RFID solutions [3], [4]. In an extension of this initial application scope, the EPCN could also serve as a backbone for Ubiquitous Computing and the Internet of Things (IOT), enabling smart environments to easily recognize and identify objects, and retrieve information from the Internet to facilitate their adaptive functionality.

Discovery Services (DS) for the EPCN are distributed systems that serve the following fundamental lookup function: Given an identifier for a real-world object, e.g., an EPC, they return a list of Internet addresses of EPC Information Services (EPCIS) that offer additional information about the object. Without DS acting as a broker between items and their information sources, the EPCN could not achieve the flexibility and global scalability necessary to live up to its vision. Currently, several proposals for DS are under development [5], but no official standard has been finalized. Only the related Object Naming Service (ONS) [6] has already been specified. ONS has severe security drawbacks in its architecture and design, and also involves problems of international control over a future critical business infrastructure [7], [8]. This poses similar challenges for DS research.

Currently, no sufficient mechanisms to address the privacy requirements of its various information stakeholders are provided by the EPCN architecture. If no care is taken, the EPCN could

Manuscript received March 09, 2011; revised May 16, 2011; accepted July 04, 2011. This research was supported in part by the German Federal Ministry of Education and Research under Grant 01IA08001E as part of the Aletheia Project. Paper no. TII-11-097.

The authors are with the Institute of Information Systems, Humboldt-Universität zu Berlin, 10178 Berlin, Germany (e-mail: bfabian@wiwi.hu-berlin.de).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2011.2166783

enhance "transparency" of global good flows and inventory not only for authorized business partners, but also for adversarial business intelligence and personal profiling, extending many of the attacks possible for close-range RFID (see [9] and [10]) to a next level of global exposure [8]. Recently, the European Data Protection Supervisor demanded that privacy should *by design* be integrated into emerging IOT and RFID technologies [11].

In this spirit, this paper focuses on privacy concerns for the discovery phase of RFID information services. It presents SHARDIS, a DS architecture based on Distributed Hash Tables (DHT), and its implementation on the research platform PlanetLab [12]. SHARDIS will be deployed as an infrastructure peer-to-peer (P2P) network, offering high scalability and performance. Furthermore, it provides new mechanisms for protecting client privacy from eavesdroppers even if *no earlier key distribution* to the clients can be arranged, mitigating an important practical problem of earlier work.

This paper is structured as follows. Related work is given in Section II. The architecture of the SHARDIS discovery service is described in Section III. Section IV discusses privacy benefits and security aspects. The implementation and experimental performance evaluation are presented in Section V.

II. Related Work

Many protocols exist that provide device and service discovery in small-scale and often local networks. For example, Universal Description, Discovery, and Integration (UDDI) is a Web service standard for service description and discovery, but is currently mostly used within a single organization, lacking an established global infrastructure, and would not scale for looking up vast numbers of object identifiers in the IOT [13].

An early proposal for global information discovery in the EPCglobal architecture framework [2] was the ONS [6], which was designed by EPCglobal on the base of the Domain Name System (DNS). ONS provides discovery of manufacturer EPC Information Service (EPCIS) for a given class-level EPC. Though serial-level discovery of arbitrary EPCIS would be in theory feasible with ONS, it is explicitly not considered. Such a general lookup functionality of serial-level information from multiple providers in the EPCglobal Network will be provided by so-called Discovery Services (DS).

At present, the final scope and requirements for Discovery Services are not clear. Proposals range from an extended serial-level ONS to more complex middleware layers, for a comparison of important proposals see [5]. Notable earlier designs are based on a classical client-server paradigm and have been proposed by the BRIDGE project [14], [15], the company Afilias, and an IETF working group on Extensible Supply-chain Discovery Services (ESDS) [13], [16]. Similar to some approaches discussed in BRIDGE, [17] proposes to integrate actual data and their aggregation into the discovery-service layer, in contrast to the separation of discovery and data access originally envisioned by EPCglobal.

In order to provide decentralization and robustness, several approaches based on the P2P paradigm and especially DHTs [18] have been proposed for full DNS, ONS, and DS. For full DNS, proposals for P2P architectures include [19]–[22], many of them based on the Chord DHT [23]. Out of those, CoDoNS [20], for example, combines the decentralization, scalability, simple administration, and robustness of a DHT with proactive caching to reduce lookup latency, and has been deployed and tested on the Internet. Furthermore, cooperative DNS lookups [24], [25] and also hybrid architectures of DNS and DHT have been discussed in the literature [21], [26].

For ONS and DS, multiple P2P designs have been proposed. OIDA [27], based on the Bamboo DHT [28], especially considers security and privacy aspects and has been implemented and tested on PlanetLab [29]. A distributed ePedigree architecture that is also providing ONS functionality by a DHT was presented in [30]. A global supply chain information architecture based on the FreePastry DHT [31], [32] was simulated with up to 20,000 nodes in [33]. Similar architectures are described in [34] and [35]. In [36], the potential of decentralizing ONS and DS by DHTs was also investigated and, based on theoretical analysis and simulation, an efficiency comparison of centralized vs. decentralized architectures for the EPCN was conducted; in nested-package scenarios, network capacity improved with the decentralized solution.

Many of these works indicate the feasibility and high scalability benefits of P2P solutions. However, only few of them consider security aspects, and with the exception of [27] and the corresponding key distribution problem, none of these proposals has taken privacy challenges into account.

Beyond the scope of this paper is an in-depth comparison of Shamir's secret-sharing scheme with others, among them Blakley's scheme [37] and the Information Dispersal Algorithm (IDA) [38]. An interesting approach for providing share authenticity in secret-sharing is presented in [39] and [40]. Such verifiable secret-sharing schemes directly establish the authenticity of the shares released by dealer and participants. Though non-interactive proposals do exist [41], in many variants the possibility of live interaction between participants is assumed, which cannot be easily transferred to our application scenario where share storage and retrieval from the DHT are asynchronous and additional communication should be minimized to reduce privacy risks. In SHARDIS, we provide end-to-end document authenticity and data origin authentication by digital signatures.

In the application area of RFID and the IOT, [42] uses a secret-sharing approach to distribute keys over multiple RFID tags. To create a highly available and secure DHT-based storage of private data, [43] develops a scheme dividing data into multiple blocks with the use of secret-sharing.

Secret-sharing in conjunction with DHTs was proposed for another application area by the Vanish project [44] in order to protect the confidentiality of archived data, e.g., messages posted on a web forum. The authors propose to encrypt a user's data with a random encryption key, then to split this key into shares, and to store the shares into a DHT with fluctuating membership. After some time, due to churn or deletion of old data, the key shares become unavailable, resulting in the key being permanently lost and the data eventually vanishing. In contrast to their work, the goal of SHARDIS is to maintain the data as long as they are needed, supported by an infrastructure DHT with expected low churn rates. In a response to Vanish, [45] presented attacks targeting the integrated data replication mechanism of the underlying DHT or by using Sybil attacks. Both attack paths are not directly applicable to SHARDIS, since it does

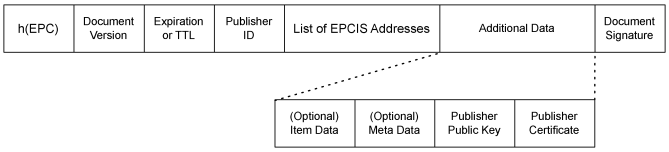

Fig. 2. SHARDIS address document.

not use the replication mechanism of the underlying DHT. Furthermore, SHARDIS as a cooperative industrial infrastructure network will in practice be based on cryptographically secured node identifiers involving a certification authority, rendering the creation of necessary amounts of sybils at least very difficult (Section IV-D).

III. SHARDIS Architecture

A. Requirements

SHARDIS is designed to satisfy core functional requirements for a DS S in the EPCN (following [5]). S should be flexible in its support for different Object Identifier (OID) schemes. At a minimum, it must be compatible with the EPC identification framework [46]. If the OID is structured into a class-level and serial-level part, S should be able to deliver address data for partial OIDs at the class-level, such as the current ONS does for partial SGTIN EPCs consisting of EPC Manager and Object Class. But more importantly, S should also be able to work with *fully serialized* OIDs, for example, a complete SGTIN EPC (Fig. 1) consisting of the following main fields: Company Prefix (representing usually the manufacturer), Item Reference (indicating the class of an object), and a Serial Number that enables globally unique identification and tracking of individual object instances.

Multiple independent publishers should be able to provide data for an OID by storing corresponding address data in S. Those documents include addresses of servers for EPC Information Services (EPCIS), which provide data about the objects carrying those OIDs. Additional data about the object, the publisher, and the document itself should be included, while a digital signature of the document will provide means to verify its authenticity (Fig. 2).

Furthermore, SHARDIS aims to satisfy the following important nonfunctional requirements: scalability, robustness, data integrity, and client privacy *without* key predistribution (contrasting [27]).

B. Organizational Aspects

From an organizational perspective, SHARDIS will be deployed as a global business infrastructure with controlled membership. Node providers are assumed to be companies, bound by contracts to (at least some of) its partners and to a global management coordinator such as the consortium EPCglobal, which is already adopting a similar role for the EPCN. This approach creates incentives to maintain the common infrastructure by providing nodes and bandwidth. Furthermore, it makes a corresponding certification authority (CA) infrastructure for information providers and nodes possible where the central management coordinator issues long-standing certificates to participating companies that authenticate their public keys. This is useful for providing end-to-end data authenticity via digital signatures, and for introducing further security measures on the node level (see Section IV-D). In addition, the coordinator can monitor the status of the DHT by periodic node uptime measurements, and could distribute software updates to the participants. Since the coordinator does not participate in the actual discovery service functionality, he presents no communication bottleneck in daily SHARDIS operations.

C. P2P Architecture for Scalability and Robustness

Distributed Hash Tables are P2P systems that offer a lookup functionality analogous to a hash table, but in a distributed and decentralized fashion, involving multiple computers without central control [18]. They offer a simple lookup and storage interface based on a one-to-one correspondence between data items and keys. The underlying distributed algorithms determine which nodes are responsible for storing the data by organizing keys and nodes in a logical *overlay network*, which is in general independent of the physical or IP network topology on lower layers using concepts such as consistent hashing [23], [47] with only a small amount of local information about the whole system. Consistent hashing balances data items to nodes in a roughly uniform way, and allows for node joining and leaving, without the need for major redistribution of keys and data in the running system.

Most DHTs resolve lookups in $O(\log N)$ hops through the overlay network, where N is the number nodes in the DHT, which offers excellent scalability. This scalability is enhanced by the fact that the routing table size and amount of state information stored at any particular node also scales with $O(\log N)$, which means that every node just needs to know a very small part of the whole overlay graph. DHTs also offer high robustness to faults, avoid single points of failures (e.g., they have no single root like ONS or DNS), and distribute responsibility and load among participants in a systematic way by means of a prearranged topological overlay structure [18]. For SHARDIS, FreePastry [32] was chosen as the underlying DHT, an open-source implementation of the classical Pastry [31] (see Section V).

Fig. 3 shows a view on the SHARDIS system architecture for publishing and retrieval. The left hand of the figure depicts a physical supply chain network that can be highly complex and dynamic in practice. An RFID-tagged product passes through several stations of the supply chain network, which could extend to a consumer's RFID-equipped smart home environment and also include reverse logistics back to distributor and manufacturer, e.g., for recycling. Every station records and provides data about the item during processing, which is indexed by the EPC. These data are made available to information clients by web services, the EPCIS (right-hand side of the figure, step 3). In order to support high flexibility, the EPCglobal approach proposes DS to locate all EPC Information Services relevant for a particular EPC. The middle of the figure depicts the main SHARDIS operation, the provisioning of service address documents via distributed shares (step 1), and the corresponding queries (step 2) of a client in order to discover those information services.

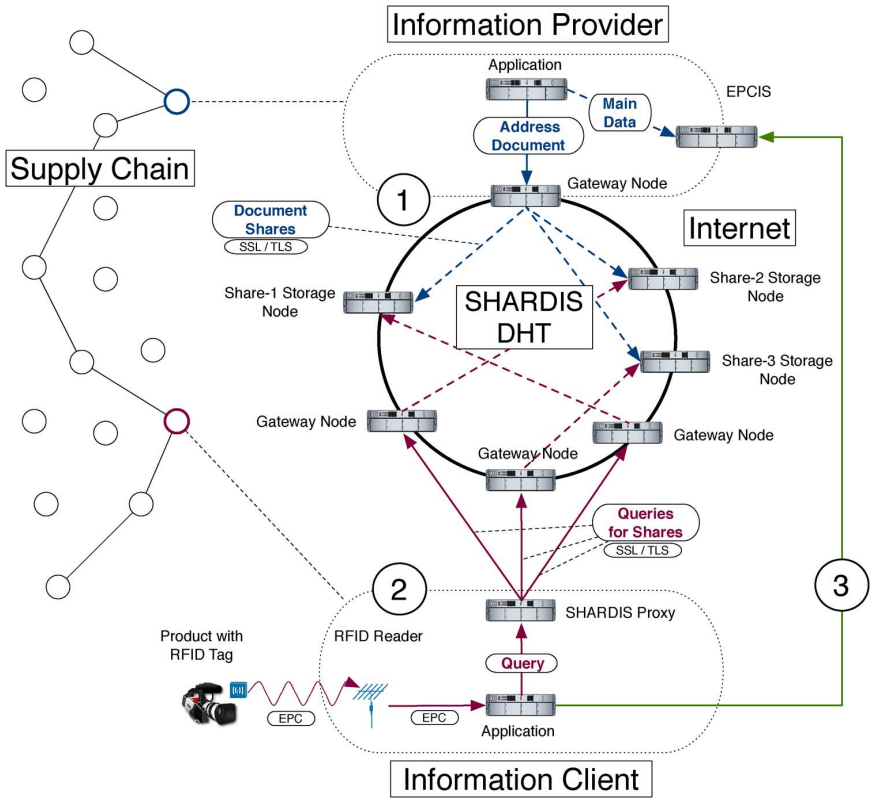

Fig. 3. SHARDIS architecture.

D. SHARDIS Publishing Procedure

1) Document Split: Before publishing, the information provider partitions the SHARDIS address document D (Fig. 2) into multiple shares using Shamir's secret-sharing scheme [48], using a finite field. The advantages and principles of working in a finite field are discussed in [49, pp. 96] and [50]. More precisely, D is split into n shares $(n > k)$ in such a way that knowledge of any $t := k + 1$ or more shares makes D easily computable, but the knowledge of only k or fewer shares leaves D completely undetermined. Important notations are depicted in Table I.

In order to support flexible, unrestricted document lengths and to reduce overhead in case of small documents, secret-sharing is applied to each byte of D. Each document byte D_j $(j = 1,\ldots,l)$ is represented as an element of the finite field $\mathbb{Z}_\Pi$. The publisher chooses l polynomials $f_j(x)$, one for each byte, by setting each $a_{0j} := D_j$ $(j = 1,\ldots,l)$, and choosing all other coefficients a_{ij} randomly (but nonzero). Furthermore, the publisher selects $x_1, x_2, \ldots, x_n$ from $\mathbb{Z}_\Pi$ at random (pairwise distinct). Each y_{ij} is then calculated as $y_{ij} = f_j(x_i) = a_{0j} + a_{1j}x_i^1 + \ldots + a_{kj}x_i^k \bmod \Pi$. The document shares for D are therefore obtained (for $i = 1,\ldots,n$) as vectors $s_i = (x_i, y_{i1}, \ldots, y_{ij}, \ldots, y_{il})$, by calculating

$$\begin{bmatrix} y_{11} & y_{12} & \ldots & y_{1l} \\ \vdots & \vdots & \vdots & \vdots \\ y_{i1} & y_{i2} & \ldots & y_{il} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \ldots & y_{nl} \end{bmatrix} = \begin{bmatrix} 1 & x_1^1 & \ldots & x_1^k \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_i^1 & \ldots & x_i^k \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n^1 & \ldots & x_n^k \end{bmatrix} \begin{bmatrix} a_{01} & \ldots & a_{0l} \\ a_{11} & \ldots & a_{1l} \\ \vdots & \vdots & \vdots \\ a_{k1} & \ldots & a_{kl} \end{bmatrix}.$$

TABLE I
MATHEMATICAL NOTATIONS

Notation	Interpretation
e	EPC of a certain product of interest.
D	Address document for e.
l	Length of D in bytes.
D_j	j-th byte of D, represented in $\mathbb{Z}_\Pi$, $j = 1,\ldots,l$.
k	Degree of polynomial f_j.
t	Threshold $t := k + 1$ to recover D_j.
n	Number of shares to be generated.
Π	Fixed public prime, chosen as $\Pi := 257$ in SHARDIS.
$\mathbb{Z}_\Pi$	Finite field used for all computations.
f_j	Polynomial to split D_j, randomly chosen.
p_j	Interpolated polynomial to reconstruct D_j, $p_j = f_j$.
a_{ij}	i-th coefficient of polynomial f_j.
a_{0j}	0-th coefficient of polynomial f_j, $a_{0j} := D_j$.
x_i	The x-value of share s_i of D (constant for all y_{ij}).
y_{ij}	The y-value $y_{ij} = f_j(x_i)$, obtained from D_j for share s_i.
s_i	i-th share of D, $s_i := (x_i, f_1(x_i), \ldots, f_l(x_i))$.
Δ_t	Time interval indicating the share lifetime.
h	Cryptographic hash function specific to the DHT.
$h_{(e,i)}$	$h_{(e,i)} := h(e \circ i \circ \Delta_t)$.
N	Number of nodes in the DHT.
c	Average number of overlay IDs per node.

Here, n, k, and Π are assumed to be public fixed parameters of SHARDIS. Π is set to 257, the prime nearest to the largest integer representable by a byte, for a motivation of this choice, see [50, p. 107].

We give a conceptual example. Suppose, the document D to be published is 3 bytes long and can be represented as the triple (5,2,6), each byte an element of $\mathbb{Z}_7$. The publisher wishes to divide D into $n = 3$ shares, and $t = 3 = k + 1$ shares should

be necessary to reassemble D. The publisher generates random integers for coefficients a_1 and a_2 as well as for x-values, e.g., $x_1 = 2, x_2 = 3, x_3 = 4$. Conceptually, he applies secret sharing to each byte of the document separately as follows:

$$\begin{bmatrix} 0 & 2 & 4 \\ 3 & 3 & 3 \\ 5 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 4 \\ 1 & 3 & 2 \\ 1 & 4 & 2 \end{bmatrix} \begin{bmatrix} 5 & 2 & 6 \\ 2 & 4 & 6 \\ 3 & 5 & 0 \end{bmatrix}.$$

The publisher combines each x_i with the corresponding y_{ij} values to create the final document shares: $s_1 = (2, 0, 2, 4)$, $s_2 = (3, 3, 3, 3)$, $s_3 = (4, 5, 0, 2)$. In order to reconstruct D, the client needs all three document shares in our example. From those, she reconstructs each document byte D_j separately from x_i and y_{ij} at corresponding positions j ($i \in \{1, 2, 3\}$) by polynomial interpolation, as discussed in Section III-E2 below. In the rest of the paper, the term *share* always refers to the full *document share*.

2) Storage of Document Shares: The information provider stores each share at a different overlay key of the DHT. The procedure to calculate this overlay key is publicly known, therefore, any client in possession of the EPC e can use the same strategy to receive the shares. On the other hand, an adversary not knowing e should get no information about it. SHARDIS adopts the common procedure to generate an overlay key from a data identifier by applying a cryptographic hash function (CHF) such as SHA-1 [51]. The generation of overlay keys for every document share using a CHF h works in the following way.

For every share s_i, $i \in \{1, \ldots, n\}$, the overlay key $h_{(e,i)} := h(e \circ i \circ \Delta_t)$ is generated by applying the CHF to the concatenation ($\circ$) of the EPC e, share number i, and the time interval during which the share should be available. In general, the interval Δ_t between two republications can be freely chosen, but it has to suffice two conditions: First, it has to be known to the client to enable her to find the shares. Second, the interval cannot be too small because all publishers need enough time to republish their shares. The interval Δ_t is integrated into the string used by the CHF h, and is proposed to be an (English) month name with the corresponding year as a tradeoff between possible profile length and republication.

Each share s_i is stored in the DHT by contacting a DHT node acting as gateway, preferably belonging to the same organization. Gateways are responsible for directly transporting the shares to the corresponding storage nodes. As a result of this process, every share s_i is stored at the DHT node responsible for the part of the overlay key space, which includes $h_{(e,i)}$.

With each share, a time-to-live value (TTL) is published that states the time interval in which the share should be accessible by clients, in general the full interval Δ_t. This TTL leaves the DHT member with an opportunity to clear its storage from obsolete entries. Because a share is only valid for a period of time, this could lead to "publishing peaks" before a new time interval starts. This can be prevented by publishing shares for future time periods in a timely manner. If only a few entries for a time period exist, the TTL could be used by an adversary to associate shares, but the privacy enhancement offered by SHARDIS is primarily based on the fact that it is difficult for an adversary to gather enough shares to reconstruct the document (see Section IV). As a convention, the share will stay available for a reasonable amount of time after the TTL interval ends, in order to prevent lookup problems related to time synchronization.

Multiple DHT gateways can be used to store shares in parallel, which is especially useful to increase the performance when publishing many address documents at once. As a tradeoff, the trustworthiness of the additional gateways with respect to client privacy needs to be considered, since they could serve as harvesting points for shares of that particular publisher during publishing.

A further important feature is that the DHT is able to store multiple data items for every particular overlay identifier. First, this is desirable to cover the very unlikely case of a collision when applying the CHF, when shares of different EPCs would be mapped to the same overlay ID. Second, more importantly, this feature allows for multiple information providers who publish different address documents for the same EPC (Section III-E3 below).

E. SHARDIS Lookup Procedure

1) Retrieval of Document Shares: The retrieval of a document from SHARDIS by a client with knowledge of EPC e works in principle similar to publishing. The client calculates $h_{(e,i)}$, $i \in \{1, \ldots, n\}$, and retrieves the shares s_i from the DHT.

Usually in DHT designs, the client contacts a node of the DHT and queries for the overlay ID (in our case $h_{(e,i)}$) of the data item she is interested in. This request is either recursively routed to the storage node responsible for $h_{(e,i)}$, or the client is informed of the next hop closer to the target node and reaches the target by iteratively contacting the next hop. In both cases, the privacy-relevant information interest e of the client is obfuscated by the CHF to $h_{(e,i)}$. Since a CHF is very hard to invert, e cannot be inferred from $h_{(e,i)}$ directly. But at present it is not clear how large the space of all possible identifiers in the EPCN and IOT will be; DS may be used for EPCs, comprising several other frameworks than SGTIN-96 only [46], but also for several other numbering frameworks, possibly also for IPv6. If the space of all possible identifiers is too small, dictionary attacks may be feasible where adversaries generate lookup tables for all possible input values to the DHT. In addition to including the time interval Δ_t, shared random salts are preferable as input to the CHF, but would create a distribution challenge. Random salts could be distributed to clients by storing them on the RFID tags [27], which would restrict the usage of SHARDIS to clients who have or had access to the actual tag. This approach would be viable for many application scenarios. But in order to provide maximum flexibility, the main countermeasure in SHARDIS is to reduce the number of DHT nodes that would be able to reconstruct the client's obfuscated information interest e from $h_{(e,i)}$. In order to achieve this goal, the usual DHT retrieval procedures are changed in the following way.

The SHARDIS lookup process consists of two steps: (1a) identify the nodes holding the shares of a document without "leaking" $h_{(e,i)}$; then (1b) retrieve the shares from the respective nodes directly. In SHARDIS these two steps are implemented in a service layer, thus enabling external clients who are not part of the DHT to conduct this lookup.

During request (1a), the client does not use the complete hash value $h_{(e,i)}$ of the share she is interested in, but uses two node identifiers λ and ρ derived from $h_{(e,i)}$ instead (see below and

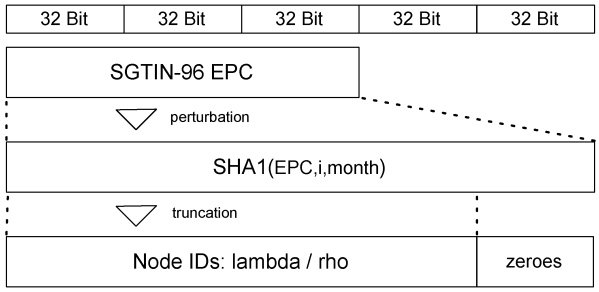

Fig. 4. From EPC to SHARDIS node ID.

$share\_ids \leftarrow \{h(e \circ i \circ \Delta_t) : i \in \{1, \ldots, n\}\}$
forall $h_{(e,i)} \in share\_ids$
  $\lambda \leftarrow$ setzero$(h_{(e,i)}, 129, 160)$
  $\rho \leftarrow \lambda + 0x100000000$
  $N_\lambda \leftarrow$ lookupnode(λ)
  $N_\rho \leftarrow$ lookupnode(ρ)
  if$(dist(N_\lambda, h_{(e,i)}) > dist(N_\rho, h_{(e,i)}))$
    $shareset \leftarrow$ retrieve$(h_{(e,i)}, N_\rho)$
  else
    $shareset \leftarrow$ retrieve$(h_{(e,i)}, N_\lambda)$
end
return $shareset$

Fig. 5. SHARDIS retrieval algorithm.

Fig. 4). From the replies of this first lookup, the client calculates the node responsible for $h_{(e,i)}$ and the share s_i. This procedure and the lookup process are explained in more detail in the following and in Fig. 5. For identifying the node that is responsible for $h_{(e,i)}$, all nodes of the DHT have to comply with an identification convention. In FreePastry, every possible 160-bit overlay identifier (the range of SHA-1) can be used as node identifier. In SHARDIS, however, only a subset of the possible overlay identifiers will be used as node identifiers. The most significant 128 bits identify a node, while the least significant 32 bits of a node identifier are set to zero (see Fig. 4). This leaves a range of at least 2^{32} identifiers between two nodes, which ensures that one of the nodes responsible for λ and ρ is also responsible for $h_{(e,i)}$.

In step (1a), the client first computes the two node identifiers λ and ρ closest to the hash value $h_{(e,i)}$. The client asks a node of the DHT to identify the nodes N_λ and N_ρ responsible for these two node identifiers—since nodes are in practice only sparsely distributed in the overlay ID space, it is likely that each single physical node is responsible for multiple node IDs, and therefore, in general, $N_\lambda \neq \lambda$ holds, as well as $N_\rho \neq \rho$. Any member of the DHT could be used as contact partner for this first step without privacy loss, since neither e nor the full hash $h_{(e,i)}$ are used.

For step (1b), the client calculates the node closer to $h_{(e,i)}$ and queries it directly. Since minimal absolute distance between overlay and node IDs determines storage nodes in FreePastry [31], this node is responsible for storing the share s_i. The lookup algorithm is depicted in Fig. 5. The function $\texttt{setzero}(h_{(e,i)}, v, w)$ sets the bits from v to w of $h_{(e,i)}$ to zero. $\texttt{lookupnode}(id)$ returns the node responsible for the overlay ID id, whereas $\texttt{retrieve}(h_{(e,i)}, N_\tau)$ returns the share for overlay ID $h_{(e,i)}$ directly from the storage node N_τ. This query process could be stopped once at least t shares (arbitrarily chosen out of n) have been retrieved successfully. A

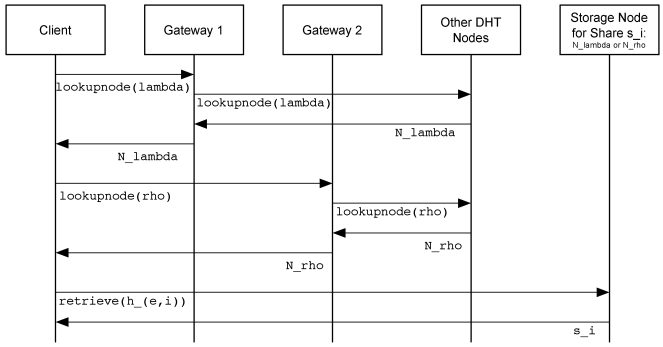

Fig. 6. SHARDIS retrieval communication (for each Share s_i).

simplified example of the network communication for a lookup in SHARDIS is given in Fig. 6.

Step (1b) involves a potential privacy vulnerability: If the contacted node N_τ is under control of the profiling adversary, he will be able to discern the IP address of the client and the full value $h_{(e,i)}$ the client queried for. However, the one-way property of the CHF prevents the adversary from inferring the EPC e. Furthermore, an inference of e from the clear text document D is prevented if the adversary does not possess enough shares to reconstruct D.

2) Document Reconstruction: Once a client has successfully retrieved at least $t = k + 1$ distinct shares, she can start the document reconstruction procedure where all operations are again conducted in the finite field $\mathbb{Z}_\Pi$. In order to keep the notation simple, we assume without loss of generality that the first t shares have been retrieved. The reconstruction of the document D from $s_i = (x_i, y_{i1}, \ldots, y_{ij}, \ldots, y_{il})$ with $i = 1, \ldots, t$ requires the recovery of each byte D_j from the corresponding (x_i, y_{ij}). For each such byte, the following considerations apply. Given any subset of $t = k + 1$ points $\{(x_1, y_{1j}), \ldots, (x_{k+1}, y_{(k+1)j})\}$ with pairwise distinct x-values, it can be shown that there is a unique polynomial p_j of degree less than or equal to k, such that $p_j(x_i) = y_{ij}$, for all $i \in [1, k + 1]$ (see [52, pp. 159–162]). In order to compute $a_{0j} = D_j$, the *Lagrange* form of the interpolating polynomial is typically applied in secret-sharing literature (see [53, pp. 245] and [54]).

For SHARDIS, another approach was adopted: the interpolation polynomial introduced by *Newton* (see [55, pp. 175–176]). Both Lagrange and Newton algorithms for interpolation based on t points have complexity of $O(t^2)$ (see [56, p. 41]). Experimental comparison (not included in this paper for reasons of space) showed that the execution time may in general be reduced by using the Newton interpolating polynomial instead of the established Lagrange approach.

3) Reconstructing Multiple Documents: SHARDIS allows multiple information providers to publish documents for the same EPC. This entails multiple shares matching one overlay ID $h_{(e,i)}$. In such a case, the client has to recombine the share sets that belong to the *same* documents. As follows from the computational results in Section V-B2, reconstructing multiple documents by simply testing all possible combinations of shares is not an option if speed is crucial and a large number of documents have been published for the same EPC.

In order to reduce the burden on the client, a reconstruction hint can be hidden inside the shares. In SHARDIS, x-ordinates

of shares are chosen using the formula $x_i = x_{i-1} + d$, where d is document specific. This distance d is chosen by the information provider during the publishing process and is the same for all shares of a given document. Two different documents have two different share distances, with high probability.

If multiple documents have been published for a single EPC, the results to a lookup and retrieval from the DHT are t share sets that contain one share per document. Each set corresponds to one hash value $h(e \circ i \circ \Delta_t)$. These sets can be ordered by the value of i that was used to retrieve them. This order can be used to calculate d from any combination of share sets. E.g., if share sets for i and $i+2$ are evaluated, d can be calculated as the difference between an x-ordinate of a share in the set $i+2$ and the x-ordinate of a share in the set i, divided by 2: $d = (x_{i+2} - x_i)/2$. After retrieval, the client sorts the sets of shares and takes the first two sets. From these two sets, she calculates all possible distances and tries to find shares that match the distances in the remaining sets.

This will drastically reduce the search space because shares that were not created for the same document are unlikely to match in distance. In the best case, this lowers the number of reconstruction trials to the actual number of documents that were encoded by the shares. Hints are potentially also provided to an adversary, but the privacy enhancement offered by SHARDIS is primarily based on the fact that it is difficult for an adversary to gather enough shares to reconstruct the document, not on the hardness of recombining the correct shares an adversary is already possessing (Section V-B2).

IV. Security and Privacy Analysis

A. Data Integrity

Data integrity and authenticity in SHARDIS is provided by digital signatures on the published documents D. This procedure applies established and classical protection measures based on public-key cryptography [49, pp. 378]. The information provider calculates a cryptographic hash of his document and signs it with his private key. This signature is appended to the document. Once a client has recovered enough shares, she attempts to reconstruct the document and the signature. She decrypts the signature using the public key of the publisher, applies the same cryptographic hash function to the document, and compares the results. If any bits have changed due to accident or malicious intent, the hashes will not match and the document is considered as inaccurate. Depending on a local security policy the client would stop the process here, or retry to gather a different set of shares. Since the signature cannot be forged without the possession of the private key of the publisher, this mechanism ensures origin authentication and end-to-end authenticity of documents, and can also detect maliciously modified shares by verifying the resulting document. Therefore, advanced secret-sharing techniques such as verifiable secret-sharing schemes [39], [40], [57] are not necessary in this application setting.

The client may know the publisher's public key in advance, or retrieve and verify it using a public-key infrastructure (PKI) and store it for a longer time. Alternatively or additionally, the publisher may include a certificate binding his identity to the public key in the document, which is signed by a Certification Authority (CA) run by a trustworthy entity such as the global management coordinator.

B. Privacy Adversary Model and Protection Mechanisms

Enhanced privacy compared to the clear-text ONS is one of the key goals of SHARDIS. It is designed to raise the bar an adversary has to pass to profile users of the system, but without making a predistribution of any cryptographic keys to the clients necessary that could involve major scalability issues [27]. In this paper it is assumed that the address document D itself is not confidential from the publisher's point-of-view, similar to public DNS information (otherwise, stronger cryptographic protection would become necessary [27]).

The main privacy adversary considered is that of a *casually profiling insider*: The adversary may be another participant of the P2P discovery service who in general "plays nicely" and is offering regular information services—but on the other hand, as a secondary business similar to search engine providers, investigates its log files for information that can be used to profile the clients of the DS, i.e., individuals or companies who issue queries for information to the DS. In general, the adversary is interested in tuples (IP,E) where IP is an (potential) client identifier such as his Internet protocol address, and E is a set of EPCs e indicating objects the client is interested in. The protection mechanisms provided are explicitly not designed to protect against a global eavesdropper who can control major portions of the global Internet traffic, or one who is able to control a very large percentage of SHARDIS nodes.

The first mechanism is based on Shamir's secret-sharing scheme [48]. The main idea is to let the information provider split the document D that contains the EPCIS addresses into multiple shares. If an adversary cannot capture enough of these shares, an easy profiling by inference of which client has queried for what kind of object, is prevented. A single share or a too small a set of shares, e.g., captured from network traffic, cannot be used to reconstruct the original document D. The second mechanism is changing the shares and overlay key of an address document D over time (Section IV-D). Third, the query mechanism is designed in such a way that a client reveals her information need only to the node that is currently responsible for the information, and even then only in the obfuscated form of a cryptographic hash.

This section focuses on the possible privacy breach by inferring an EPC from the information contained inside of an address document D. For example, meta data or an URL included in D could directly include or make an inference of the exact EPC possible. Compared to ONS, the use of a DHT for document lookup obfuscates the EPC of interest in the query by applying a CHF to create the overlay ID; but the answer document must be in clear text if no additional key management procedures for encryption can be arranged between hitherto unknown cooperation partners [27]. Here, the secret-sharing mechanism described in Section III is an essential new probabilistic safeguard. Further countermeasures in SHARDIS are briefly discussed at the end of this section.

C. Probability of Document Compromise

If an adversary cannot capture at least $t = k + 1$ shares, the system of linear equations used for reconstructing D will

TABLE II
PROBABILITY OF DOCUMENT COMPROMISE ($n = 10, t = 7$)

N	$m = 20$ (fixed)	$m = 0.2 * N$ (20%)
100	0.00086435840	0.00086435840
10^3	$1.4567850 * 10^{-10}$	0.00086435840
10^4	$1.5279503 * 10^{-17}$	0.00086435840
10^5	$1.5351937 * 10^{-24}$	0.00086435840
10^6	$1.5359194 * 10^{-31}$	0.00086435840

contain fewer equations than unknowns, which means that there are many possible solutions, none of which is more probable than any other. The adversary cannot identify what the client was asking for by reconstructing the answer D.

In practice, the identifier space of a DHT is usually large, for example $c = 2^{160}$ with FreePastry, whereas the number of nodes is very small in comparison. This means that every DHT node is responsible for a very large set of overlay IDs, and the chance that two or more shares of D are distributed to the same node is non-negligible. In the following model it is assumed that the average number of overlay IDs per node is a constant parameter c. The following logical sequence is assumed in the model: a) The adversary chooses his m nodes, and then b) the publisher stores the n shares, each share to exactly one node. The probability $P(c,t)$ that exactly t shares are stored under the adversary's control can be modeled as follows:

$$P(c,t) = \frac{\binom{cm}{t}\binom{cN-cm}{n-t}}{\binom{cN}{n}}. \quad (1)$$

The rationale is to calculate the probability $P(c,t)$ that an adversary gets exactly t shares as the number of possible ways to distribute exactly t shares to the set of overlay identifiers under the adversary's control, which has the size $c \cdot m$. The other $n-t$ shares are distributed to non-adversary overlay IDs (numbering $cN - cm$). The total probability that the adversary gathers *at least* t shares can be computed as the sum of the probabilities with t varying from t to n (assume in practice $m > n$ for a stronger adversary):

$$P(c,'' geqt'') = \sum_{j=t}^{\min(n,m)} P(c,j) = \frac{\sum_{j=t}^{\min(n,m)} \binom{cm}{j}\binom{cN-cm}{n-j}}{\binom{cN}{n}}. \quad (2)$$

For a casual privacy adversary who participates as a company with 20 nodes, the probability to be able to collect enough shares to reconstruct the document for client profiling, is negligible (see Table II, middle column). But also against a more powerful adversary who controls 20% of all the nodes, SHARDIS offers good protection (right column). Table II also demonstrates that increasing the number of nodes under the adversary's control m relative to the number of nodes N does not change the probability of the document compromise, given that t and n remain constant. An approximation (not included for space limitations) shows that this probability mainly depends on the ratio m/N of adversary nodes to all nodes.

Fig. 7 also supports the intuition: the more nodes under the adversary's control (m), the more vulnerable the system (P). The graph also shows that a larger *recovery ratio* t/n greatly reduces the adversary's reconstruction probability P. Increasing

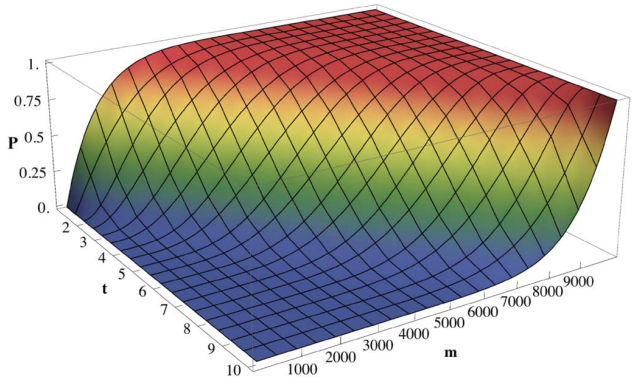

Fig. 7. Reconstruction probability P, depending on varying thresholds t and number m of adversary nodes ($N = 10^5$, $n = 10$).

t, however, leads to a larger communication overhead for the client, and reduces the additional redundancy provided by the sharing scheme.

In the analysis above, a static model is assumed, abstracting from the real-world phenomenon that nodes may join or leave the network at any time, a phenomenon called *churn* in literature. Compared to other P2P networks with free membership, however, the churn rate of SHARDIS, as an infrastructure P2P network by companies bound by contracts, is expected to be relatively low. Furthermore, the model assumes no additional internal replication of the shares is taking place in the DHT, though this could be taken into account by increasing m. In SHARDIS, document availability can be increased by lowering the recovery ratio, in a tradeoff to privacy.

D. Further Privacy and Security Considerations

The privacy protection in SHARDIS rests in part on the assumption that it is hard for an adversary to gain enough shares of a document to reconstruct its content. Even if he knew the document represented by a share and the request of that share by a client revealed the client's information interests, he cannot build a profile over an arbitrary period of time, because the location storing a share for an EPC moves in the overlay over time due to *republication* after each time interval Δ_t, which also involves the generation of new shares by means of a new polynomial. The change of Δ_t will with high probability result in a different storage location. Furthermore, a client can cache responses in order to further conceal her interests.

In addition to the main adversary model of a passive, rather casual profiler, other attacks must be considered in SHARDIS. In a Sybil attack, a single entity assumes many identities within a P2P network in order to gain control over a large fraction of overlay IDs [58]. Sybil attacks can be used to stage further attacks on the DHT, such as sublime eavesdropping attacks, modified overlay routing, to outright denial of service. With secret-sharing, they can also be used to collect more document shares than a fair-playing participant could [45]. Countermeasures against Sybil attacks could include centralized certification and special registration procedures, which were adopted for SHARDIS. Further measures include the use of physical network characteristics or social networks for identifying friends from Sybils, and computational puzzles, see [59] and [60].

Within SHARDIS, a node public-key infrastructure (Node PKI) is assumed that is run by the information publishers and the

central management and CA entity (for example, EPCglobal), see Section III-B. Each node has a unique public-private key pair. Corresponding CA and monitoring procedures must prevent a single enterprise from issuing certificates for node public keys in vast amounts, preventing the possibility to back up Sybil overlay IDs with the necessary cryptographic assurance. The Node PKI would also be used to secure the authenticity of overlay routing information, and to provide confidentiality and authenticity of arbitrary overlay communication, for example, by Transport Layer Security (TLS) [61]; here tradeoffs with increased latency must be investigated further. In addition, the Node PKI can be used to authenticate nodes to clients and publishers; the publisher certificates, on the other hand, will provide a possibility for mutual authentication of publishers to nodes to prevent unauthorized information storage (e.g., spam) in the DHT or document deletion by outsiders, and could also be used for audit trails on DHT document management. However, those features are yet to be integrated into a published open-source DHT, and are currently also not implemented in SHARDIS. FreePastry has started to include some support for TLS.

V. Implementation and Experiments

A. SHARDIS Implementation

The SHARDIS implementation includes two core components, the publisher and client software both written in Java, and a globally distributed DHT installed on the international research platform PlanetLab (PL) [12]. This DHT was developed using a modified version of the open-source storage utility Past, which utilizes FreePastry as overlay network [32]. A new service interface was added to Past, which offers means to generate, insert, and retrieve shares of documents. It also offers a lookup of nodes responsible for a certain overlay key. The service interface was implemented using a mechanism, called `AppSocket`, offered by FreePastry to open a direct line of communication between two nodes. Publishing uses the standard mechanism of Past. It first performs a lookup to determine the storage nodes and then transfers the data from the publisher's gateway node directly to the destination nodes, preventing any other nodes from seeing all shares. The retrieval process was implemented as described in Section III-E. Parameter values used for the implementation are $n = 10$ and $t = 7$, as they seem to offer a good trade-off between privacy (see Section IV) and query overhead.

Publisher and client software ran on contemporary desktop PCs (Pentium Core2 Duo, 2.1 GHz, 4 GB RAM, Sun Java 1.6). The nodes for the DHT are recruited from PL, which offers root access to virtual machines distributed across the globe. PL enables experiments on distributed services with currently more than 500 locations worldwide [12], including scientific institutions and enterprises, all sharing physical resources and bandwidth. In order to generate a set of stable nodes a PL service called CoMon was used. CoMon offers means to inspect important node performance measures. The service was queried every 12 h for a set of stable nodes that were used for the DHT (see Fig. 9). Therefore, only a relatively small, but responsive portion of nodes available through PL was used during the experiments.

TABLE III
Split and Reconstruction Duration (milliseconds, $t = 7$)

	Share Generation (ms)			Document Reconstruction (ms)		
Document Length	Mean	Median	σ	Mean	Median	σ
32	0.26	0	0.57	0.16	0	0.44
64	0.4	0	0.49	0.25	0	0.44
128	0.68	1	1.15	0.6	1	0.5
256	1.14	1	0.49	1.26	1	0.54
512	2.15	2	0.45	2.45	2	0.66
1024	4.26	4	0.7	4.96	5	1.09
2048	7.6	8	1.63	9.03	9	2.51
4096	16.62	16	1.63	19.66	20	3.83

B. Performance Experiments

1) Local Document Split and Reconstruction ($E1$): The first experiment measured the time required for splitting and recombining documents of different sizes. For each document length (l, in byte), 1000 random documents were generated. For each document, seven shares were created and recombined to reconstruct the document. Table III shows the results.

As this experiment shows, the execution time of generating shares and reconstructing documents from shares grows linear for the tested documents lengths. This is consistent with what could have been expected. The implementation provides a new polynomial and set of shares for each byte, resulting in a linear factor based on the document size.

2) Local Document Reconstruction With Multiple Publishers ($E2$): The second experiment focuses on reconstructing multiple documents that were published for the same EPC e. As shown in the previous experiment, reconstructing a single document is relatively fast: about 5 ms for a 1024-byte document. Since SHARDIS offers the possibility to publish multiple documents per EPC, a client has to be able to extract multiple documents from retrieved share sets. Here, the challenge is to identify the shares that represent an original document, which can be identified by its clear text structure and verified by the attached digital signature.

In the first part of this experiment, documents were reconstructed by "brute force," i.e., simply testing all possible combinations of shares. For each of p publishers, a randomly created document was split into $t = 7$ shares. For each document, one share was inserted into buckets for the EPC hashes $h_{(e,i)}$ ($i \in \{1, \ldots, 7\}$), representing the corresponding DHT storage nodes. In each reconstruction try, one share per set was chosen and a potential document candidate reconstructed. If an original document was reconstructed, its shares were removed from the sets, otherwise another combination was tested. The experiment was repeated 100 times for each number of publishers p, and the averages were calculated (Table IV).

The average execution time grows rapidly with the number of publishers p. Even for a relatively small number of four publishers the average execution time has grown by factor 100 in magnitude compared to only two publishers.

It is highly undesirable for the client to spend time on combining shares that do not belong to the same document. The execution time appears to be dominated by the time it takes to reconstruct a document, as the execution time grows linearly with the number of combination attempts. Fig. 8 shows the execution time in relation to the number of share combinations tested. It

TABLE IV
SHARE COMBINATION USING BRUTE FORCE, AVERAGES ($t = 7$)

p	Duration (ms)	Tested Share Combinations
2	175	33.09
3	1593	383.8
4	10430	2478

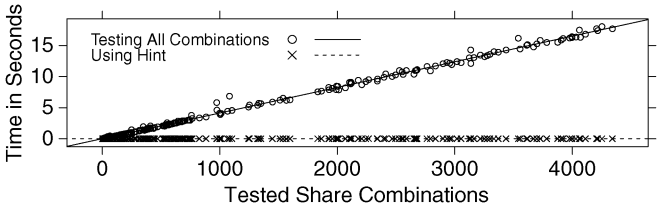

Fig. 8. Execution time (sec), depending on share combinations ($E2$).

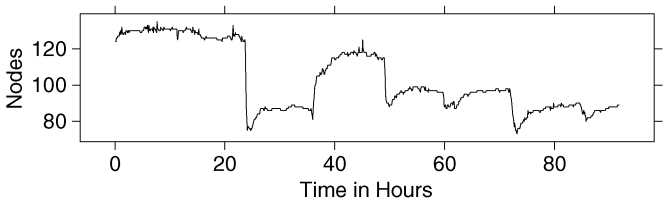

Fig. 9. Number of DHT nodes during experiment $E3$.

TABLE V
SHARE COMBINATION USING HINTS, AVERAGES ($t = 7$)

p	Duration (ms)	Tested Share Combinations
2	14.08	2.01
3	21.31	3.04
4	27.47	4.14
5	32.52	5.16
10	72.03	10.73
20	109.7	23.29
100	1039	254.7

depicts the measured values for all publisher numbers. The time complexity of this brute-force approach is polynomial ($O(p^t)$), since the number of possible combinations increases polynomially with the number of publishers and the execution time grows linearly with the number of combination attempts.

The second part of the experiment used the same setup as the first. This time, however, reconstruction hints were stored in the shares to identify correct share combinations (Section III-E3). The experiment was again repeated 100 times per publisher number. As Table V shows, the hints are sufficient to massively reduce the necessary combination attempts. Time complexity is still polynomial, but evaluating the hint by simple arithmetic is magnitudes faster than computing a polynomial interpolation. Using the hint also exhausts the search space faster because three shares can suffice to exclude combinations.

3) SHARDIS Publish and Retrieval via the Internet ($E3$): The goal of this experiment was to measure the time it takes to publish and retrieve document shares via the Internet by using a SHARDIS implementation on PlanetLab (PL). Three different lookups were measured. (i) FreePastry Lookup (Single Entry): The document was stored without split up, as a single entry.

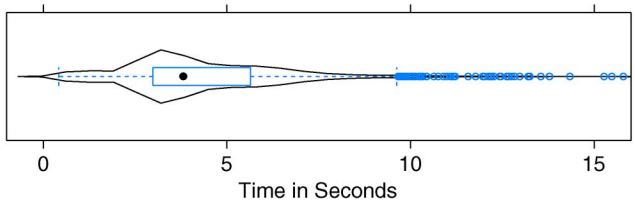

Fig. 10. Duration of document publishing via Internet ($E3$).

The lookup time of this single entry can be seen as baseline. (ii) FreePastry Lookup: FreePastry was used as a service for storing a split-up document. (iii) SHARDIS Lookup: The enhanced service interface was used to perform parts of the lookup at the client. For (ii) and (iii), the following steps were iterated.

1) Randomly generate a 12-byte EPC.
2) Randomly generate a 1024-byte document.
3) Generate ten shares from the document.
4) Generate tenoverlay IDs $h(e \circ i \circ \Delta_t)$ for those shares.
5) Store the ten shares using the overlay IDs in SHARDIS on PL.
6) Lookup the SHARDIS nodes on PL that store those ten shares.
7) Retrieve seven shares from those nodes on PL.
8) Reconstruct the document.

The execution time of the three steps that communicate with the DHT (5, 6, 7) were recorded. The gateway node used to communicate with the DHT was chosen at random for each iteration of the steps.

Nodes on PL differ from future productive SHARDIS nodes in terms of uptime and performance: PL nodes are sometimes unreliable, and can go offline for several reasons at any point in time. There are no contracts on uptime and service quality. Furthermore, an experiment shares its hardware and bandwidth with multiple other concurrent experiments. A set of nodes should also not be expected to stay operational for a long-term experiment [62]. These conditions led to some failures during the communication with and within the DHT. Nonetheless, PL offers interesting insights into distributed real-world deployments. Since the purpose of this experiment was to gain an impression of the response times of a normally operating SHARDIS implementation, only performance values for 10 successfully inserted and seven successfully retrieved shares were evaluated in this experiment, leading to a total count of about 2000 measurements. Fig. 10 shows the time required to successfully insert ten shares into the DHT as a combination of frequency distribution and box plot.

The difference between a FreePastry and a SHARDIS lookup is the location at which the steps 6 (lookup) and 7 (retrieval) take place. For FreePastry, both steps are executed by the DHT node. In SHARDIS, both steps are executed by the client. SHARDIS also uses a modified lookup (Section III-E) that requests twice as much nodes as FreePastry during the lookup procedure. Both lookups result in a list of 10 nodes used for the retrieval step. With a difference in median of 0.364 s between FreePastry (2.134 sec) and SHARDIS lookup (2.498 s), the latter is slightly slower, see Fig. 11 showing the time for the three different lookups. In theory, a single SHARDIS lookup

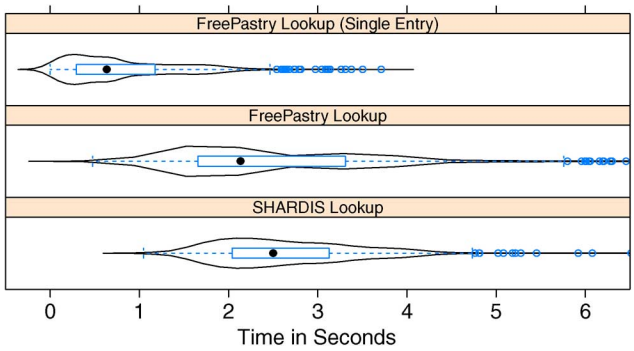

Fig. 11. Duration of document retrieval via Internet $(E3)$.

should correspond to $2t$ single FreePastry lookups (plus t direct queries) if no secret-sharing would have been used; in practice, this overhead is smaller since many operations can be conducted in parallel.

SHARDIS is slower, but still comparable to a lookup in the underlying DHT. It is also offering a low enough latency for many application areas such as automated inventorying. This trade-off between privacy and latency is similar to other privacy-enhancing network technologies. For very latency-critical applications, for example, involving human interaction under high time constraints, its latency should be improved in future work, where research such as [63] will be considered.

VI. Conclusion

This paper presented SHARDIS, a P2P-based discovery service architecture for the EPCglobal Network that enhances client privacy by applying secret-sharing on the information documents of interest. SHARDIS enhances corporate and individual client privacy against profiling without needing key predistribution, making it suitable for flexible, open, and global application scenarios of RFID and the EPC framework.

SHARDIS was implemented using the global experimental testbed PlanetLab, and experimental results showed its practical feasibility, though latency-critical applications may still require performance optimization. Future work will focus on a further comparison of other secret-sharing schemes and DHT substrates to increase the performance of SHARDIS, and enhancing the security of the implemented architecture by additional protection measures.

References

[1] GS1/EPCglobal. [Online]. Available: http://www.gs1.org/epcglobal/
[2] K. Traub *et al.*, "The EPCglobal architecture framework," ver. 1.4, Dec. 2010. [Online]. Available: http://www.gs1.org/gsmp/kc/epcglobal/architecture/
[3] S. F. Wamba and H. Boeck, "Enhancing information flow in a retail supply chain using RFID and the EPC network: A proof-of-concept approach," *J. Theor. Appl. Electron. Commerce Res.*, vol. 3, no. 1, pp. 92–105, Apr. 2008.
[4] F. Thiesse and C. Condea, "RFID data sharing in supply chains: What is the value of the EPC network?," *Int. J. Electron. Business*, vol. 7, no. 1, pp. 21–43, 2009.
[5] S. Evdokimov, B. Fabian, S. Kunz, and N. Schoenemann, "Comparison of discovery service architectures for the internet of things," in *Proc. IEEE Int. Conf. Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'10)*, Newport Beach, CA, 2010, pp. 237–244.
[6] EPCglobal, "Object naming service (ONS) 1.0.1," May 2008. [Online]. Available: http://www.gs1.org/gsmp/kc/epcglobal/ons/
[7] S. Evdokimov, B. Fabian, and O. Günther, "Multipolarity for the object naming service," in *Internet of Things (IOT'08), Zurich, 2008*, ser. LNCS. New York: Springer, 2008, vol. 4952, pp. 1–18.
[8] B. Fabian and O. Günther, "Security challenges of the EPCglobal network," *Commun. ACM*, vol. 52, no. 7, pp. 121–125, 2009.
[9] S. Garfinkel, A. Juels, and R. Pappu, "RFID privacy: An overview of problems and proposed solutions," *IEEE Security and Privacy*, vol. 3, no. 3, pp. 34–43, May–Jun. 2005.
[10] A. Juels, "RFID security and privacy: A research survey," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 2, pp. 381–394, Feb. 2006.
[11] European Data Protection Supervisor, "Opinion of the european data protection supervisor on promoting trust in the information society by fostering data protection and privacy," 2010. [Online]. Available: http://www.edps.europa.eu/
[12] PlanetLab Website. [Online]. Available: http://www.planet-lab.org/
[13] A. Rezafard, "Extensible supply-chain discovery service problem statement," IETF, Internet-Draft, 2008 [Online]. Available: http://tools.ietf.org/html/draft-rezafard-esds-problem-statement-03
[14] BRIDGE, "High level design for discovery services," Aug. 2007. [Online]. Available: http://www.bridge-project.eu/
[15] C. Kürschner, C. Condea, O. Kasten, and F. Thiesse, "Discovery service design in the EPCglobal network—Towards full supply chain visibility," in *Internet of Things (IOT 2008), Zurich, 2008*, ser. LNCS. New York: Springer, 2008, vol. 4952, pp. 19–34.
[16] M. Young, "Extensible supply-chain discovery service concepts," IETF, Internet-Draft, Aug. 29, 2008,, draft-young-esds-concepts-04 [Online]. Available: http://tools.ietf.org/html/draft-young-esds-concepts-04
[17] J. Müller, J. Oberst, S. Wehrmeyer, J. Witt, A. Zeier, and H. Plattner, "An aggregating discovery service for the EPCglobal network," in *Proc. 43rd Int. Conf. Syst. Sci.*, Hawaii, 2010.
[18] H. Balakrishnan, M. F. Kaashoek, D. R. Karger, R. Morris, and I. Stoica, "Looking up data in P2P systems," *Commun. ACM*, vol. 46, no. 2, pp. 43–48, 2003.
[19] R. Cox, A. Muthitacharoen, and R. Morris, "Serving DNS using a peer-to-peer lookup service," in *Proc. 1st Int. Workshop on Peer-to-Peer Systems (IPTPS '01)*, 2002, vol. 2429, pp. 155–165, ser. LNCS, Springer.
[20] V. Ramasubramanian and E. G. Sirer, "The design and implementation of a next generation name service for the internet," in *Proc. ACM SIGCOMM'04*, Portland, 2004, pp. 331–342.
[21] Y. Doi, "DNS meets DHT: Treating massive ID resolution using DNS over DHT," in *Proc. Symp. Appl. Internet (SAINT'05)*, 2005, pp. 9–15.
[22] L. Huang, "Distributed DNS implementation in IPv6," Internet Draft, Apr. 2007. [Online]. Available: http://tools.ietf.org/html/draft-lican-huang-dnsop-distributeddns-03
[23] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. Networking*, vol. 11, no. 1, pp. 17–32, 2003.
[24] K. Park, V. S. Pai, L. Peterson, and Z. Wang, "CoDNS: Improving DNS performance and reliability via cooperative lookups," in *Proc. 6th Symp. Oper. Syst. Design and Implementation (OSDI'04)*, San Francisco, CA, 2004, pp. 199–214.
[25] L. Poole and V. S. Pai, "ConfiDNS: Leveraging scale and history to improve DNS security," in *Proc. 3rd Workshop on Real, Large Distrib. Syst. (WORLDS'06)*, Seattle, WA, 2006.
[26] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and M. Walfish, "A layered naming architecture for the internet," in *Proc. ACM SIGCOMM'04*, New York, 2004, pp. 343–352.
[27] B. Fabian and O. Günther, "Distributed ONS and its impact on privacy," in *Proc. IEEE Int. Conf. Commun. (ICC'07)*, Glasgow, U.K., 2007, pp. 1223–1228.
[28] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz, "Handling churn in a DHT," in *Proc. USENIX Annu. Tech. Conf. (ATEC'04)*, 2004, pp. 127–140.
[29] B. Fabian, "Implementing secure P2P-ONS," in *Proc. IEEE Int. Conf. Commun. (ICC'09)*, Dresden, Germany, 2009.
[30] D. Huang, M. Verma, A. Ramachandran, and Z. Zhou, "A distributed ePedigree architecture," in *Proc. 11th Int. Workshop on Future Trends of Distrib. Comput. Syst. (FTDCS'07)*, 2007.
[31] A. I. T. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Proc. IFIP/ACM Int. Conf. Distrib. Syst. Platforms*, Heidelberg, Germany, 2001, vol. 2218, pp. 329–350, ser. LNCS, Springer.

[32] FreePastry Website [Online]. Available: http://www.freepastry.org/
[33] N. Schönemann, K. Fischbach, and D. Schoder, "P2P architecture for ubiquitous supply chains," in *Proc. 17th Eur. Conf. Inform. Syst. (ECIS'09)*, Verona, Italy, 2009.
[34] M. Dias De Amorim, S. Fdida, N. Mitton, L. Schmidt, and D. S. Ryl, Anglais distributed planetary object name service: Issues and design principles INRIA, Res. Rep. RR-7042, 2009. [Online]. Available: http://hal.inria.fr/inria-00419496/PDF/RR-7042.pdf
[35] S. Shrestha, D. S. Kim, S. Lee, and J. S. Park, "A peer-to-peer RFID resolution framework for supply chain network," in *Proc. 2nd Int. Conf. Future Networks*, 2010, pp. 318–322.
[36] P. Manzanares-Lopez, J. P. Muñoz-Gea, J. Malgosa-Sanahuja, and J. C. Sanchez-Aarnoutse, "An efficient distributed discovery service for EPCglobal network in nested package scenarios," *J. Network Comput. Appl.*, vol. 34, no. 3, pp. 925–937, May 2011.
[37] G. R. Blakley, "Safeguarding cryptographic keys," in *Proc. National Comput. Conf. (NCC), AFIPS*, 1979, pp. 313–317.
[38] M. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *J. ACM*, vol. 36, pp. 335–348, 1989.
[39] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *Proc. 26th Annu. Symp. Foundations Comput. Sci. (SFCS'85)*, 1985, pp. 383–395, IEEE Computer Society.
[40] B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," in *Proc. Advances in Cryptology—CRYPTO'99*, 1999, vol. 1666, pp. 148–164, ser. LNCS, Springer.
[41] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *Proc. Annu. IEEE Symp. Foundations Comput. Sci.*, 1987, pp. 427–438, IEEE Computer Society.
[42] M. Langheinrich and R. Marti, "Practical minimalist cryptography for RFID privacy," *IEEE Syst. J., Special Issue on RFID Technology*, vol. 1, no. 2, pp. 115–128, Dec. 2007.
[43] B. N. Mills and T. Znati, "Increasing DHT data security by scattering data," in *Proc. 17th Int. Conf. Comput. Commun. Networks (ICCCN '08) (Invited Paper)*, St. Thomas, Virgin Islands, 2008, pp. 430–434.
[44] R. Geambasu, T. Kohno, A. A. Levy, and H. M. Levy, "Vanish: Increasing data privacy with self-destructing data," in *Proc. 18th Usenix Security Symp.*, Montreal, Canada, 2009.
[45] S. Wolchok *et al.*, "Defeating Vanish with low-cost Sybil attacks against large DHTs," in *Proc. 17th Annu. Network Distrib. Syst. Security Symp.*, 2010.
[46] EPCglobal, "EPC tag data standards," ver. 1.5, Aug. 2010. [Online]. Available: http://www.gs1.org/gsmp/kc/epcglobal/tds/
[47] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent hashing and random trees," in *Proc. 29th Annu. ACM Symp. Theory Comput. (STOC'97)*, El Paso, TX, 1997, pp. 654–663.
[48] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.
[49] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 4th ed. Englewood Cliffs, NJ: Prentice-Hall, 2006.
[50] B. A. Forouzan, *Cryptography and Network Security*. New York: McGraw Hill, 2007, vol. 721.
[51] D. Eastlake and P. Jones, "US secure hash algorithm 1 (SHA1)," RFC 3174, 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3174.txt
[52] G. C. Dorner, J. M. Ferrard, and H. Lemberg, "Visual Mathematics," in *Illustrated by the TI-92 and the TI-89*. New York: Springer, 2000.
[53] J. Buchmann, *Einführung in die Kryptographie*, 4th ed. New York: Springer, 2008.
[54] SSSS Website, [Online]. Available: http://point-at-infinity.org/ssss/
[55] N. Herrmann, *Höhere Mathematik Für Ingenieure, Physiker und Mathematiker*. Munich, Germany: Oldenbourg Wissenschaftsverlag, 2007.
[56] *Image Processing and Communications Challenges 2*, ser. Advances in Intelligent and Soft Computing, R. S. Choras, Ed. Berlin-Heidelberg: Springer, 2010.
[57] T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority," in *Proc. 21st Annu. ACM Symp. Theory Comput.*, 1989, pp. 73–85.
[58] J. R. Douceur, "The Sybil attack," in *Proc. 1st Int. Workshop on Peer-to-Peer Syst. (IPTPS '01)*, Berlin, Germany, 2002, vol. 2429, pp. 251–260, LNCS, Springer.
[59] G. Danezis, C. Lesniewski-Laas, M. F. Kaashoek, and R. Anderson, "Sybil-resistant DHT routing," in *Proc. ESORICS 2005*, 2005, vol. 3679, pp. 305–318, sers. LNCS, Springer.
[60] G. Urdaneta, G. Pierre, and M. V. Steen, "A survey of DHT security techniques," *ACM Comput. Surveys*, vol. 43, pp. 8:1–8:49, 2011.
[61] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol," RFC 4346, 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4346.txt
[62] N. Spring, L. Peterson, A. Bavier, and V. Pai, "Using PlanetLab for network research: Myths, realities, and best practices," *SIGOPS Oper. Syst. Rev.*, vol. 40, no. 1, pp. 17–24, 2006.
[63] F. Dabek, J. Li, E. Sit, J. Robertson, M. Kaashoek, and R. Morris, "Designing a DHT for low latency and high throughput," in *Proc. 1st USENIX Symp. Networked Syst. Design Implementation (NSDI'04)*, 2004, pp. 85–98.

Benjamin Fabian received the Diploma degree in mathematics from the Free University of Berlin, Berlin, Germany, and the Ph.D. degree in information systems from Humboldt-Universität zu Berlin.

He is a Senior Researcher and Project Manager at the Institute of Information Systems, Humboldt-Universität zu Berlin.

Tatiana Ermakova received the M.Sc. degree in information systems from Humboldt-Universität zu Berlin, Berlin, Germany.

She is a Research Assistant at the Ladislaus von Bortkiewicz Chair of Statistics, Humboldt-Universität zu Berlin.

Cristian Müller is working towards the Degree in computer science at Humboldt-Universität zu Berlin, Berlin, Germany.

He is a student assistant at the Institute of Information Systems, Humboldt-Universität Berlin.