

# Some Like It Private

## Sharing Confidential Information Based on Oblivious Authorization

Privacy-Preserving Policy-Based Information Transfer (PPIT) lets entities that lack mutual trust share sensitive information. The authors discuss the security of two efficient PPIT constructs, then propose an innovative construct that allows entities to efficiently verify the equality of their information.



EMILIANO  
DE CRISTOFARO  
*University of  
California,  
Irvine*

JIHYE KIM  
*Seoul National  
University*

Investigation authorities worldwide currently elicit substantial amounts of personal data, but critics are voicing concern over the implied privacy threats. Recent controversial situations have come to the public's attention, such as the US FBI's demand for guest information from Las Vegas hotels between 2003 and 2004 (see <http://tinyurl.com/FBI-privacy-vegas>), or the US Department of Homeland Security's practice of retaining sensitive information (including credit-card numbers) for US air passengers (<http://tinyurl.com/What-DHS-Knows-About-You>).

As we seek to conjugate the efficacy of law enforcement and privacy guarantees, we face an apparent dilemma. Data *owners* store potentially sensitive data records, such as employee files, that comprise an identifier (*id*) string (social security number, name, or serial number) and the associated bulk data (*data*). Data *requesters* want to retrieve information the owner is storing and based on appropriate authorization, often without revealing their requests' specific content.

One possible solution to protect privacy for both these entities is to employ a trusted server with a copy of the owner's data. Whenever an authorized requester wishes to retrieve some information from the owner, it contacts the server, authenticates, and receives the corresponding records (if any) over a secure channel. Although we imagine scenarios in which this method is adequate, it has some significant drawbacks. First, the owner loses privacy (that is, database records' secrecy) if an adversary compromises the server. Next, operations would be discontinued if the server were unavailable. Finally, possession of private information would require the server to demonstrate proper man-

agement and disposal of such information when no longer needed.

To address these issues, we turn to Privacy-Preserving Policy-Based Information Transfer (PPIT) protocols, which enable access to private information via digital certificates, issued by an offline certificate authority (CA). The CA isn't trusted with the data itself but is intrinsically needed to verify whether the requester is entitled to access information from the data owner. The idea behind PPIT is that the data owner and requester conditionally agree on a shared key and establish a session encryption key (à la Diffie-Hellman,<sup>1</sup> which we discuss more later) to efficiently transfer encrypted records in such a way that requesters can decrypt only those records they're authorized to retrieve. Furthermore, the requester doesn't disclose the corresponding authorizations or the records retrieved. The necessary condition upon key establishment is the *oblivious* verification of a digital signature, meaning that the data owner doesn't learn the outcome of such verification.

In this article, we revisit two PPIT constructions based on RSA digital signatures<sup>2</sup> and identity-based encryption (IBE).<sup>3</sup> Then, we propose an innovative and low-cost PPIT-derived construct that lets entities efficiently verify whether they hold the exact same private information. Finally, we evaluate these protocols to underline their applicability to real-world problems.

### Preserving Privacy with PPIT

Consider the following: the University of Springfield is confronted with an FBI investigation concerning one

of its employees (Alice). The FBI needs to access Alice's records but is reluctant to divulge her identity to the university. The FBI might be concerned about unwarranted rumors concerning Alice's reputation, for example, or that she would be alerted to its investigation. On the other hand, the university, although it must cooperate, won't grant access to all employee records due to privacy concerns. We thus encounter the two opposing desires we described in the introduction: the data owner (the university) should only disclose data to authorized requesters (an FBI agent with a valid warrant), and requesters don't want to reveal their requests' content or show evidence of alleged authorization.

PPIT protects an owner's data against unauthorized access, which benefits both owners and requesters (the latter retrieve only the information they're entitled to, restricting their liability). For instance, merely possessing any university employee's sensitive information would require the FBI to demonstrate that it treated this data appropriately and disposed of it when no longer needed. Considering several recent incidents involving massive losses (and thefts) of sensitive government and commercial employees' records (see, for instance, <http://tinyurl.com/famous-dataloss>), the FBI might be unwilling to assume additional risk. Additionally, PPIT protects the privacy of requesters' authorizations. In several cases, such as the university example, the authorization information is itself sensitive and must be safeguarded.

In addition to the basic entities (data owner, data requester, and CA), PPIT involves several other elements.

### Authorizations

In PPIT, an authorization (for a record identified by  $id^*$ , for example) is a digital signature. In this article, we denote an authorization with  $\sigma$ .

### Privacy Goals

We assume that the owner is honest but curious—that is, it faithfully inputs all its records, but during or after the interaction, it tries to learn the requester's inputs. On the other hand, requesters might be arbitrarily malicious, potentially deviating from the protocol by trying to learn more information than what they're entitled to. However, we don't allow different requesters to communicate during protocol execution. We summarize PPIT's (informal) privacy goals as follows:

- *Information privacy.* Requesters should obtain information related to any record only if they've received corresponding authorization from the CA.
- *Authorization privacy.* The owner shouldn't learn which records requesters hold authorization for or which ones they're requesting.
- *Requester unlinkability* (optional). Across multiple interactions, the owner shouldn't be able to link re-

questers' authorizations. (This prevents the owner from learning, for instance, whether a requester's authorizations have changed or whether multiple requesters hold the same authorizations, and minimizes the possibility of privacy leaks. Otherwise, if one requester's inputs are leaked, then the owner could learn all linked requesters' inputs.)

- *Forward security* (optional). PPIT should prevent malicious requesters from violating information privacy with regard to past recorded interactions, using, for instance, authorizations obtained at a later time. (Suppose the FBI interacts with the university without possessing an authorization for Alice's record: PPIT prevents it from accessing that record, but the FBI later obtains authorization. Unless the protocol enforces forward security, the FBI can use that to recover Alice's file from the recorded interaction, which would be a privacy violation if the court's authorization is time-bounded.)

Details about how we define privacy are available in prior research.<sup>4</sup>

### Developing PPIT Constructions

As we mentioned, one important building block of our PPIT constructions is based on the key-exchange protocol Whitfield Diffie and Martin Hellman proposed.<sup>1</sup> Then, we constructed two basic PPIT protocols, with possible extensions for more complex scenarios.

### Diffie-Hellman Key Exchange

The Diffie-Hellman<sup>1</sup> key exchange is a protocol by which two parties—such as Alice and Bob—establish a shared secret key over an insecure channel without any prior secret and based only on some common public parameters  $p$  and  $g$  (where  $\langle g \rangle$  generates a cyclic subgroup of  $Z_p^*$ ).

First, Alice generates a random private value  $x$ , and Bob generates a random private value  $y$ . Then, they derive and exchange their public values using the parameters  $p$  and  $g$  and their private values. Alice's public value is  $g^x \bmod p$  and Bob's public value is  $g^y \bmod p$ . Finally, Alice computes  $(g^y)^x \bmod p$ , and Bob computes  $(g^x)^y \bmod p$ . Alice and Bob now have a shared secret key  $g^{xy} \bmod p$ , which can be used for the subsequent secure communication.

When we adapt the Diffie-Hellman key exchange for the PPIT scenario (with the owner and requesters in place of Alice and Bob), we must consider two additional properties. First, a requester must obtain the shared key only if it holds an appropriate certificate. Second, the owner shouldn't learn whether a requester holds such a certificate. For these additional features, PPIT merges digital signatures related to the certificates into the key-exchange procedure. The owner derives a shared key by obviously verifying the sig-

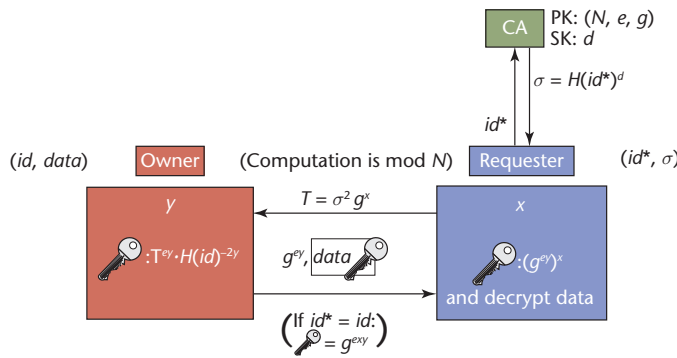


Figure 1. The basic Privacy-Preserving Policy-Based Information Transfer (PPIT) protocol, based on RSA signatures. The owner’s input is  $(id, data)$  and the requester’s input is  $(id^*, \sigma)$ . If  $id = id^*$ , and  $\sigma$  is a valid RSA signature on  $id^*$ , then the requester will learn the owner’s information without exposing its signature.

nature, whereas a requester extracts the same shared key only if it holds a valid signature (authorization) for the corresponding record.

### Basic PPIT

To describe our protocols, let’s first examine a simple scenario in which an owner stores only one record  $(id, data)$ , and the requester obtains a single authorization  $(\sigma)$  from the CA to retrieve data identified by  $id^*$ .

We build this first scheme by merging RSA signatures into the Diffie-Hellman key-exchange protocol.<sup>1</sup> Figure 1 illustrates this basic protocol. Given an RSA public key  $(N, e, g)$  with a quadratic residue group generator  $g$  and a secret key  $d$ , a requester gets from the CA a digital signature  $\sigma = H(id^*)^d$  on its  $id^*$  (for some hash function  $H$ ). Note that the signature itself is private information because anyone (including the owner) can verify it using the CA’s public key. So, instead of sending the raw signature, the requester sends a blinded version,  $T = \sigma^2 g^x \pmod N$  (for some random secret  $x$  it picked), which doesn’t reveal any information about  $\sigma$ . Although the owner can’t verify the signature’s correctness, if  $id = id^*$ , then it can extract the requester’s public key  $g^{ex}$  through signature verification. Given that

$$H(id)^{ed} = H(id),$$

$$T^e H(id)^{-2} = H(id^*)^2 g^{ex} H(id^*)^{-2} = g^{ex}.$$

Note that the base in this key exchange is  $g^e$ .

At the same time, the owner doesn’t know whether it has extracted the requester’s public key correctly: this key has a uniform distribution, regardless of signature validity.

Finally, the owner picks its random secret  $y$  and computes the session key—that is,  $(g^{ex})^y$  if the obli-

ous verification succeeds, a random value otherwise—and sends its public key  $g^{ey}$  and the associated data with the session key. Note that only if  $id = id^*$ , the session key is  $(g^{ex})^y$ , and the requester can recompute it.

The scheme protects the requester’s privacy because the owner doesn’t know whether the requester succeeds in message decryption. This construction is proven secure under the RSA assumption—that is, it is computationally infeasible to factor large numbers that are a product of (typically) two prime factors.<sup>4</sup>

We developed our second basic PPIT protocol based on anonymous IBE. An IBE system is a public-key system in which any string is a valid public key. In such a system, a trusted third party (CA), having a secret master key, lets entities generate the private key corresponding to any public-key string by signing the latter. In particular, we select Dan Boneh and Matthew Franklin’s IBE (BF-IBE)<sup>3</sup> for our PPIT presentation.

BF-IBE is a pairing-based cryptography tool using bilinear maps over groups of large prime order. For example, if  $G_1$  and  $G_2$  are two cyclic groups of some large prime order  $q$ , then  $e: G_1 \times G_2 \rightarrow G_2$  is called a bilinear map if, for all  $a, b \in \mathbb{Z}_q$ ,  $P, Q, \in G_1$ , we have  $e(P^a, Q^b) = e(P, Q)^{ab}$ .

Adapting IBE into the basic PPIT is simple. The CA’s key pair is  $PK: (q, g, \hat{g})$  and  $SK: s$ . To encrypt a record, the owner derives a key  $K$  from the record identified  $id$ , the random  $y$  picked by the owner, and the CA’s public key  $g^s$ , such that  $K = e(g^s, H(id)^y)$  for some map-to-point hash function  $H$ . Given  $g^y$  and the ciphertext, the requester can decrypt it if  $id = id^*$  and it has a signature  $\sigma = H(id^*)^s$  from the CA, by computing

$$e(g^y, \sigma) = e(g^y, H(id^*)^s) = e(g, H(id)^s)^y.$$

Note that using anonymous IBE is mandatory because PPIT requires that the owner protects the identifying information (used as a public key). The security of PPIT based on IBE is implied by anonymous IBE’s security. (An anonymous IBE scheme prevents a decryptor from learning the public key used for encryption; further details are available elsewhere.<sup>4</sup>)

### General PPIT

We now address a case in which the owner holds a database with multiple records, rather than just a single record. To this end, we explain how to efficiently extend the basic PPIT protocols we just described. We also allow requesters holding multiple authorizations to batch more requests in a single interaction.

**Multiple records.** Consider the following setup: a requester holds a digital signature  $\sigma$  on some data signed by the CA, whereas the owner has many data records (say,  $n$ ), corresponding to a set  $(id_1, data_1), \dots, (id_n, data_n)$ .

A naïve solution is to repeat the basic protocol  $n$

times. The owner encrypts (and transmits) each of its  $n$  records with  $n$  different keys. It's impossible to obtain strong privacy requirements without having the owner processing each and any of its data records. However, the naïve solution wouldn't be optimal with regard to the requester's overhead—a requester must try to decrypt  $n$  records, even though it inputs only one request.

We can improve PPIT's complexity with the following optimization technique based on key tagging. We let the owner use the same random value,  $\gamma$ , across all its records. Then, we require the owner to append every encryption with a key tag. These tags are the output of a one-way function (such as a cryptographic hash function) computed over the key itself. On the other hand, a requester computes its own tag over the alleged (Diffie-Hellman) session key. Hence, it searches and decrypts only the record with a matching tag—that is, the one for which it has the valid corresponding decryption key. This reduces the complexity on the requester side to a constant.

**Multiple requests.** We now consider the most complex scenario: a requester holding multiple authorizations (say,  $m$ ) batches more requests in a single interaction. Hence, the requester's input becomes  $((id_1^*, \sigma_1), \dots, (id_m^*, \sigma_m))$ , while the owner's input is  $(id_1, data_1), \dots, (id_n, data_n)$ .

This extension has different implications on RSA- and IBE-based PPIT. Recall that IBE-based PPIT is non-interactive, and the owner's encryption keys are independent of the requester's alleged authorizations. Hence, the communication overhead and the complexity on the owner's side aren't affected. In RSA-based PPIT, on the other hand, the owner's encryption keys do depend on the requester's alleged authorizations. Consequently, the communication overhead and owner complexity are quadratic with regard to  $n$  and  $m$ .

In the RSA-based PPIT scheme, however, we can show<sup>4</sup> that the owner can precompute some of the owner's exponentiations ahead of time and reduce the owner's complexity to a linear number of exponentiations and a quadratic number of multiplications. (Multiplications are much more efficient than exponentiations.)

On the other hand, we can also apply the optimization we discussed earlier in this setting. Thus, the complexity at the requester side would still be linear with regard to  $m$ —that is, the number of the requester's authorizations.

Figure 2 illustrates the extended RSA-PPIT protocol for multiple records and multiple requests, and Figure 3 illustrates the extended IBE-PPIT protocol. For RSA-PPIT, we omit the RSA parameters setup and the description of how a requester obtains authorizations—that is, signatures—from the CA, given that they mirror those of the basic protocol (see Figure 1).

Note that the extended RSA- and IBE-based proto-

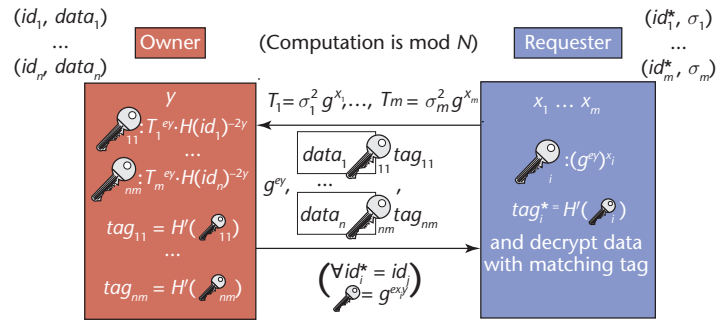


Figure 2. The extended Privacy-Preserving Policy-Based Information Transfer (PPIT) protocol based on RSA signatures. Given the owner's input  $(id_1, data_1), \dots, (id_n, data_n)$  and the requester's input  $(id_1^*, \sigma_1), \dots, (id_m^*, \sigma_m)$ , the requester will learn all  $data_i$ 's such that for some  $i$  and  $j$ ,  $id_i = id_j^*$  and  $\sigma_j$  is a valid RSA signature on  $id_j^*$ .

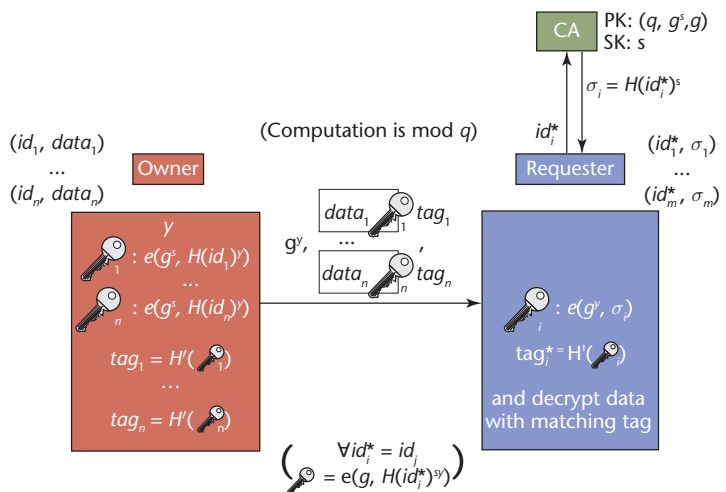


Figure 3. The extended Privacy-Preserving Policy-Based Information Transfer (PPIT) protocol based on identity-based encryption. The owner encrypts each  $data_i$  with corresponding  $id_i$  and the certificate authority's public key, and the requester can decrypt  $data_i$  only if it obtains a certificate on  $id_j$ .

cols achieve the privacy requirements we laid out earlier in the article and are proven secure under, respectively, the RSA assumption and the bilinear Diffie-Hellman assumption. We omitted proofs due to space considerations, but more details are available elsewhere.<sup>4</sup>

### Low-Cost PPIT for Set Equality

In the following, we propose an (unpublished) innovative and low-cost PPIT-derived protocol that focuses on a specific scenario. Consider two entities willing to share sensitive information in an all-or-nothing way—that is, they want to check the equality of their private information sets. We call this *problem set equality*, and (similar to PPIT) we consider a case in which only one entity (upon authorized inputs) receives the

## Sharing Sensitive Data

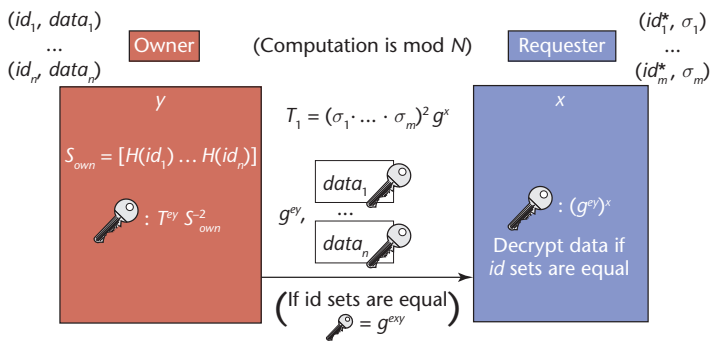


Figure 4. Efficient RSA-based Privacy-Preserving Policy-Based Information Transfer (PPIT) for set equality. The requester accumulates, blinds, and sends its private authorized inputs. Only if the owner's inputs match does the requester learn the equality and also obtain the corresponding data records.

protocol's output. (Assuming that no entity aborts the protocol prematurely, both entities receive output by running the protocol twice.)

We developed an RSA-based scheme that achieves constant complexity using RSA's multiplicative, homomorphic properties. The idea is to use an aggregated group as a basic unit in computations. For instance, considering the owner's input  $(id_1, data_1), \dots, (id_n, data_n)$  and requester's input  $(id_1^*, \sigma_1), \dots, (id_m^*, \sigma_m)$ , we denote three different multiplication groups:  $S_\sigma = \sigma_1 \sigma_2 \dots \sigma_m$ ,  $S_{req} = H(id_1^*) H(id_2^*) \dots H(id_m^*)$ , and  $S_{own} = H(id_1) H(id_2) \dots H(id_n)$ . In the first round, the requester sends  $T = (S_\sigma)^2 g^x$  (for some random secret  $x$  it picked). Given that  $(S_\sigma)^e = S_{req}$ , the owner can correctly extract the requester's public key by computing  $T^e (S_{own})^{-2} = g^{ex}$ , only if the identifier sets are the same—that is,  $S_{own} = S_{req}$ . Figure 4 illustrates the resulting scheme. Note that the requester correctly recovers the key (and decrypts data records) only if the sets are equal. The protocol is secure under the RSA assumption. (We omitted the proof due to space limitations). Again, we omit the RSA parameters setup and the description of how a requester obtains authorizations from the CA because they mirror those in Figure 1.

The protocol Figure 4 depicts is very efficient, incurring only two to three exponentiations for each entity, regardless of the entity's input size. The protocol accurately executes without any troublesome ordering assumption.

### Discussion and Evaluation

Besides using two different cryptographic primitives, the two proposed protocol instantiations we presented have some differences worth analyzing.

IBE-PPIT comes with the desirable property of noninteractivity—that is, requesters don't need to send any form of their private inputs to the owner. This feature helps maintain the computational and

communication complexity of the protocol linear with regard to the number of requester authorizations.

The cryptographic primitive behind RSA-PPIT (the RSA standard digital signature scheme) involves much less expensive operations. This makes it the better choice for interactions with a single authorized request. Furthermore, although both IBE-PPIT and RSA-PPIT meet the privacy requirements we presented earlier, as well as authorization unlinkability, only RSA-PPIT achieves forward security. In fact, a requester recording transcripts of IBE-PPIT can use authorizations obtained later to decrypt information contained in those transcripts. This might violate privacy requirements in several practical scenarios. RSA-PPIT is immune to this problem because it includes ephemeral values bound to only one interaction. We could modify IBE-PPIT so that the owner and the requester establish an ephemeral Diffie-Hellman key, but only at the cost of increasing computational complexity and making the protocol interactive.

As we turn to cryptography for valuable solutions to several real-world problems, efficiency becomes an imperative requirement. Unpractical protocols implementing private and authorized information transfer might yield valuable theoretical results but fail to help in reality. To this end, we implemented the different PPIT instantiations and analyzed their performance. Our goals were to identify possible obstacles and optimizations when confronting the real deployment of such protocols, determine whether our PPIT instantiations are fast enough for their intended use, and compare our protocols to the naïve solutions.

We tested prototype implementations on a single machine, a Dell PC with two quad-core Intel Xeon 1.60-GHz CPUs with 8 Gbytes RAM (we performed all computation on a single core). We developed the source code in ANSI C using the OpenSSL and PBC libraries. We used 1,024-bit moduli for RSA-PPIT, and 512-bit group elements and 160-bit primes for IBE-PPIT. All tests measured the total computation time for the PPIT interaction, but we didn't measure runtimes for operations that we can precompute or the potential transmission delay required to transfer data on a communication channel.

We considered two sets of experiments:

- The owner stores many records, varying from 1 to 1,000, and receives a single request.
- The owner stores 1,000 records and receives many requests, varying from 1 to 1,000.

We tested RSA- and IBE-PPIT (see Figures 2 and 3), as well as naïve solutions derived from simply reiterating the basic protocols discussed earlier. Note the naïve solutions incur a quadratic number of exponentiations or pairing operations (either at the owner or

**Table 1. Total computation time for Privacy-Preserving Policy-Based Information Transfer (PPIT) constructions.**

Owner	Requester	RSA-PPIT (seconds)	Naïve RSA (seconds)	IBE-PPIT (seconds)	Naïve IBE (seconds)
1	1	0.02	0.02	0.02	0.02
250	1	0.82	1.50	3.18	3.86
500	1	1.62	2.98	6.34	7.70
750	1	2.43	4.50	9.50	11.55
1,000	1	3.54	5.99	12.67	15.36
1,000	250	21.49	626.88	12.13	2,706.75
1,000	500	42.01	1,253.44	12.92	5,413.55
1,000	750	63.44	1,880.16	13.68	8,120.72
1,000	1,000	90.01	2,506.88	14.49	10,827.31

requester side) and thus aren't scalable in real-world scenarios. The complexity of related work satisfying the same privacy requirements as PPIT is comparable to naïve solutions (see the "Related Work in Privacy-Preserving Information Transfer" sidebar). In particular, implementing PPIT with multiple records or requests using adaptations of Oblivious Signature-Based Envelopes,<sup>5</sup> Public-Key Encryption with Keyword Search,<sup>6</sup> or Authorized Private Search<sup>7</sup> would incur quadratic complexities.

Table 1 reflects the measured computation time for the tested protocols. RSA-PPIT is roughly four times faster than IBE-PPIT for single requests, whereas IBE-PPIT is preferable for multiple requests. Both RSA- and IBE-PPIT achieve a remarkable speed-up compared to the naïve solutions, and speed-up grows exponentially for multiple request settings. (Additionally, recent work from one of us [Emiliano De Cristofaro] further improves the RSA-PPIT scheme in the context of *authorized private set intersection*,<sup>8</sup> which computes the intersection of a requester's and an owner's private sets, given that the requester's inputs are authorized.)

As a final observation, we recall that the communication complexity is linear in the size of the owner's database for IBE-PPIT and quadratic in the size of the owner's database and the requester's authorizations for RSA-PPIT. We can compare this to the square-logarithmic communication complexity Symmetric Private Information Retrieval (SPIR) protocols achieve.<sup>9</sup> However, SPIR protocols have an increased computational overhead and don't guarantee that the requester's searches are authorized. In fact, in our experiments, we found that for several practical settings, transferring the entire encrypted database is more efficient than running PIR.<sup>10</sup>

**W**e plan to follow up with solutions for developing a noninteractive PPIT protocol supporting forward security and extending privacy guarantees against arbitrarily malicious adversaries. □

### Acknowledgments

This research was supported by the US Intelligence Advanced Research Projects Activity (IARPA) under grant number FA8750-09-2-0071 as well as by the Basic Science Research Program through the National Research Foundation of Korea, funded by the Ministry of Education, Science and Technology (2009-0070095) and by the "Developing Future Internet Network Model—Mathematical Approach" of the National Institute for Mathematical Sciences.

### References

1. W. Diffie and M. Hellman, "New Directions in Cryptography," *IEEE Trans. Information Theory*, vol. 22, no. 6, 1976, pp. 644–654.
2. R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Comm. ACM*, vol. 21, no. 2, 1978, pp. 120–126.
3. D. Boneh and M.K. Franklin, "Identity-Based Encryption from the Weil Pairing," *SIAM J. Computing*, vol. 32, no. 3, 2003, pp. 213–229.
4. E. De Cristofaro et al., "Privacy-Preserving Policy-Based Information Transfer," *Proc. Privacy Enhancing Technology Conf. (PETS 09)*, LNCS, Springer, 2009, pp. 164–184.
5. N. Li, W. Du, and D. Boneh, "Oblivious Signature-Based Envelope," *Distributed Computing*, vol. 17, no. 4, 2005, pp. 293–302.
6. D. Boneh et al., "Public-Key Encryption with Keyword Search," *Proc. Int'l Conf. Theory and Applications of Cryptographic Techniques (Eurocrypt 04)*, LNCS 3027, Springer, 2004, pp. 506–522.
7. J. Camenisch et al., "Blind and Anonymous Identity-Based Encryption and Authorized Private Searches on Public Key Encrypted Data," *Proc. Int'l Conf. Practice and Theory in Public Key Cryptography (PKC 09)*, LNCS 5443, Springer, 2009, pp. 196–214.
8. E. De Cristofaro and G. Tsudik, "Practical Private Set Intersection Protocols with Linear Complexity," to appear in *Proc. Int'l Conf. Financial Cryptography and Data Security*, 2010.

### Related Work in Privacy-Preserving Information Transfer

The problem of privacy-preserving information transfer bears some resemblance to those addressed by other cryptographic constructs.

Oblivious Signature-Based Envelopes (OSBEs)<sup>1</sup> solve a similar problem as Privacy-Preserving Policy-Based Information Transfer (PPIT) in a single-record, single-authorization setting. OSBEs allow a sender to release some information to a receiver, conditional on the latter possessing a signature from a trusted authority on a message known to both parties, while the sender learns nothing about the signatures the receiver holds. As with PPIT, parties can instantiate OSBEs with both RSA signatures and identity-based encryption (IBE). However, OSBEs don't consider the more general case with multiple records or authorizations. A possible extension to OSBEs would be to run multiple executions in parallel, but this involves quadratic complexities comparable only to naïve PPIT solutions.

Private Information Retrieval (PIR)<sup>2</sup> is also related to PPIT. PIR lets a client privately retrieve information from a public database (it requires a communication complexity strictly smaller than transferring the entire database). In other words, the client's query target is kept hidden from the database server. However, because data is public, the server isn't concerned with data privacy and doesn't require authorized queries, which are opposite to PPIT's requirements.

Oblivious Transfer (OT)<sup>3</sup> allows a receiver to obliviously obtain one message from a set offered by a sender. The sender doesn't learn which message the receiver retrieves, and the receiver doesn't access any other message. Particularly, the generalization of OT<sup>4</sup> assisting keyword search resembles PPIT. Still, OT is different from PPIT because the receiver's search isn't authorized.

The so-called symmetric PIR (SPIR)<sup>5</sup> combines OT and PIR by enforcing both OT's server privacy requirements and PIR's need to minimize communication complexity. Yet, the requester's queries need not be authorized, contrary to PPIT.

Public-Key Encryption with Keyword Search (PEKS)<sup>6</sup> is a primitive that allows a designated entity to search encrypted data via keywords. In a typical scenario, Bob sends encrypted email to Alice using Alice's public key. Both the email's contents and the keywords are encrypted, sent, and stored in Alice's email server. PEKS aims to enable Alice to let the email server test whether, for

instance, "urgent" is a keyword from the message, but the server should learn nothing else about the email. Although motivating scenarios are different, PEKS could be adapted to PPIT (for example, if Alice is set to be the certificate authority [CA], the email server the requester, and Bob the owner). However, the requester overhead would increase to  $O(nm)$  bilinear map operations.

Interestingly—and independent from our PPIT work—PEKS was recently extended to Authorized Private Search (APS) on public-key encrypted data,<sup>7</sup> which permits authorized private searches by keywords on encrypted data in a public-key setting and satisfies all PPIT requirements. (APS also protects the privacy of the requester's interest against the CA—that is, the CA doesn't learn which request is being authorized, which might not be necessary for a PPIT setting.) Yet, APS isn't as efficient as PPIT. Decryption in APS has a quadratic complexity, whereas in PPIT it has a linear complexity. Moreover, APS encryption involves six bilinear map operations (which are highly expensive underlying cryptographic operations), whereas IBE-PPIT requires only one.

#### References

1. N. Li, W. Du, and D. Boneh, "Oblivious Signature-Based Envelope," *Distributed Computing*, vol. 17, no. 4, 2005, pp. 293–302.
2. B. Chor et al., "Private Information Retrieval," *Proc. Symp. Foundations of Computer Science (FOCS 95)*, IEEE CS Press, 1995, pp. 41–50.
3. M. Rabin, *How to Exchange Secrets by Oblivious Transfer*, tech. report TR-81, Harvard Aiken Computation Laboratory, 1981.
4. W. Ogata and K. Kurosawa, "Oblivious Keyword Search," *J. Complexity*, vol. 20, nos. 2–3, 2004, pp. 356–371.
5. Y. Gertner et al., "Protecting Data Privacy in Private Information Retrieval Schemes," *Proc. ACM Symposium on Theory of Computing (STOC 98)*, ACM Press, 1998, pp. 151–160.
6. D. Boneh et al., "Public-Key Encryption with Keyword Search," *Proc. Int'l Conf. Theory and Applications of Cryptographic Techniques (Eurocrypt 04)*, LNCS 3027, Springer, 2004, pp. 506–522.
7. J. Camenisch et al., "Blind and Anonymous Identity-Based Encryption and Authorized Private Searches on Public Key Encrypted Data," *Proc. Int'l Conf. Practice and Theory in Public Key Cryptography (PKC 09)*, LNCS 5443, Springer, 2009, pp. 196–214.

9. C. Gentry and Z. Ramzan, "Single-Database Private Information Retrieval with Constant Communication Rate," *Proc. Int'l Colloquium Automata, Languages, and Programming (ICALP 05)*, LNCS 3580, Springer, 2005, pp. 803–815.

10. R. Sion and B. Carbutar, "On the Computational Practicality of Private Information Retrieval," *Proc. International Colloquium on Automata, Languages, and Programming (NDSS 07)*, Internet Soc., 2007.

**Emiliano De Cristofaro** is a PhD candidate at the University of California, Irvine. His research focuses on efficient yet provably secure privacy-preserving protocols, as well as network

security and applied cryptography. De Cristofaro has a Laurea in computer science from the University of Salerno, Italy. He's a student member of IEEE and the International Association for Cryptology Research. Contact him at [edecrist@uci.edu](mailto:edecrist@uci.edu).

**Jihye Kim** is a postdoctoral researcher in the Information Security and Cryptology Research Center at Seoul National University. She's working on privacy-preserving protocols and threshold schemes, and her research interests are network security, applied cryptography, and fault-tolerant and distributed computing. Kim has a PhD in computer science from the University of California, Irvine. She's the corresponding author for this article. Contact her at [jihyek@snu.ac.kr](mailto:jihyek@snu.ac.kr).