INVITED PAPER *Special Section on Information and Communication System Security*

Analysis of Existing Privacy-Preserving Protocols in Domain Name System

Fangming ZHAO[†a)], **Yoshiaki HORI**[††], ***and*** **Kouichi SAKURAI**[††], ***Members***

SUMMARY In a society preoccupied with gradual erosion of electronic privacy, loss of privacy in the current Domain Name System is an important issue worth considering. In this paper, we first review the DNS and some security & privacy threats to make average users begin to concern about the significance of privacy preservation in DNS protocols. Then, by an careful survey of four noise query generation based existing privacy protection approaches, we analyze some benefits and limitations of these proposals in terms of both related performance evaluation results and theoretic proofs. Finally, we point out some problems that still exist for research community's continuing efforts in the future.

key words: *domain name system (DNS), privacy, private information retrieval, random noise*

1. Introduction

With the development of automatic information processing, it is necessary to consider privacy protection in relation to personal information. An overview of the evolution of data privacy protection is presented in [1]. Two of the main international institutions in this context are the Council of Europe's 1981 Convention for Protection of Individuals with regard to Automatic Processing of Personal Data [2] and the Organization for Economic Cooperation and Development (OECD) Guidelines on the Protection of Privacy and Transborder Flows of Personal Data [3], in which 30 developed nations work together. With the rise of the importance of computers in the western economies and global trade, the documents have played a leading role in the development of privacy laws in the EU, Canada, and other jurisdictions. Their main principles are: collection limitation, purpose specification, use limitation, data quality, security safeguards, openness, individual participation, and accountability. These rules describe personal data as any information relating to an identified or identifiable individual. The expression of data protection in various declarations and laws varies, but all require that personal data must be kept secure. Thus, information systems must take responsibility for the data they manage. Therefore, the main challenge in privacy protection is to share some data while protecting personal data.

The critical internet infrastructure component *Domain Name System* (*DNS*) was designed in the late eighties. Since the immense growth of the Internet was not foreseen, the scalable design did not take the abuse patterns that come with that into account, and DNS stakeholders need to be aware of the current limitations of the protocol and corresponding implementations. Except for those famous security threats that attract most researchers' attention, we also have to concede that there are still some privacy-disclosure problems which users and many institutions ignored. It is easy to see that the privacy leakage risk of users' DNS queries was overlooked. Users can find that all DNS messages are transmitted in the human-readable text. A malicious DNS server, which delegates each user's DNS queries in its domain, can gain statistical query information of everyone. Since this information can show users' private internet surfing interest and habit, hackers can lead to further attacks by the information such as IP spoofing against those *hot* IP addresses, which may cause commercial loss. Moreover, a passive attack, such as *eavesdropping*, or an active attack, such as $man-in-the-middle$, can carry out the same attacks as malicious DNS servers. Sharp increase in popularity (deduced from being a frequent target of DNS query) of a web-site may lead authorities to conclude that something "subversive" is going on. The problem can also manifest itself in less sinister settings. Many internet service providers keep detailed statistics and build elaborate profiles based on their clients' communication patterns [4]. If the surveillance result of DNS queries shows high popularity of some sub-domain names, the domain registrar can later reserve such popular names after their expiration date and sell them at higher prices. Finally, we can conclude that current DNS query protocol: by divulging both sources and targets of queries, represent yet another source of personal information that can be exploited either by eavesdroppers that reside in the quries' transmission route or potentially unscrupulous service providers.

In this paper, we discuss important privacy issues related to DNS and present a comprehensive survey of privacy-preserving methods provided by the existing approaches. We gave a full DNS privacy analysis and proposed the first random noise based range query approach in [36]. Then, towards improving the limitation of our first proposal, we discussed the Information Theoretic Private Information Retrieval (PIR) theory, and proposed a PIR based DNS query approach [37]. Inspired by our initial research in the issue of DNS privacy, more attention was attracted from lots of researchers. Castillo-Perez.S and Garcia-Alfaro.J's

Manuscript received November 24, 2009.

[†]The author is with Toshiba Corporate Research & Development Center, Kawasaki-shi, 212–8582 Japan.

[††]The authors are with the Dept. of Computer Science and Communication Engineering, Kyushu University, Fukuoka-shi, 819–0395 Japan.

a) E-mail: fangming.zhao@toshiba.co.jp

DOI: 10.1587/transinf.E93.D.1031

research group, in [38]–[40], later, carefully analyzed our original works, and proposed an enhanced approach based on our random noise range query schemes. Lu and G. Tsudik's research group in University of California proposed another random noise and DHT structure based DNS protocol to solve the same problem continually [41], [42]. We believe all of those efforts has accelerated the development of the next generation's DNS protocol which will be launched by the whole research community.

The rest of this paper is organized as follows: Sect. 2 analyzes two related works of this research: One is the DHT structure based DNS query protocol by the research group of UC [41], [42], the other one is the computational secure PIR. Section 3 gives a review of DNS and DNSSEC. Section 4 analyzes those security and privacy adversary threats. In Sect. 5, we introduce our first random noise range query protocol [36] in Sect. 5.2, then in Sect. 5.3 we discuss an enhanced approach [38], [40] which based on our range query protocol. In Sect. 5.4, we introduced our second protocol Information Theoretic PIR theory based DNS query protocol [37], [39]. Section 5.5 will give a carefully evaluation of four existing protocols. Finally, Sect. 6 concludes the paper and lists the remaining issues and gives further research direction.

2. Related Works

In this section, we discuss two related works of our research. For the first one, we introduce a DNS structure based DNS protocol which also aims at the privacy protection problem. The second related work we will discuss is another famous privacy preserving theory: *Computationally Secure PIR*, which is distinct from the *Information Theoretic PIR* used in our protocol (Sect. 5.4).

2.1 DHT Structure Based Approach: *PPDNS*

Lu and G. Tsudik's research group in University of California proposed another random noise and DHT structure based DNS protocol to study the same problem [41], [42]. Their work was constructed on the CoDoNS mechanism [6], which was named '*PPDNS*'. In this subsection, we will discuss their approaches.

PPDNS modified the basic DNS infrastructure to a hash space and DHT protocol based system. All clients and DNS nodes share a hash function that maps any domain name into a circular space which is based on the SHA1 [5] hash function. In this case, each nonempty domain name is mapped to an *identifier*. The basic method in *PPDNS* protocol is the same to our random noise based range query protocols: each target identifier should be perturbed by a group fixed noise before sent to name servers. Those fixed noise was named *confusing identifiers*. Each client has a security parameter m that represents the number of nonempty identifiers it expects in the query range. A notation ρ was defined in the system: the ratio of the number of nonempty identifiers to the total number of identifiers. All clients in the same domain could get ρ from the local name server and can also update it periodically.

If a client C wants to resolve a domain name, the process should be as follows: 1). C first computes the identifier i^* by the hash fuction. 2). C determines his security parameter m, which is the number of nonempty identifiers in his query range. 3). C constructs the range query, which is only comprised of the start and end identifiers for i^* by the formula:

$$Q(i^*) = \left[\left\lfloor \frac{i^*}{2^s} \right\rfloor \cdot 2^s, \quad \left(\left\lfloor \frac{i^*}{2^s} \right\rfloor + 1 \right) \cdot 2^s - 1 \right]$$

Here, the range size is computed as 2^s, $s = \log 2^{\frac{m}{\rho}}$. (for details about the security proof of this formula, please refer to their paper). 4). C sends the query range towards its destination node through the local name server by the DHT protocol. 5). All nodes pass the query onto the next intermediate node until it reaches the destination. 6). Until a node happens to own a cache of the queried range, it directly responds to C's local name server, and the query stops. In the range transmission process, range splitting can also occur and several subrange query may be generated at intermediate nodes. Those nodes on the transmission route will return all records to the local name server if anyone has complete records for a subrange query. In this case, the local name server caches all responses received from those intermediate nodes and send the full query range to C finally.

It is easy to see that, in their *PPDNS* protocol, the requirement of modifying the whole DNS infrastructure is the most obvious difficulty, what we sort as a drawback of usability. This absolutely new protocol also needs additional computation ability on both clients and servers sides to support the hash computation, query range generation and range splitting. As with other protocols, let's look at the bandwidth consumption and privacy-preserving effectiveness of this protocol. In the request process, since the range query is only comprised of two identifiers, which is the start and the end of the range, the consumption of the bandwidth is quite small. For comparing to other protocols easily, we just consider it as the consumption of one host name in the transmission route. In the response process, if the number of identifiers among the range query is n, the IP addresses that were returned to the local name server, should also be n. On the effectiveness of privacy-preserving side, the privacy disclosure probability should be $\frac{1}{n}$ in the request process, and $\frac{1}{n}$ also in the response process.

2.2 Computationally Secure PIR

The computationally secure PIR (CPIR) was proposed by B. Chor and N. Gilboa in their works [11] after the information-theoretic PIR. The main difference between these two classes (Information-Theoretic PIR and CPIR) is that information-theoretic privacy can be efficiently achieved only if the database is replicated at $k \geq 2$ non-communicating servers. On the contrary, in computational privacy setting, the replication of the database is not needed.

Here, two privacy definition were used:

Information-Theoretic Privacy The distribution of the queries the user sends to any server is independent of the index he wishes to retrieve. This means that each server cannot gain any information about user's interest regardless of his computational power.

Computational Privacy The distributions of the queries the user sends to any server are computationally indistinguishable by varying the index. This means that each server cannot gain any information about user's interest provided that he is computationally bounded.

The security of the CPIR scheme, however, rests on the assumption that certain computational problems are infeasible for the server. So now we no longer grant the server unlimited computational power. In practice, the security of all known CPIR schemes rests on number theoretic assumptions, most notably the Quadratic Residuosity Problem [12] and the ϕ-hiding assumption [15]. The Quadratic Residuosity Problem is to determine whether z is a quadratic residue modulo m, where $gcd(z, m) = 1$ and $z, m \in N$. Recall that z is a quadratic residue modulo m if there exists an integer a such that $a^2 \equiv z \bmod m$. The ϕ-hiding assumption also depends on a number theoretic concept. Recall that ϕ is the Euler totient function, where $\phi(m)$ gives the number of integers k such that $0 < k < m$ and $gcd(m, k) = 1$, i.e. m and k are relatively prime. Note that computing $\phi(m)$ on input m is just as hard as factoring m [15]. We say that a composite integer m ϕ-hides a prime p, if $p \mid \phi(m)$. The ϕ-hiding assumption then states that it is computationally infeasible to decide whether a small prime p divides $\phi(m)$, where m is a composite integer of unknown factorization. More information about number theoretic concepts in use can be found in [13] and [14]. Since the CPIR may be applied in DNS privacy problems in future, we just give a basic introduction in our paper. For more details about CPIR, please refer papers [10]–[12], [15].

3. Overview of DNS and DNSSEC

3.1 Overview of DNS

The Domain Name System (DNS) is a hierarchically distributed database that provides information fundamental to Internet operations, such as translating between human readable host names and Internet Protocol (IP) addresses. This database associates names, which are referred to as domain names, with certain data contained in resource records (RRs). Records linked to a domain name can be of different types, but the address type is the most common one. There can be multiple RRs of the same type for one domain name. The set of resource records of the same type is called a resource record set (RRset).

Since domain names need to be globally unique, a hierarchical naming scheme is used. A domain name refers to a node in a tree which is called the domain name space. This tree of domain names is very similar to the structure of a UNIX file system. Each subtree is called a domain. For example, the subtree rooted on the .com node is called the .com domain and includes all domain names ending with .com. The nodes that are direct children of the root node are called top level domains. Communication with the DNS database follows the client/ server paradigm. The domain name tree is divided into zones, which usually are contiguous parts of the tree. Zones are defined by the process of delegation which assigns to some organization the responsibility of managing particular subdomains. A zone may contain information about a domain and its subdomains. Top-level zones, such as .edu, would mostly contain delegation information. For each zone, there are authoritative servers answering all queries concerning domain names in that zone.

An authoritative name server is a name server that gives answers that have been configured by an original source, for example, the domain administrator or by dynamic DNS methods, in contrast to answers that were obtained via a regular DNS query to another name server. An authoritative-only name server only returns answers to queries about domain names that have been specifically configured by the administrator. In principle, authoritative name servers are sufficient for the operation of the Internet. However, with only authoritative name servers operating, every DNS query must start with recursive queries at the root zone of the Domain Name System and each user system must implement resolver software capable of recursive operation. Caching techniques are employed to reduce the number of requests in order to speed up the resolving process and to reduce network traffic. The Domain Name System supports DNS cache servers which store DNS query results for a period of time determined in the configuration (TTL, time-to-live) of the domain name record in question. Typically, such caching DNS servers, also called DNS caches, also implement the recursive algorithm necessary to resolve a given name starting with the DNS root through to the authoritative name servers of the queried domain. With this function implemented in the name server, user applications gain efficiency in design and operation. The combination of DNS caching and recursive functions in a name server is not mandatory, the functions can be implemented independently in servers for special purposes.

A DNS client program is called a resolver. There are two kinds of resolvers: stub resolvers and real resolvers. A stub resolver is basically a library that needs to be installed on every host that wants to access the DNS database. Every time a query needs to be sent, functions of this library are called and the process of retrieving the desired information is run. Specifically, the stub resolver sends a recursive query to a resolver which will reply with the information needed. A real resolver is generally located on a DNS server and serves a group of stub resolvers. When a recursive query is received, the resolver usually sends an iterative query to one of the root DNS servers serving the root domain. Iterative queries allow a DNS server, which does not have the requested mapping, to indicate the next server in the chain which is closer to the authoritative server for those queries.

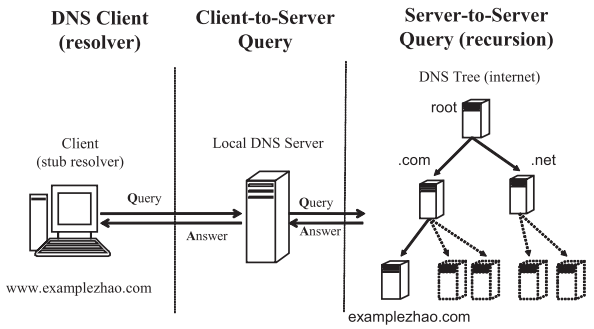


Fig. 1 DNS query process.

Root servers are essential to the functionality of the DNS system. There are currently 13 root DNS servers distributed all over the planet.

As the example shown in Fig. 1, the resolution of *www.examplezhao.com* involves following steps: First, the client types the domain name into the web browser which consults its local stub resolver. If the queried name can be resolved in the stub resolver, the query is answered and the process is completed. Else, the resolution process continues with the client querying a local DNS server to resolve the name. When the DNS server receives a query, if the queried name matches a corresponding resource record in local zone information, the server answers it authoritatively. Else, the server then checks to see if it can resolve the name using locally cached information from previous queries. If a match is found here, the server answers with this information. Again, if the preferred server can answer with a positive matched response from its cache to the requesting client, the query is completed. If the queried name does not find a matched answer at its preferred server-either from its cache or zone information, the query process can continue. This involves assistance from other DNS servers to help resolve the name. By default, the DNS client service asks the server to use a process of recursion to fully resolve names on behalf of the client before returning an answer. For more details about DNS query protocol and related information please refer RFC documents [17]–[19].

3.2 Overview of DNSSEC

The Domain Name System SECurity (DNSSEC) extension is a set of specifications of the IETF for guaranteeing authenticity and integrity of DNS Resource Records (RRs) such as NAPTR records. This is done via cryptographic electronic signatures signed with a trusted digital certificate to determine the authenticity of data. It is a specification of an extension to the DNS through the definition of additional DNS Resource Records that can be used by DNS clients to validate the authenticity of a DNS response, the data integrity of the DNS response, and where the response indicates no such domain or resource type exists, this negative information can also be authenticated. In other words, if an attacker attempts to create a DNS response that has been altered from the original authentic response in some fashion, and the attacker then attempts to pass the response off as an authentic response, then a DNSSEC-aware DNS client should be able to detect the fact that the response has been altered and that the response does not correspond to the authoritative DNS information for that zone. In other words, DNSSEC is intended to protect DNS clients from forged DNS data. This protection does not eliminate the potential to inject false data into a DNS resolution transaction, but it adds additional information to DNS responses to allow a client to check that the response is authentic and complete.

To achieve this, DNSSEC defines a number of new DNS resource records (RRs), namely the DNSKEY, RRSIG, NSEC and DS RRs, and two new message header bits: checking Disabled (CD) and Authenticated Data (AD), and it relies on functions provided by Extended DNS mechanisms (EDNSO). With DNSSEC a zone administrator "digitally signs" a Resource Record Set (RRSet), and publishes this digital signature, along with the zone administrator's public key, in the DNS. In checking a DNS response, a DNSSEC client can retrieve the related RRset digital signature and then check this signature using the public key against the locally calculated hash value of the RRset, and then validate the zone administrator's public key against a hierarchical signature path that leads to a point of trust. If all these checks succeed then the client has some confidence that the DNS response was complete and authentic. DNSSEC implies different actions for different roles. For a DNS zone administrator, DNSSEC is essentially the process of signing RRSets with a private key, publishing these signatures for each RRset in the zone file, and publishing the zone public key in the zone file. In addition the zone administrator has to get the zone's public key signed by the parent zone administrator. For a DNS client DNSSEC is the ability to perform a number of additional checks on a DNS response that can result in greater trust in the authenticity and accuracy of the DNS response. And for the DNS itself DNSSEC essentially represents a number of additional Resource Records that hold digital signatures of DNS information, as well as key information. For details of DNSSEC, please refer to RFC files [30]–[32].

Recently, one research point that attracts some researchers' attention about DNSSEC is the reliability problem from communication under the UDP protocol, which means that DNSSEC increases the UDP payload length of the server response and the IP fragmentation of the UDP datagram may undermine the reliability of communication. Details of the research of communication reliability in DNSSEC are well discussed in paper [35].

4. DNS Threats Overview

Although invented in the early days of the Internet, DNS's design is such that it manages to be scalable to the size and the dynamics of the Internet in present days. The immense growth of the Internet was not foreseen, and the scalable design did not take the abuse patterns that come with that into account. DNS stakeholders need to be aware of current

limitations of the protocol and corresponding implementations. Except for those famous security threats that attract most researchers' attention, at the same time we also have to concede that there are still some privacy-disclosure problems which are ignored by many users and institutions. In this section, we describe and discuss DNS threats in two categories respectively: security threats and privacy threats. As we have mentioned, the second part: Privacy Threats Analysis, is our main target in this paper.

4.1 DNS Security Threats

In this part, we mainly introduce some threats that happened most often which related to the security issues [27]–[29].

IP Spoofing is the creation of IP packets with a forged (spoofed) source IP address with the purpose to conceal the identity of the sender or impersonating another computing system.

Cache Poisoning is a technique that tricks a DNS server into believing it has received authentic information when, in reality, it has not. Once the DNS server has been poisoned, the information is generally cached for a while, spreading the effect of the attack to the users of the server.

Query Prediction Because DNS uses the connectionless UDP for its transport, DNS packets can be forged by an attacker. So it is possible to make correct guesses and create a 'legitimate' DNS reply.

DoS Attack A Denial of Service attack is executed from one attacking host to one victim host. The attacking host will try to consume as much resources from the victim or the infrastructure leading to the victim's network, so that the service to normal users is degraded. This attack are either aimed at a specific service (like DNS) or aimed wider to a whole part of the network.

Distributed DoS Attack Distributed DoS attacks share many characteristics with normal DoS attacks. However, we make the distinction here that DoS attacks are executed from one host, and Distributed DoS attacks are executed from multiple 100 s or 1000 s of hosts.

Man − in − the − Middle A man-in-the-middle attack is a general description for attacks that are executed when an attacker residing between a DNS server and a client. The attacker therefore has knowledge about the connection and can use that to eavesdrop on the connection or even inject data into it.

As discussed in Sect. 3.2, DNSSEC uses cryptographic electronic signatures signed with a trusted digital certificate to determine the authenticity of data. It can provide: a) origin authentication of DNS data, b) data integrity, and c) authenticated denial of existence. However, we found that DNSSEC is not omnipotent that they are not appropriate for solving our motivation problem - Privacy Leakage (DNSSEC also can not solve (*D*) *DoS attacks*, we will not discuss this problem in this paper). In the next part of this section, we mainly analyze several privacy disclosure risks which DNSSEC can not solve.

4.2 DNS Privacy Threats

As we have mentioned, the privacy-disclosure problems were ignored when the critical internet infrastructure component *DNS* was designed in the early eighties. Since all DNS messages are transmitted in the human-readable text, the privacy leakage could happen during the whole DNS query process. In this paper, we are concerned with those two kinds of privacy related attacks.

- A *passive attack* means that the adversary is restricted to observing the channel. That is he is allowed to read DNS messages passing over the network, but he cannot make any modifications.
- An *active attack* means that the adversary can observe the channel and also add, delete or modify any DNS messages that pass over the channel.

Eavesdropping belongs to the *passiveattacks*. In this attack, the adversary configures the respective network interface in promiscuous mode; the adversary's computer receives any DNS packets sent on the network, including DNS query packets destined for other nodes. Since packets are unencrypted, the attacker can read packets' data, which includs users' privacy related personal web accessing information. Many available applications can be used for eavesdropping, including tcpdump [25] and wireshark [26].

Man in the Middle (*MITM*) belongs to *active attacks*. Since the adversary resides between users and DNS servers, he can read, modify, inject, or drop any packet. In this case, for example, by the statisical result of an user's web accessing habit, the adversary can choose some special sites such as internet banks as their attack targets. The adversary previously makes a fake bank site that quite similar to the one that the user always uses. When the user want to access the bank site by resolving the URL, the adversary could modify the DNS packet and send back a false IP address which links to fake bank site, and finally got the user's account information. (This could also be another type of IP spoofing attack)

Honest − but − Curious (*HBC*) *Server*, that is, the DNS service providers responsible for the service are expected to provide the required services (name resolving), but any employees that work for such providers could steal user's DNS query statistical data and profit from it. Those DNS servers can gain privacy related statistical query information of everyone in their domain. Moreover, Sharp increase in popularity (deduced from being a frequent target of DNS query) of a web-site may lead authorities to conclude that something "subversive" is going on. Many internet service providers keep detailed statistics and build elaborate profiles based on their clients' communication patterns [4]. If the

surveillance result of DNS queries shows high popularity of some sub-domain names, the domain registrar can later reserve such popular names after their expiration date and sell them at higher prices.

One practical DNS privacy related problem was well disscussed in [38] by S. Castillo-Perez's group. By their analysis, privacy disclosure of DNS queries happens in the use of DNS on ENUM (Elephone NUmber Mapping) protocols of VoIP (Voice over IP) for the translation of traditional telephone numbers into internet URLs. The ENUM service is a set of protocols used in VoIP applications whose main goal is the unification of the traditional telephone E.164 system with the IP network of the Internet. Designed and developed by the Internet Engineering Task Force (IETF) in late nineties, ENUM allows the mapping of IP services by using an indirect lookup method based on DNS technologies. In this manner, by simply using existing DNS implementations, ENUM allows retrieving lists of IP based services, such as SIP (Session Initiation Protocol) identifiers for VoIP applications, e-mail addresses, Web pages, etc., associated to the principal of an E.164 telephone number. Instead of resolving host or service names into IP addresses, the ENUM service also translates E.164 telephone numbers into Uniform Resource Locators (URLs) embedded within NAPTR records. For a more detailed introduction to the suite of protocols associated with ENUM, we refer the reader to [38]. Obviously, vulnerabilities on the DNS allowing the disclosure of data associated with people's information, such as their telephone numbers, is a critical privacy threat. The worst case scenarios that HBC servers start keeping statistics of ENUM queries and building people's profiles based on their communication patterns, may lead to further violations, such as spam, scams, untruthful marketing, etc.

Nowadays, more and more people are becoming increasingly concerned about the privacy of their personal data. They would like to avoid giving out much more about themselves than is required to be aggregated by the DNS service. However, unfortunately, there are few specific approaches that aim at this kind of privacy disclosure problem. In the next section of this paper, we will introduce our original works towards the DNS privacy threats based on *Range Query* [36] and *Information Theoretic PIR theory* [37]. Then, we will also introduce an extension work based on *Range Query* and DNSSEC.

5. Privacy-Preserving Domain Name System

After the discussion on the DNS privacy threats in Sect. 4, we introduce our privacy-preserving DNS protocols which is mainly based on our papers in MUE2007 [36] and IPC2007 [37]. At the same time, we will also give a survey of the enhanced approaches [38]–[40] of Castillo-Perez.S and Garcia-Alfaro.J's research group based on our initial approaches. This section is organized as below: In Sect. 5.1, we define important notations and conventions that will be used throughout this section. In Sect. 5.2, we will introduce our first random noise range query protocol. In Sect. 5.3, we introduce an enhanced approach which is based on our original protocol, from Castillo-Perez.S and Garcia-Alfaro.J's research [38], [39]. In Sect. 5.4, towards several flaws of our original approach and the enhanced approach, we will introduce our second protocol - Information Theoretic PIR theory based DNS query protocol. Finally, in Sect. 5.5, we give a full evaluation of all these existing protocols.

5.1 Preliminaries

Following notations and conventions are used throughout our protocols.

- $DSer_m$: DNS servers
- U: a client user
- H_i: host name
- n: a privacy requirement parameter for the user, be decided by users
- $Q\{H_i\}_{i=1}^{n}$: a range/group of queries (host names)
- $IP\{IP_i\}_{i=1}^{n}$: a range/group of IP addresses
- P_i: probability of successful guessing
- A_i: answer from the DNS server
- X_{ij}: a single bit from IP_i
- DBL_{Client}: a database/library that stores lots of host names which can be classified by their categories. In addition, data items (host names) stored in it will be updated periodly to keep their validities
- $f_r(H)$: randomization generation function. $f_r(H)$ will generate the random noise from the DBL_{Client}

5.2 Random Ranges Based DNS Query Protocol

This random noise range query approach is from our work in MUE2007 [36]. The basic idea behind range query is very simple, even trivial: instead of querying by a specific host name, the client queries a range/group of host name $Q\{H_i\}_{i=1}^{n}$. It means the query should include two or more host names, and only one name in the range is the host's target, other $(n-1)$ host names should be generated randomly from a local periodly-updated database DBL_{Client}, which can also be called the random noise. In this approach, the client could protect the privacy of his/her DNS query data by perturbing it with a random noise in the query range $Q\{H_i\}_{i=1}^{n}$. The randomization algorithm of the user is chosen so that aggregate properties of the data can be recovered with sufficient precision, while individual entries are significantly distorted.

5.2.1 Protocol Details

This section gives a description of the protocol:

1. U decides a privacy requirement parameter n of his website-browsing.
2. U inputs the H that he/she wants to access, and at the same time automatically runs the $f_r(H)$.

3. $f_r(H)$ generates the $Q\{H_i\}_{i=1}^n$ from the DBL_{Client}, and inserts H to it to get the $Q\{H_i\}_{i=1}^n$.
4. U sends the $Q\{H_i\}_{i=1}^n$ to DSer.
5. DSer Collects all the IP addresses according to the $Q\{H_i\}_{i=1}^n$, then sends the $IP\{IP_i\}_{i=1}^n$ back to U.
6. U picks up the target IP, and then accesses the website.

5.2.2 Protocol Analysis

The adoption of random noise in the range query model is fresh and applicable, in here we call it "user-defined" randomization, which means as specific users, user could define the privacy protection requirement parameter n by themselves. By adopting the randomization function $f_r(H)$ to generate query ranges from the *periodly − updated* DBL_{Client}, users not only could make sure the validity of those host names in the ranges (some *url* may not be used after a period), but also could leave less burthen to the client than using the 'heavy' $fixed - range$ method which means for a given target host name, the protocol generate a fixed range of noise and remember the fixed range by the cache. We find the randomized range generation method is more efficiency than the 'fixed-range' method from the viewpoint of cache costs. For personal users, we have said the range query function $f_r(H)$ should be integrated in the local anti-virus software, as a "Higher Security Option", which could be selected freely by those users with different privacy requirement; For group users, such as a research institute, where there is totally trust between each others and the local DNS server in the group, we suggest the range query should be integrated in the local DNS server or a proxy, the server or proxy will run the Range Query protocol to take care of the privacy of the whole group.

For an optimistic situation with random noise range query approach, the only information that is divulged to the server and other third parties is that the target query lies in the interval [1, n] which translates into the probability of correctly guessing i: $P_i = \frac{1}{n}$. We can also deduce the bandwidth cost of this approach: Let us consider the number of host name query and the response delegating the consumption of bandwidth. So the bandwidth consumption could be both expressed as n. Since this approach does not need to change the DNS infrastructure, we also suggested using our approach in conjunction with the DNSSEC technonogy, which can also provide authenticity and integrity to users. Later, our enhanced suggestion was adopted by Castillo-Perez.S and Garcia-Alfaro.J's research group. As the subsequence of our initial work, they completed our idea and gave a perfect implementation and evaluation of both two works in [38]–[40]. We will give a description of the enhanced approach in the next section.

5.3 Enhanced Protocol: DNSSEC Based Random Ranges

Castillo-Perez.S et al. gave a carefull security analysis of our random noise range query approach in the paper of [38]–[40]. They proved our protocol could be attacked by a kind of active attacks (Sect. 4.2) in their works, and they also gave a DNSSEC based enhaced approach. We introduce their works in this section.

5.3.1 Active Privacy Reduction Attack

The active privacy reduction attack happens when the range query $Q\{H_i\}_{i=1}^n$ sent by U fails. If active attackers can manipulate the network traffic by RST attacks [22] or by using the ICMP traffic method [23], or by controlling the local DNS server, they could launch an privacy reduction attack against our random noise based range query protocol. The method of the active privacy reduction attack is based on dropping the query range $Q\{H_i\}_{i=1}^n$ repeatedly. If the range query $Q\{H_i\}_{i=1}^n$ that is sent by U fails, U will be forced to restart the random noise generating process and get a new range query for the real target by the protocol in Sect. 5.2. Here, each new range query is diffrent from previous ones by the function $f_r(H)$ for the reason of efficiency (as disscussed in Sect. 5.2.2). Let's assume the curious adversary drops $Q\{H_i\}_{i=1}^n$ j times, which means U must generate random range $Q\{H_i\}_{i=1}^n$ j times then send to the local server respectively after each failure: $Q_1\{H_i\}_{i=1}^n$, $Q_2\{H_i\}_{i=1}^n$, $\cdots$, $Q_j\{H_i\}_{i=1}^n$. In this case, Adversaries either residing the network or controlling the local server could guess the target in the successful probability P_j by the intersection result of each query range.

$$P_j = \frac{1}{|Q_1\{H_i\}_{i=1}^n \cap Q_2\{H_i\}_{i=1}^n \cap \cdots \cap Q_j\{H_i\}_{i=1}^n|}$$

Here, $Q_j\{H_i\}_{i=1}^n$ means the j-th consecutive range exchanged for the resolution of the query.

By using the following ideal scenario, let us exemplify the probability to successfully predict the target H under this active privacy reduction attack. In a simple scenario, let assume the $U's$ database $DBL_{Client} = \{H_1, H_2, H_3, H_4, H_5, H_6\}$, and the real target of U is H_2. By the protocol in Sect. 5.2.1, U first perturbs the target by the random noise H_1 and H_6 from the DBL_{Client}, then U sends $Q_1 = \{H_1, H_2, H_6\}$ to the local DNS server. At this moment, the adversary could guess the $U's$ target with the successful probability $P_{i1} = \frac{1}{3}$. By active attacks as we have mentioned, the adversary catches the query range Q_1, and drops it. Then, U has to construct a new random range by the random noise H_1 and H_5 from the DBL_{Client}, $Q_2 = \{H_1, H_2, H_5\}$. By the same active attack, the adversary catches the Q_2 again and calculates the intersection between the previous range and the current one, $Q_1 \cap Q_2 = \{H_1, H_2, H_6\} \cap \{H_1, H_2, H_5\} = \{H_1, H_2\}$. Consequently, the successful guessing probability is up to $P_{i2} = \frac{1}{2}$. Now, let's assume the adversary successfully carries out the active attack again toward $U's$ $Q_3 = \{H_2, H_3, H_5\}$, then the target H_2 could be deduced from the calculatioin of intersection: $P_{i3} = Q_1 \cap Q_2 \cap Q_3 = \{H_2\} = 1$.

5.3.2 Enhaced Protocol Details

To overcome the active privacy reduction attack in our first protocol, Castillo-Perez.S et al. presented an enhanced protocol. The new protocol, by adding several processes to our original approach, can also provide authenticity and integrity mechanisms on DNS procedures.

In addition to using the DNSSEC to the new approach, they modified several processes as follows:

- Number of servers which receive range query should be more than one: $\{DSer_1, \cdots, DSer_m\}$
- U should use the $f_r(H)$ generating a group of Q_k, which $k \in [1, m]$.
- The size of each range query should still be $|Q_k| = n$
- $f_r(H)$ generates Q_k from DBL_{Client}, such that $\cap_{i=1}^{n} Q_k \{H_{ki}\} = \emptyset$, and $\cap_{k=1}^{n} Q_k \{H_{ki}\} = \emptyset$
- U sends the Q_k randomly to different servers $DSer_\omega$, $\forall\omega \in [1, m]$.
- Each server, $DSer_\omega$, $\forall\omega \in [1, m]$, should enable DNSSEC protocol.
- U verifies DNSSEC signatures of all responses from $DSer_\omega$ until all responses are received correctly.
- U picks up the target IP address and discards all the noise responses.

As the result of the enhanced approach, for obtaining a query $Q\{H_i\}$ within a range of size n from m different servers, the successful guessing probability could be optimized to $P'_i = \frac{1}{n \cdot m}$. Castillo-Perez.S et al. also implemented and evaluated the enhanced protocol in [39] on a real network scenario, with the Python language based *dnspython* [20]. They also use the MeTooCrypto [21] and OpenSSL library to apply the verification of digital signatures defined by DNSSEC. For more details about their implementation, please refer their papers. Their evaluation was divided to four stages (Table 1): *a* means the implementation environment is under DNS and TCP protocols. *b* means the implementation environment is under DNSSEC and TCP protocols. *c* means the implementation environment is under DNS and UDP protocols. *d* means the implementation environment is under DNSSEC and UDP protocols.

After analyzing the performance evaluation of these 4 kinds of combination [39], we found that 1): the utilization of DNSSEC instead of DNS in our random noise based range query protocol is both acceptable and valuable (by comparing the result between stage a, b and c, d). The result shows that DNSSEC is well suited to our protocol by adding the integrity and authentity. 2): We could get a much better performance of our protocol under the UDP instead of TCP. The result shows that the average resolution time could be reduced more than 6 times if we use the UDP protocol. The reason is legible: TCP based experiments show worst performance than UDP based queries - due to the penalty imposed by the traffic that guarantees the delivery of packets. Obviously, among four stages of implementation, the protocol that under both DNSSEC and UDP is the best choice for the wide deployment in future.

Table 1 Four stages of the implementation and evaluation.

Stage \ Protocol	DNS	DNSSEC	TCP	UDP
a	○		○	
b		○	○	
c	○			○
d		○		○

Recently, the reliability problem of communication attracts many researchers attention when DNSSEC is implemented under UDP protocol, which means that DNSSEC increases the UDP payload length of the server response and the IP fragmentation of the UDP datagram may undermine the reliability of communication. We consider that the same problem also happens in our enhanced protocols. Since that research is out the scope of our paper's motivation, we do not discuss the reliability problem in here. Please refer the paper of Rikitake. K's research group [35].

5.3.3 Protocol Analysis

We analyze both the original protocol and the enhanced protocol in this section. When considering the effectiveness of privacy-preserving, we found that the original protocol could protect the users' privacy with the successful guessing probability $P_i = \frac{1}{n}$ under the optimal case (without active attacks). If active attacks were carried out by adversaries, the successful guessing probability would be increased to $P_j = \frac{1}{|\cap_{j=1}^{n} Q_j\{H_i\}_{i=1}^{n}|}$. By the enhanced protocol, we could get both a more appropriate successful guessing probability $P'_i = \frac{1}{n \cdot m}$ and the guarantee of authentity and integrity of DNS data.

By the evaluation results of the enhanced protocol, it is obvious that the penalty of adopting the DNSSEC is not unacceptable when considering the result of security guarantee. However, we found that both protocols have got unpleasant side-effect on the bandwidth consumption. Let us consider the number of host name query and the response delegating the consumption of bandwidth. The bandwidth consumption of the original protocol was proved to be both expressed as n in the Sect. 5.2.2. In the same way, the bandwidth consumption of the enhanced protocol could be both expressed as $m \cdot n$. Which is to say, the enhanced protocol gained a privacy guarantee of m times by also sacrificing the bandwidth consumption m times. In next section, we will introduce our information theoretic PIR based proposal which is towards decreasing the bandwidth consumption.

5.4 Information Theoretic PIR Based DNS Query Protocol

For solving the bandwidth consumption cost which was discussed in the previous sections, we propose a new query model which is based on the information-theoretic PIR theory in this section. We first take a look at the definition of

the PIR theory, and then we introduce the protocol details and the performance evaluation. Finally, we give a careful analysis of this protocol comparing to protocols discussed in Sect. 5.2 and 5.3.

5.4.1 Information Theoretic PIR

The notion of private information retrieval (PIR) was introduced by Chor, Goldreich, Kushilevitz and Sudan [9] and has already received a lot of attention from various cryptographic research. The study of PIR is motivated by the growing concern about the user's privacy when querying a large commercial database. Recently, the problem was also studied by D. Boneh and R. Ostrovsky [7], [8] to implement an anonymous searchable encryption service for cryptographic filesystems, and T. Nakamura et al. also apply the PIR theory to a novel anonymous authentication system [10]. Next, we will give a formal definition of the information-theoretic PIR theory as follows:

Definition 2.2.1 A one-round, $(1-\eta)$-secure, k-server private information retrieval (PIR) scheme for a database $x \in \{0,1\}^n$ with recovery probability $1/2+\epsilon$, query size t, and answer size ι, consists of a randomized algorithm (user) and k deterministic algorithms $S_1, \cdots, S_k$ (servers), such that

1. On input $i \in [n]$, the user produces k t-bit queries $q_1, \cdots, q_k$ and sends these to the respective servers. The jth server sends back an ι-bit string $a_j = S_j(x, q_j)$. The user outputs a bit $f(a_1, \cdots, a_k)$ where f depends on i and his randomness.
2. For every $x \in \{0,1\}^n$ and $i \in [n]$ we have $Pr[f(a_1, \cdots, a_k) = x_i] \geq 1/2 + \epsilon$
3. For all $x \in \{0,1\}^n$, $j \in [k]$ and any two indices $i_1, i_2 \in [n]$, the two distributions on q_j (over the user's randomness) induced by i_1 and i_2 are $\eta - close$ in total variation distance.

We say that the scheme uses b bits, if the user only uses b predetermined bits from each query answer of length ι: he outputs $f(a_{1|s_1}, \cdots, a_{k|s_k})$ where the sets $S_1, \cdots, S_k$ are of size b each and are determined by i and the user's randomness.

The scheme is called linear, if for every j and q_j the jth server's answer $S_j(x, q_j)$ is a linear combination (over GF(2)) of the bits of x.

The setting $\eta = 0$ corresponds to the case where the server gets no information at all about i. All known non-trivial PIR schemes have $\eta = 0$, perfect recovery ($\epsilon = 1/2$), and only one round of communication. Servers are not allowed to communicate. We furthermore assume a secure channel between the user and the servers, i.e. a server cannot monitor transmissions to and from another server.

Using $k \geq 2$ non-communicating servers allows for PIR with less than n bits of communication. Each of the k servers has a copy of the n-bit database x. The individual server should learn nothing about i, even if it has unlimited computational resources. Since the k servers are not allowed to communicate with each other, this gives information-theoretic privacy for the user. To retrieve an item from the database, the user is allowed to send a query q_j to database j, which will send back an answer a_j. The user now selects b bits of each answer and combines them to compute the value of x_j. We will show the details of how our protocol works in the case of $k = 2$, which means we use two local servers to protect users' privacy.

5.4.2 Two-Servers PIR Based DNS Query Protocol

In this section, we introduce the two-servers PIR based DNS query protocol. The basic idea behind this scheme is to distribute the random noise based query range to two local DNS servers separately. The difference between two ranges is that the target H is inserted in either ranges, such that $Q\{H_i\}_{i=1}^{n}$, $Q_2\{H_i\}_{i=1}^{n+1}$, where H $\in Q_2$ is the desired query defined by U. Once Q_1 and Q_2 are generated by $f_r(H)$ from DBL_{Client}, such ranges are sent towards two independent name servers: $DSer_1$ and $DSer_2$. Assuming the resolution of DNS queries of type A, each server resolves every query linked with its range and obtains all the associated IP addresses that related to two query ranges. Then $DSer_1$ computes $A_1 = \sum_{i=1}^{n} \oplus X_i$ and $DSer_2$ computes $A_2 = \sum_{i=1}^{n+1} \oplus X_i$. Both A_1 and A_2 are send back to U, who can privately retrieve target IP using the logical operation exclusive disjunction (XOR): $A_1 \oplus A_2$ (U XORs answers of two servers' response). Our protocol must strongly abide by the assumption from PIR theory:

- Collusion does not exist between two local servers
- No such adversaries can monitor two query ranges from two local servers simultaneously

This protocol executes the following steps as shown in Fig. 2:

1. U generates two query ranges Q_1 and Q_2 from the DBL_{Client} that satisfying: $Q_2\{H_i\}_{i=1}^{n+1} = Q_1\{H_i\}_{i=1}^{n} \cup H$ (H is $U's$ target). $Q_1\{H_i\} \cap H = \emptyset$, $\cap_{i=1}^{n} Q_1\{H_i\} = \emptyset$, $\cap_{i=1}^{n+1} Q_2\{H_i\} = \emptyset$, $|Q_1| = n$, $|Q_2| = n+1$.

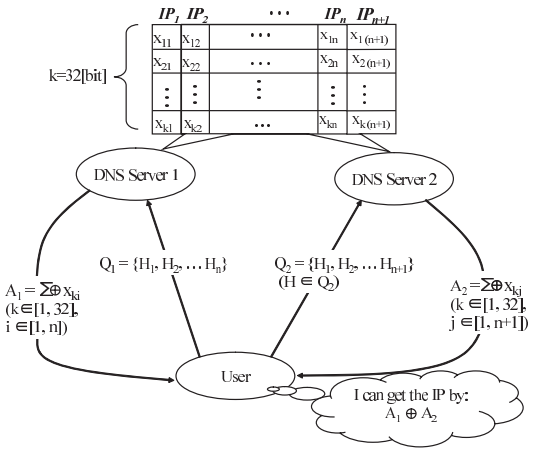


Fig. 2 Two-servers PIR based DNS Query protocol.

2. U sends $Q_1\{H_i\}_{i=1}^{n}$ and $Q_2\{H_i\}_{i=1}^{n+1}$ to $DSer_1$ and $DSer_2$ respectively.
3. $DSer_1$ and $DSer_2$ collect all IP addresses $IP\{IP_i\}_{i=1}^{n}$ and $IP\{IP_i\}_{i=1}^{n+1}$ according to the $Q\{H_i\}_{i=1}^{n}$ and $Q_2\{H_i\}_{i=1}^{n+1}$.
4. $DSer_1$ and $DSer_2$ compute $A_1{=}\oplus\sum X_{ki}$, $A_2{=}\oplus\sum X_{kj}$ ($k \in [1, 32], i \in [1, n], j \in [1, n+1]$, we assume the length of IP is 32 bits), then send results A_1 and A_2 to U.
5. U retrieves the target IP by $A_1 \oplus A_2$.

5.4.3 Performance Evaluation and Analysis

In order to implement this protocol, we should add a few functions to custom local servers for adopting the two-servers PIR protocol on servers side. On clients' side, we should add the XOR computing fuction for retrieving the IP address after receiving two responses from two local servers. Castillo-Perez.S et al. also give a implementation for this protocol in their papers [38], [39]. The implementation is also based on both TCP and UDP protocols for a performace comparision. The main core of the DNS resolution is also based on the *dnspython* [20] module. For the servers' side, the source for NSD server version 3.1.1 was adopted and modified for the two-servers PIR protocol. Please refer their papers for much detail about the implementation method.

After analyzing the performance evaluation, we could find that the performance of two-servers PIR based protocol under TCP protocol is not quite well, and the reason has been discussed in Sect. 5.3.2. For the case of UDP, the latency increases linearly with the size of the range of queries, even with the *timeout strategy* added (*Timeout strategy*: if any datagram appears to be missing, after the timeout expires, both servers and clients should send the same request or response again, which will also cost some external time). Another reason for the heavier time consumption of this protocol should be the *XOR* computation both on the servers' side and the clients' side, resulting from the adoption of PIR mechanism. The time consumption of the servers' side increases linearly with the size of the range of queries. However, the consumption in clients' side will not change because no matter how large query ranges are, the slight computation of any client is only 'XOR' two 32 bits's data strings.

When consider the privacy protection capability, we found the protocol is quite special from those two protocols introduced in Sect. 5.2 and 5.3. First, in the queries' request process, by using two local servers, we could decrease the privacy disclosure probabilities separately against eavesdroppers and each single server that: $P_{server1} = 0$, $P_{server2} = \frac{1}{n+1}$. So the total probability can be calculated by the *Law* of *Total Probability* [16] (sometimes also be called *Law* of *Alternatives*):

$$P_r(A) = \sum_n P_r(A \cap B_n) = \sum_n P_r(A|B_n)P_r(B_n)$$

By this law, the total probability of privacy disclosure can be expressed as: $P = \frac{1}{2n+2}$. ($P = P_1 \cdot \frac{1}{2} + P_2 \cdot \frac{1}{2} = \frac{1}{2n+2}$).

Second, in the query response process, since both responses from local servers are *XOR*'s computing results of a range IP address, they disclose nothing of the target to eavesdroppers. Only the client, who can receive both responses from two local servers, can retrieve the target IP address by *XOR* two responses. Finally, since this protocol is depending on the security assumption of PIR thoery, we acknowledge the protocol will be invalidating if an adversary can eavesdrop both query transmission channels, or both servers are controlled by the same party. All of these discussions must be under the assumption of Information-Theoretic PIR theory in Sect. 5.4.2.

5.5 Evaluation of Existing Protocols

After giving a survey of three existing privacy-preserving DNS protocols in Sect. 5.2, 5.3, and 5.4, we compare their effectiveness and usability in this section.

5.5.1 Effectiveness of Protocols

Since the motivation of this research is privacy protection in DNS protocols, the effectiveness here means protocols' strength towards privacy adversaries. For all these three protocols, we say each of them has its strong point.

In the first protocol, which is our original proposal, as discussed in Sect. 5.2, we found it is the most cost-effective one of all three protocols. With the smallest cost (requirement from clients&servers), it could achieve a reasonable privacy disclosure probability $P = \frac{1}{n}$ in the ideal case when the *active attack* does not happen. Even if active attackers carry out their aggression accidentally, this protocol can still react on its position to some extent with a appropriate parameter n.

In the second protocol, which is the enhanced protocol of our original one, it successfully prevents the possibility of *active attack* that might happen in our first protocol. By sending random noise based range queries to m servers, the privacy disclosure probability was decreased to $P = \frac{1}{m \cdot n}$. Additionally, this protocol also can provide the data integrity and authentity by associating with DNSSEC protocols. We say the enhanced protocol provides better privacy&security guarantee than the original one.

In the third protocol, which is the PIR theory based approach, it provides users a distinctive privacy guarantee than those two protocols before. Under the assumption of information theoretic PIR theory, by adding another 'deceptive' local DNS servers, not only could this protocol provide privacy protection in the query request process: privacy disclosure probabilities $P_1 = 0$, $P_2 = \frac{1}{n+1}$ separately against eavesdroppers and each single server, but it can also provide perfect privacy protection in the query response process: no information is divulged about the target IP address. A complete comparison of all four protocols that were introduced in this paper is shown in Table 2.

Table 2 A comparison of bandwidth consumption.

Protocol \ Consumption	Request	Response
Original Protocol (sec 5.2)	n	n
Enhanced Protocol (sec 5.3)	$m \cdot n.(m \geq 1)$	$m \cdot n$
PIR based Protocol (sec 5.4)	$2n + 1$	2
PPDNS Protocol (sec 2.1)	1	n

Table 3 A comparison of privacy disclosure probability.

Protocol \ Probability	Request	Response
Original Protocol (sec 5.2)	$\frac{1}{n}*$	$\frac{1}{n}*$
Enhanced Protocol (sec 5.3)	$\frac{1}{m \cdot n}$ $(m \geq 1)$	$\frac{1}{m \cdot n}$
PIR based Protocol (sec 5.4)	$\frac{1}{2n+2}$	0
PPDNS Protocol (sec 2.1)	$\frac{1}{n}$	$\frac{1}{n}$

$*$:If the *active attack* happens, the value would be $\frac{1}{|\cap_{j=1}^{n} Q_j[H_i]_{i=1}^{n}|}$.

5.5.2 Usability of Protocols

From the history of security research, many wonderful security and privacy enhancing techniques have been proposed and launched by the research community only to quietly fade into obscurity due to usability issues. Whether a protocol can be widely employed should be mostly decided by its general usability. In the issue of usability, we discuss it from two folds: bandwidth consumption and computation cost.

All these three protocols depend on the random noise in the range query. The bandwidth consumption drawback is directly determined by the size of queries' range. In these protocols, a bigger range size equals a higher guarantee of privacy protection capability. By our analysis in the first two protocols (Sect. 5.3 and 5.4), the bandwidth consumption increases linearly with the size of the range of queries in both the request process and the response process, which means the bandwidth consumption is inversely proportional to the prediction probability. The third protocol, which is based on the information theoretic PIR theory, could only improve the bandwidth consumption drawback partially (only in the response process). We show the bandwidth consumption relationship of all three protocols in Table 3.

For the issue of computation cost, we found all three protocols cost both clients' and servers' computation resource. Our first original protocol cost the least CPU resource of both clients and servers, because there is no more computation request on both sides except for the random noise generation and multiple queries' resolution. In the second enhanced protocol, not only the cost of the random noise generation was enlarged m times, but the cryptographic processing from the DNSSEC also costs some computing resource on both sides. Finally, in our third PIR based query protocol, in addition to the random noise generation and multiple queries' resolution, there is a new computing request for the *XOR* on both servers' and clients' sides. However, by the performance evaluation results [39] discussed in our paper, we found for the hardware of computers nowadays, the cryptographic computation cost does not impact too much in our protocols. We think that this problem could be ignored because the fast development of powerful PC hardware.

By our discussion about the Effectiveness and usability issues in this section, we can conclude that the reason which may restrict widely using of existing privacy-preserving DNS protocols might be the drawback on bandwidth consumption with a higher probability.

6. Concluding Remarks and Future Works

In this article, we have analyzed four existing privacy-preserving DNS protocols recently proposed in the literature. The preservation of privacy is achieved by introducing noise during the execution of DNS queries in all the four approaches. After carefully studying the effectiveness and usability of each protocol both by the performance evaluation and by the theoretic proof, we found each of these approaches has its strong point. However, at the same time, all of them also have their limitation to some extent. We found that one of the most important factors that may block those approaches to be widely applied is the bandwidth consumption. Although we gained some ideal results for the privacy preservation by the adoption of random noise mechanism, we finally paid a bandwidth consumption price for noise query ranges. Moreover, two approaches: PPDNS (Sect. 2.2) and Information Theoretic PIR based Protocol (Sect. 5.4) that were discussed in our paper need to modify the significant DNS infrastructure in some measure, such as they require some function should be enhanced in both clients and servers sides. These requirements will become new difficulties when they are widely applied.

In our future work, we are interested in finding a perfect method that neither consumes too much bandwidth relatively nor requires significant modification of those existing protocols. As mentioned earlier in the paper, privacy in DNS is unfortunately ignored by the majority of Internet users. For the reason, finding simple and unobtrusive ways of making average users aware of both the need for effective DNS protocols and the need to protect their privacy is a major challenge. We hope all of our work could be an initial step in this line of research and would attract more attention from the whole research community.

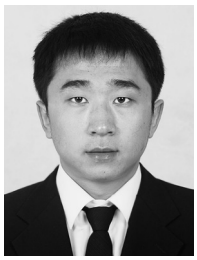
Acknowledgments

We thanks Dr. Takashi Nishide for carefully reading the paper and providing detailed suggestions on protocol design and comments on the English writing. Also many thanks to the anonymous reviewers in the IEICE editorial committee for their insightful comments on this work.

References

[1] Privacy International, Overview of privacy, 2006. Available at: http://www.privacyinternational.org/article.shtml?cmd%5B347%5D=x-347-559474

[2] Council of Europe: Convention for the protection of individuals with regard to automatic processing of personal data, 1981. Available at: http://conventions.coe.int/Treaty/en/Treaties/Word/108.doc
[3] OECD, "Guidelines on the protection of privacy and transborder flows of personal data, 1980," Available at http://www.oecd.org
[4] Federal Trade Commission, "Protecting consumers from spam, spyware, and fraud," A Legislative Recommendation to Congress, 2005.
[5] Secure Hash Standard, National Institute of Standards and Technology, Federal Information Processing Standard 180-2, Washington, 2002.
[6] V. Ramasubramanian and E.G. Sirer, "The design and implementation of a next generation name service for the internet," SIGCOMM 2004, 2004.
[7] D. Boneh, G.D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," Proc. Eurocrypt 2004, LNCS 3027, 2004.
[8] D. Boneh, E. Kushilevitz, and R. Ostrovsky, "Public key encryption that allows PIR queries," Cryptology ePrint Archive, 2007. http://eprint.iacr.org/2007/073
[9] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," J. ACM, vol.45, no.6, pp.965–981, 1998. Earlier version in FOCS'95.
[10] T. Nakamura, S. Inenaga, D. Ikeda, K. Baba, and H. Yasuura, Anonymous Authentication Systems Based on Private Information Retrieval, Networked Digital Technologies, 2009.
[11] B. Chor and N. Gilboa, "Computationally private information retrieval," Proc. 32nd ACM Sym. Theory of Computing, 2000.
[12] E. Kushilevitz and R. Ostrovsky, "Single-database computationally private information retrieval," Proc. 38th IEEE FOCS, pp.364–373, 1997.
[13] A. Menezes, P. van Oorschot, and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.
[14] K.H. Rosen, Elementary Number Theory and Its Applications. Addison-Wesley, 2000.
[15] C. Cachin, S. Micali, and M. Stadler, "Computationally private information retrieval with polylogarithmic communication," Proc. Eurocrypt'99, LNCS 1592, pp.402–414, 1999.
[16] M.J. Schervish, Theory of Statistics, Springer, 1995.
[17] M. Lottor, "Domain administrators operations guide," RFC 1033, Nov. 1987.
[18] P. Mockapetris, "Domain names concepts and facilities," RFC 1034, Nov. 1987.
[19] P. Mockapetris, "Domain names implementation and specifications," RFC 1035, Nov. 1987.
[20] Nomium Inc., A DNS Toolkit for Python. http://www.dnspython.org/
[21] Mee Too Crypto. http://chandlerproject.org/bin/view/Projects/MeTooCrypto
[22] M. Arlitt and C. Williamson, "An analysis of TCP reset behaviour on the Internet," ACM SIGCOMM Computer Communication Review, vol.35, no.1, pp.37–44, 2005.
[23] A. Singh, O. Nordstrom, C. Lu, and A. dos Santos, "Malicious ICMP tunneling: Defense against the vulnerability," 8th Australasian Conference on Information Security and Privacy, ACISP 2003, pp.226–235, Australia, 2003.
[24] P. Faltstrom and M. Mealling, "The E.164 to uniform resource identifiers dynamic delegation discovery system application," Request for Comments, RFC 3761, 2004.
[25] Tcpdump Online, http://www.tcpdump.org/
[26] Wireshark Online, http://www.wireshark.org/
[27] P. Ferguson and D. Senie, "Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing," RFC 2827, 2000.
[28] A. Hubert and R.van Mook, Measures for making DNS more resilient against forged answers, 2007. http://www.ietf.org/internet-drafts/draft-ietf-dnsext-forgery-resilience-00.txt
[29] M. Santcroos and O.M. Kolkman, DNS Threat Analysis. NLnet Labs document version 1.0. 2007.
[30] O. Gudmundsson, "Delegation signer (DS) resource record (RR)," RFC 3658, Dec. 2003.
[31] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "DNS security introduction and requirements," RFC 4033, March 2005.
[32] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "Resource records for the DNS security extensions," RFC 4034, March 2005.
[33] D. Eastlake and C. Kaufman, "Domain name system security extensions," RFC 2065, Jan. 1997.
[34] D. Eastlake, "Domain name system security extensions," RFC 2535, March 1999.
[35] K. Rikitake, K. Nakao, S. Shimojo, and H. Nogawa, "UDP large-payload capability detection for DNSSEC," IEICE Trans. Inf. & Syst., vol.E91-D, no.5, pp.1261–1273, May 2008.
[36] F. Zhao, Y. Hori, and K. Sakurai, "Analysis of privacy disclosure in DNS query," Proc. International Conference on Multimedia and Ubiquitous Engineering, pp.952–957, IEEE Computer Society, 2007.
[37] F. Zhao, Y. Hori, and K. Sakurai, "Two-servers PIR based DNS query scheme with privacy-preserving," Proc. International Conference on Intelligent Pervasive Computing, pp.299–302, IEEE Computer Society, 2007.
[38] S. Castillo-Perez and J. Garcia-Alfaro, "Anonymous resolution of DNS queries," Lect. Notes Comput. Sci., International Workshop on Information Security, International OTM Conference, Mexico, 2008.
[39] S. Castillo-Perez and J. Garcia-Alfaro, "Evaluation of two privacy-preserving protocols for the DNS," 6th International Conference on Information Technology: New Generations (ITNG 2009), pp.411–416, IEEE Computer Society, USA, 2009.
[40] J. Garcia-Alfaro, M. Barbeau, and E. Kranakis, "Evaluation of anonymized ONS queries," Workshop on Security of Autonomous and Spontaneous Networks, France, 2008.
[41] Y. Lu and G. Tsudik, "PPDNS: Privacy-preserving domain name system," 30th IEEE Symposium on Security and Privacy, 2009.
[42] Y. Lu and G. Tsudik, Towards Plugging Privacy Leaks in Domain Name System, Cornell University Library, 2009. http://arxiv.org/abs/0910.2472



Fangming Zhao received the M.E. degree in information engineering from Kyushu University in 2008. He is currently working as a researcher at Computer Architecture & Security System Laboratory, Corporate Research & Development Center, Toshiba Corporation. His research interests include information security and applied cryptography.

Yoshiaki Hori received B.E., M.E, and D.E. degrees on Computer Engineering from Kyushu Institute of Technology, Iizuka, Japan in 1992, 1994, and 2002 respectively. From 1994 to 2003, he was Research Associate at the Common Technical Courses, Kyushu Institute of Design. From 2003 to 2004, he was Research Associate at the Department of Art and Information Design, Kyushu University. From 2004, he was Associate Professor at the Department of Computer Science and Communication Engineering, Kyushu University. Since 2009, he has been Associate Professor of Department of Informatics, Kyushu University. His research interests include network security, network architecture, and performance evaluation of network protocols on various networks. He is a member of IEEE, ACM, and IPSJ.

Kouichi Sakurai received the B.S. degree in mathematics from the Faculty of Science, Kyushu University and the M.S. degree in applied science from the Faculty of Engineering, Kyushu University in 1986 and 1988 respectively. He had been engaged in the research and development on cryptography and information security at the Computer and Information Systems Laboratory at Mitsubishi Electric Corporation from 1988 to 1994. He received D.E. degree from the Faculty of Engineering, Kyushu University in 1993. Since 1994 he has been working for the Department of Computer Science of Kyushu University as Associate Professor, and now he is Full Professor from 2002. His current research interests are in cryptography and information security. Dr. Sakurai is a member of the Information Processing Society of Japan, the Mathematical Society of Japan, ACM and the International Association for Cryptologic Research.