Resolving Islands of Security Problem for DNSSEC

Eunjong Kim, Ashish Gupta, Batsukh Tsendjav and Dan Massey
Department of Computer Science,
Colorado State University,
Fort Collins, Colorado, USA
kimeu, ashishg, batsukh, massey@cs.colostate.edu

ABSTRACT

The DNS Security Extensions (DNSSEC) were developed to add origin authentication and integrity. DNSSEC defined a public key infrastructure over DNS tree hierarchy for the public key validation. In DNSSEC, a parent zone authenticates public keys of its child zones. The authentication hierarchy is broken when a parent does not support DNSSEC. This paper proposes an effective mechanism to overcome this partial deployment problem. Our solution uses a public bulletin board for zones to post their DNSKEY information. Resolvers use posted key information to find key authentication chains that can be used to validate the DNSKEY. Bulletin Board(BB) provides complete trust relationship information when the key authentication hierarchy is broken, and distributes the complete key information even when false zones provide the invalid keys. The bulletin board does not guarantee the correctness of DNSKEY information, but it does guarantee the completeness of the key information. Our approach helps DNS zones to deploy DNSSEC even when their parent zones do not deploy DNSSEC, and it does not require any changes to the current DNSSEC protocol and the existing software. [1]

General Terms

Bulletin Board, Trust Relationship, Sanity Check, Starting Points

Keywords

key authentication, key distribution, authentication chains, DNS Security

1. INTRODUCTION

Domain Name System (DNS) is a distributed naming system which associates many types of information with domain names, but most importantly, provides the IP address associated with the domain name. DNS uses a tree structure to enforce a delegation hierarchy on the naming system. Although DNS is a robust and scalable system and most Internet services rely on DNS to work, it is susceptible to a number of security threats such as denial of service attacks, man-in-the-middle attacks, cache poisoning, and so forth. This is because security was not a main concern when DNS was designed.

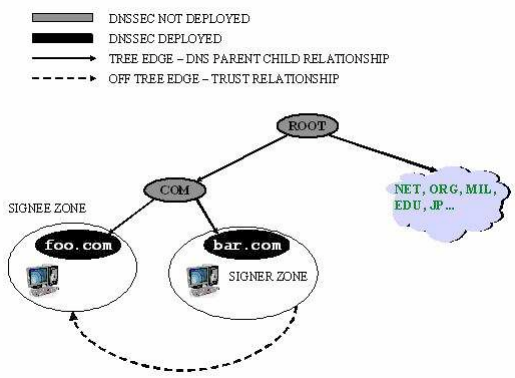

Figure 1: Partially deployed DNSSEC

To provide integrity and origin authenticity of the DNS data, DNSSEC uses a cryptographic mechanism. All DNS records are signed by zone owners and verified by resolvers. DNSSEC does not address confidentiality because DNS data is public information. DNSSEC uses delegation hierarchy for public key authentication. Therefore, all the zones from root to the queried zone must deploy DNSSEC to provide the public key authentication of the queried zone. This assumption makes deploying DNSSEC practically very difficult in the current Internet. Full deployment of DNSSEC is not feasible due to numerous political and economical reasons. If a zone deploys DNSSEC but its parent does not deploy DNSSEC, it faces the key authentication problem because the key authentication hierarchy is broken.

In Figure 1, node bar.com and foo.com follow DNSSEC. In current DNSSEC, resolvers will be unable to authenticate node foo.com because .com zone and root zone create a hole in the DNSSEC authentication chain. Therefore, we will require an off-tree edge that corresponds to a trust relationship between bar.com and foo.com. In this trust relationship, bar.com signs the public key of foo.com. If the resolver has the public key of bar.com preconfigured then it can authenticate foo.com key.

There have been some developments on deploying DNSSEC in recent years, and some domains have already deployed DNSSEC such as ".se",".nl", and ".jp" [1] [2]. This partial deployment results in island of security where subtree where all nodes except the root deploy DNSSEC.Under the current partially DNSSEC deployed environment, each security-aware resolver is required to

[1]This work was partially supported by NSF grant CNS0524172.

IWCMC'06, July 3–6, 2006, Vancouver, British Columbia, Canada.

preconfigure the public keys for the root of every island of security. These preconfigured public keys (of the trust anchors) are used to build the authentication chains (trust chains) and allow resolver to validate a signed DNS response from Bulletin Board[9]. However, preconfiguring all DNSKEYs for every root of security island is not practical since scalability and key management in each resolver is a major issue in today's fast-growing Internet. Consequently, there is a need for a systematic approach or mechanism that enables resolvers to learn DNSSEC keys.

In this paper we propose a systematic mechanism to provide the key authentication during the incremental deployment of DNSSEC. We create a public bulletin board for DNS public key (DNSKEY) information. Zones whose parents do not deploy DNSSEC post their DNSKEY information on the bulletin board. Resolvers can query DNSKEY information from the bulletin board. The Bulletin Board cannot determine the correctness of DNSKEY record because it does not know which zones are correct; however, it does guarantee the completeness of the data. Validating DNSKEY records is up to the resolver based on its local information and policy. Our system can be adapted without modifying the DNSSEC protocol and the existing software.

The rest of this paper is organized as follows. Section 2 defines our formal model and the trust relationship. In Section 3 we describe the bulletin board approach to support the DNSSEC incremental deployment in today's Internet. Section 3 presents three components of our system with fundamental mechanisms. Section 4 discusses the related work. We conclude this paper and discuss future work in Section 5.

2. FORMAL MODEL

We model the DNS as a directed connected graph $G = (V, E)$,where $V = V_n \cup V_d \cup V_s$ and $E = E_n \cup E_d \cup E_s$. The nodes in V represent the set of DNS zones and are classified into three groups. The nodes in V_n represent unsigned zones. In Figure 1 nodes root and com represent unsigned zones. Nodes in V_d represent signed zones [2] with keys signed by the parent. In Figure 1 nodes bar.com and foo.com represent signed zones. These are zones whose nameservers follow DNSSEC and each zone's public keys are signed by its parent zone. V_s is the set of security-aware zones with keys not signed by their parent. These are zones that deploy DNSSEC but their parents do not deploy DNSSEC. Hence, their public keys are not signed by their parents.

Each edge in E connects two zones and roughly corresponds to a DNS parent-child relationship. An edge is expressed with $(a \rightarrow b)$ where $a, b \in V$, and a is a parent of b and b is a child of a. Each edge in E_n connects two non security-aware zones, each edge in E_d connects two security-aware zones, each edge in E_s connects a security-aware zone and a non security-aware zone. Any zones in V should attach to at least one other zone and cannot be isolated. Each zone has zero or one parent zone, and zero or more child zones.

2.1 Trust Relationship

In order to establish the trust relationship between v and z, they should mutually agree on the existence of the trust relationship. We define a *trust relationship* from v to z as follows: a trust relationship exists from v to z if and only if v is authorized by z to sign z's public key. Moreover, z makes the authorized signers list publicly available and the authenticity of that list provided by z signing this information with its own private key. In addition to having a signers' list, a zone is required to have a signees' list which is a list of zones whose public keys the zone signed with its private key and making that list publicly available as well.

[2]Zones whose RRsets are signed and that contains properly constructed DNSKEY[9]

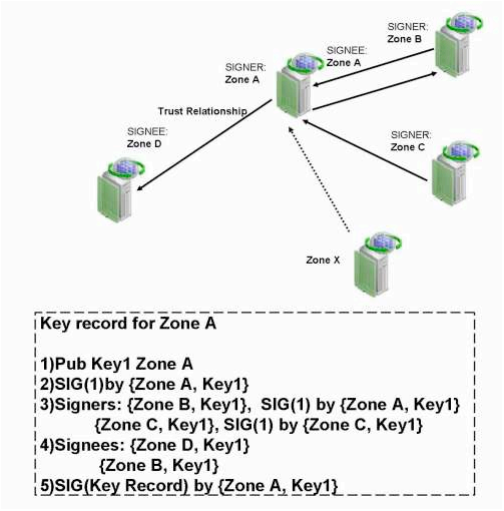

Figure 2: Trust Relationship and Key Record

In Figure 2, zone A has multiple trust relationships with zones B, C, and D. Zone A cross-signs with zone B, and is a signer of zone D, and is a signee of zone C. We also displayed zone X who claims to have a trust relationship with zone A. As zones needed represent their trust relationships with one another, we created a record called *Key record* which will provide all the details of zone's trust relationship. Figure 2 shows the key record by listing signers and signees. Key record information thus prevents unauthorized zones from claiming to have trust relationship. In Figure 2 Zone X claims it signs zone A but it is not listed in zone A's signer's list. In order to capture signer-signee relationship we introduced a new record called *bulletin board key record (BBKR)*. A BB key record consists of the following:

- zone's public key
- signature of the public key signed by zone itself
- signers list: signer zone and its key, signature of zone's public key signed by signer
- signees list: signee zone name and its key
- signature of BB key record signed by zone's public key

Definition 1 (Trust Relationship)
A trust relationship from v to z is expressed as $T(v \Rightarrow z)$ and is defined iff v signs z's public key and z provides an authorized signers' list based on mutual agreement. v provides signed signees's list which includes z.

The trust relationship has the following properties:

- **Local:** Zones are free to select their own trust relationship. Establishing trust relationship is a fully local decision. There are no global rules or restrictions on how to choose trust partners and who can or cannot be a signer. It can be done based on each zone's local policy.

- **Dynamic:** A trust relationship between a signer and a signee is flexible. New trust relationships can be established by the mutual agreement. Trust relationships can be revoked by either signer or signee. The revocation of the existing trust relationship should be done by the signee zone and be publicly announced. The revocation of the existing trust relationship can be forced by signers.

Definition 2 (Signer)
A zone v is defined as a signer when $v \in V_d$. Signer v signs another zone's public key proving that the signee zone is the entity that is bound to the public key being signed, and keeps that record in its list of signees.

Definition 3 (Signee)
A zone z is defined as a signee when $z \in V_s$ and is a root of an island of security, and the public key of z is signed by some other zones (signers), and z provides the authorized approval record about the signers.

Definition 4 (Key signer list record)
A key signer list record contains all the signers of the zone's public key and is a complete list.

Definition 5 (Key signee list record)
A key signee list record is a record that contains all the signees whose public keys the zone signed and a complete list.

Definition 6 (Conflict information)
It is the information with conflicting paths. If any pair of paths in conflict information has a common zone that signs two or more different entities which lead to different entities claiming the same zone then there is a valid conflict between pair of paths.

3. BULLETIN BOARD APPROACH

Currently, public key infrastructure in DNSSEC depends on DNS tree hierarchy. Hence, it is difficult to incrementally deploy DNSSEC in the Internet. Therefore, we are proposing a practical solution for deploying DNSSEC incrementally in today's Internet without changing the protocol itself or the existing software such as BIND that provides DNSSEC functionalities.

Broadly, our approach is to provide a public repository called the bulletin board (BB) for key information. So, if a zones parent does not deploy DNSSEC, it uses this public repository for key authentication. This public repository is used by resolvers to validate a zone's DNS public key (DNSKEY). When a resolver queries the key information to the repository, it receives the complete key information registered on the repository. In order to provide origin authentication of the response sender, all responses from the repository are signed.

When a zone deploys DNSSEC and its parent does not deploy DNSSEC, the zone locally establishes trust relationship with a zone (or multiple zones) other than its parent. The zone gives an authorization to sign its public key to other zones by listing their name in the signer's list. The signers in turn provides a guarantee that this name and key pair is valid by signing the public key. In order to provide the zone's public key information, the zone registers with and posts its key record in the BB. The key record shown in Figure 2 includes public key, signature of its public key signed by itself, its key signers list, list of signatures of its public key signed by signers, and key signees list to a publicly available BB. Those local trust relationships are announced by the zone itself via the publicly

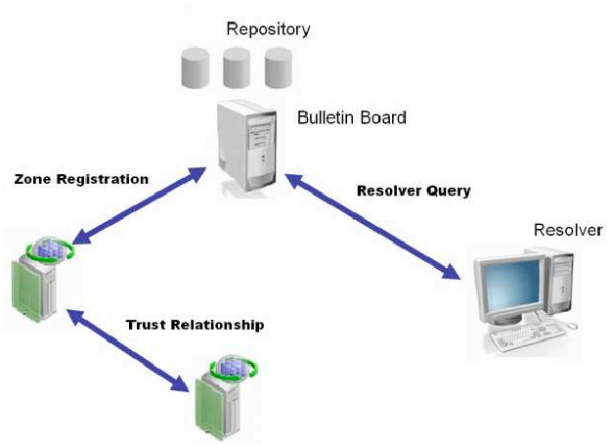

Figure 3: System Overview

available BB and found by others from there. In order to guarantee authenticity of the response, BB signs every key record in the response.

The BB public key is publicly available and assumed to be preconfigured in all resolvers [3]. Each resolver also has a set of preconfigured trust anchors' public keys.

The BB key record is used by any resolver to build the authentication chain by connecting the zone to the trust anchor. The key signee information is used to build the authentication chain in the reverse order, namely starting from the resolver's trust anchors. Each authentication chain should be completed by either encountering a node which is an element of the resolver's trust anchor set, or reaching the maximum length of the authentication chain. The maximum length is locally set by the resolver according to its policy, or by reaching a node which is self-signed.

One of the problems with our approach is that a lot of bogus information can be posted on BB either by attackers or due to misconfiguration because BB does not validate the correctness of the key signer information in the BB key record. Hence, attackers can simply flood enormous bogus BB key records to overwhelm the BB. In order to filter bogus BB key records, BB does a simple check on the received BB key records before posting them, which we refer to as *a sanity check*.

The sanity check is done by verifying if registering zone's signer is already registered or its public key is preconfigured in BB. There is an initial set of zones whose public keys are preconfigured. We call them the *starting points*.

During the repository initialization phase, BB is preconfigured with starting point public key. The starting points' keys are not intended to be trusted. They simply help BB to filter bogus BB key records. In our approach we have three basic entities: a) zone, b) bulletin board, and c) resolver.

3.1 Zones

A zone registers its BB key record to BB in order to help resolvers to authenticate its public key. Key records help create the authentication graph from the trust anchors of a resolver to the queried zone (or from the zone to the trust anchors). A zone itself is responsible for properly monitoring all BB key records associated with its name by periodically querying BB. If there exist some BB key records not sent by the zone itself (possible attack

[3]In our further discussion, security aware resolver and resolver will be used interchangeably.

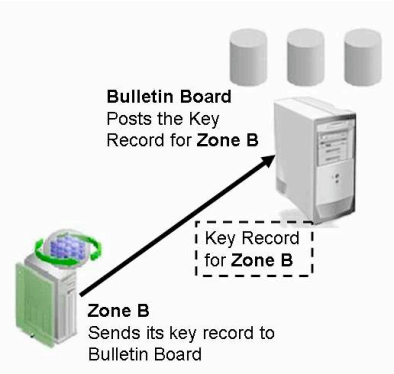

Figure 4: Zone Registration

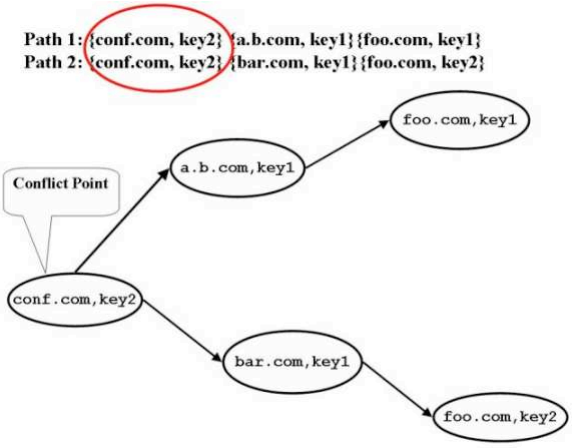

Figure 5: Conflict Report

or misconfiguration), it tries all its possible legal and economical (offline) measures to force the entity that claims the zone's identity to remove the false records posted.

Because of the importance of the records posted on BB, only zones who post the records are allowed to remove them. A zone is also responsible for updating its BB key record on the BB after the signatures associated with it expires or BB key records are changed. Key records on BB can be dynamically updated, and a zone can revoke its public key when it is compromised by an attacker. To revoke a key record the owner signs the revocation record with its private key and post it on the BB. Thus, only someone with private key can revoke the public key. A zone's new public key can be registered by using the BB key record registration procedure.

3.2 Bulletin Board

The bulletin board provides complete key information. Only when all key signers are registered in the BB, can the key information be posted. The BB *does not* decide which information is valid, or who is a correct zone.

The BB receives three kinds of messages: zone register or revocation request, conflict report (discussed below), and BB key record query for a zone.

- Zone register request: A zone requests to add or update its key record. Since BB does not know if the zone is really who it claims to be, it posts that information after the sanity check. As long as legitimate signers sign the zone's public key, this request will be processed and posted. In the Figure 4, the zone B requests BB to post its BB key record. First zone B sends a zone registration request to BB, and after BB performs the sanity check on it, the records will be posted on BB. The zone B will also be acknowledged.

- Key record query: A resolver or a zone query for key record list for a specified zone name. All the BB key records associated with that name will be signed by the BB and be sent to the resolver. The resolver will check the records against the BB signature preconfigured.

- Conflict report: If a resolver finds a conflict when trying to build an authentication chain, it reports the conflict BB key record to the BB with the proof (conflicting paths that can be verifiable by everybody). To limit bogus conflict reports, all conflict reports are signed by the resolver's zone. Once a resolver finds a conflict information, it sends the report to its zone, and the zone signs and submits the conflict report after it validates the report. If the conflict report is invalid, the zone which signs this report likely hurts its reputation maintained by the BB. Upon receiving the conflict report, the BB checks the validity of the report and posts it. Conflict paths occur when the paths share a common node which is called the conflict point. In Figure 5 we illustrate an example where two authentication paths contain a same node. The conflict node in this case is zone *conf.com* with *key1*, and it signs two zones that will lead to two different keys for a same zone name.

Although the system does filter out the bogus information flooded by naive attacks, it does not stop or prevent legitimate signers being involved in the attack or tricked into signing an attacker. If the BB has multiple BB key records for a zone, it should respond with the complete list of BB key records for the queried zone.

3.3 Resolver

Resolvers authenticate the public key of the queried zone by finding a path from one of the trust anchors to the zone (or from the zone to one of the trust anchors) and then validating this path, known as authentication chain.

The process of building the authentication chain is done by recursively querying the BB until it reaches one of the trust anchors preconfigured in the resolver. Since the BB does not determine the correctness of the registered data, it is the resolver's job to figure out which one is the correct zone records in case multiple BB key records belong to one zone. In some cases, the authentication chains do not end in one of the trust anchors in a resolver. In order to help resolvers determining the correct authentication chain, we propose a reputation mechanism to penalize misbehaving or malicious zones. However, this procedure is local to the resolver and each resolver builds its own reputation information list for zones depending on its own past experience with the zones and the local policies.

As shown in Figure 6, a resolver performing secure name resolution faces a broken authentication chain in DNS tree, it will query BB to get the key records for zone B which is a root of the island of security. After receiving the BB key records, the resolver chooses to either believe the BB key records it received from BB and end the secure name resolution, or NOT believe the BB key records if it does not find a path from the queried zone to one of its trust anchors.

Posting conflict information on the BB helps resolve to validate authentication chain. Finding conflicting paths and validating them

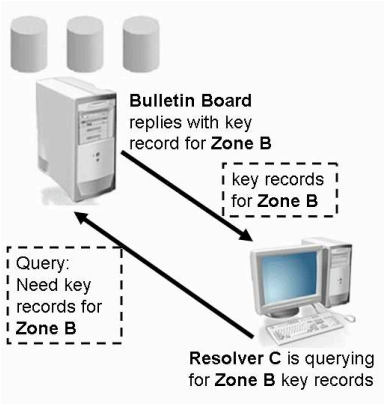

Figure 6: Resolver querying Bulletin Board

are difficult, but once this information is found, it can be easily verified by any resolver or zone. If a resolver observes conflict while authenticating the key of a zone, it sends this feedback with conflicting paths to the BB. In order to eliminate bogus feedbacks, it should be signed by the resolver's zone key.

There can be misconfigurations or attacks that can lead to formation of cycles in the authentication chain from zone to trust anchor. These kinds of cycles involving more than one node are handled by a resolver not authenticating that chain if the length of the authentication chain is more than K where K is a number chosen by the resolver depending on its local policy.

3.3.1 Resolver Search Rules

Determining which BB key records to believe is a local decision in each resolver. To make our system a better approach we defined the resolver search rules under the following assumptions.

- **Trust Anchor Set(TAS)**, $\{T_1, T_2, ..., T_m\}$, is the entire set of trust anchor T_i
  - No more than 1/3 of TAS can be invalid.
  - For any zone v_i, $\exists$ path, T_i v_i.
- **Configured Trust Anchor(CTA)**: A set of trust anchors whose keys are preconfigured in a resolver.
  - CTA $\subseteq$ TAS
  - No more than 1/3 keys in CTA can be invalid.
  - At any given time, $|\text{CTA}| > 0$.
- $a \rightsquigarrow b$ means $\exists$ a validation path from a to b, where $a \in V_s, b \in V_d$.

A resolver authenticates the zone's public key with the following steps:

1. finds the root of island of security z_2 which the zone z_1 belongs
2. send a query to bulletin board to get the key signer record of z_2. Let's call the signer of of z_2 be z_1'
3. follow the authentication chain from z_1' to root of island z_1".
4. repeat (2),(3) with $z_2 = z_1$" until it either encounters a node z_1" such that its public key is preconfigured in the resolver (resolver's TA) or reaches one of the starting points.

Now the resolver can 1) authenticate the zone's public key using trust anchor's public key by following the path found from trust anchor to the zone, or 2) choose to authenticate the authentication chain depending on its reputation list values for the nodes in the path. If there are multiple authentication chains of which none contains any of the trust anchors of the resolver, it chooses the authentication path with the highest accumulated reputation points.

We propose caching of pre-validated DNSKEYs in the resolver. The advantage of pre-validated DNSKEYs is that it saves a resolver's computation time. Resolvers do not have to re-validate cached DNSKEYs. We also propose to use breadth first search to build the authentication graph because we do not want the resolver to go and follow a long useless authentication chain out of all possible chains when using depth-first like searches. Fan-out problem occurs when a zone signs many other zones and it is encountered while building authentication chain. We use a simple heuristic function that chooses edges randomly to expand.

3.3.2 Zone Reputation Mechanism

In order to monitor and limit the zone's misbehavior, we use the zone reputation mechanism which is locally managed by a resolver. If the resolver authenticates the key without any conflict during the key record authentication procedure, the reputation values of all nodes in the authentication chain are increased. Therefore, it is to the advantage of the zone to be involved in a valid signing chain. If a zone is involved in an invalid chain, then it will hurt its reputation among resolvers. We have developed a preliminary reputation mechanism. Developing concrete rules is ongoing work and are currently following the approach from [6].

4. RELATED WORK

There has been a lot of research on key authentication problem which resulted in standardized hierarchies and systems. Two main models emerged are centralized hierarchy and web of trust models. X.509 is a widely used standard for defining digital certificate and is based on centralized hierarchy for key authentication. With X.509 standard, only a Certification Authority(CA) or someone designated by a CA is allowed to be a signer. The X.509 framework relies on the assumption that CAs are organized into a global certifying authority tree and that all users within a community of interest have keys signed by CAs a common ancestor in this global tree. [5].

One popular example of web of trust model is Pretty Good Privacy (PGP) [8]. PGP is a tool based on public key infrastructure for cryptographic privacy and authentication. PGP certificate format is different than X.509 as it can contain more than one digital signature. PGP uses self-signed signatures with third-party attestation to these signatures. Digital signatures can be given by anyone in PGP. There has been some development in the area of reputation management which we applied to our resolver. Resnick et al. propose a reputation system which collects, distributes, aggregates feedback about participant's past behavior [6]. Their properties include long-lived entities which inspire expectations of future interactions, capture and distribution of feedback of current interactions, use of feedback to guide future interactions. Due to memory constraint at resolver it will be difficult to store all the used zone's signed-signee information for future use hence, this filtering can be done based on reputation of a zone involved in signer-signee relationship.

Even though there has been a lot of research on fundamental problem of key authentication and distribution, research work on key authentication for DNSSEC is very limited. Patrick et. al. [7] suggested certificates to be locally managed within domains by entities called enterprises and can be authenticated by one or

more peer authority called keyservers. This work aims at solving problem of scalability of key distribution as number of domain increases. It also reduces the maximum number of zones involved in authentication chain to four. In case of colluding zone attack our approach can help actual zone to detect attacking zones. Scalability of our approach can be improved by using distributed BB implementation. This approach is dependent of a centralized authority (CA) as it uses digital certificates.

The DNSSEC allows verification of records obtained by alternate means. Cox et al. present one alternative storage systems for DNS records using DHash, a peer-to-peer distributed hash table built on top of Chord , a distributed lookup protocol[10]. Jones et.al. [4] proposed another architecture for distribution of public keys called Internet Key Service(IKS). IKS helps in distributing key management workloads across various IKS servers instead of DNS server for the zone. In this approaches a naive attacker can do a colluding zone attack. After PGP framework for web of trust had been proposed, PGP key servers have become quite popular way of distributing PGP keys, and one such example is Marc at MIT proposed a keyserver architecture in [3]. Even though the idea proposed here is similar to ours, BB mechanism has more advanced features such as a sanity check (low level filter), and use of public keys for authenticity, and it is the only work that is applicable to DNSSEC. The DNSSEC allows verification of records obtained by alternate means. Cox et al. present one alternative storage systems for DNS records using DHash, a peer-to-peer distributed hash table built on top of Chord [10].

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed an efficient and systematic approach that supports the key validation in partially deployed DNSSEC. Our solution also exploits a central BB to authenticate and distribute DNSKEYs. BB provides complete key information for any registered zone. Registered zones are only roots of islands of security, and resolvers use regular DNSSEC authentication mechanism while inside an island of security. Our approach benefits the registered zones immediately. These zones not only can post their key information, they can also periodically verify if there are other zones posting information claiming their identities. Each resolver determines locally the validity of the data posted on the BB, and resolvers are free to choose its reputation mechanism locally.

This paper is the first step towards incrementally deploying DNSSEC effectively using a publicly available bulletin board for public key validation and distribution mechanism. Though this approach does not guarantee the correctness of the public keys, it does guarantee the completeness of the publicly accessible key information. When resolvers have complete information about zones, it becomes easy to make decisions locally. Our approach makes it possible to delegate the responsibilities of validating keys to individual resolvers, therefore, it eliminates the need for central authority that makes the final decision. Availability of the zone's key information is solved by providing the zone registers with the BB, and implementing BB with a distributed system solves the single point of failure.

6. REFERENCES

[1] DNSSEC in .NL. http://www.nlnetlabs.nl/dnssec.

[2] DNSSEC in .SE. http://dnssec.nic-se.se.

[3] Marc Horowitz, PGP public key serve. http://www.mit.edu/people/marc/pks, 1997.

[4] John Jones, Daniel Berger, and Chinya Ravishankar. Layering a Publick-Key Distribution Service over Secure DNS. In *21st Annual Computer Security Applications Conference*, pages 409–418, 2005.

[5] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized Trust Management. In *Proceddings of the 1996 IEEE Symposium on Security and Privacy*, page 164, 1996.

[6] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation Systems. *Communications of the ACM*, 43(12):45–48, December 2000.

[7] Patrick McDaniel and Sugih Jamin. A Scalable Key Distribution Hierarchy. In *Technical Report, Electrical Engineering and Computer Science, University of Michigan*, pages CSE–TR–366–98, 1998.

[8] Philip R. Zimmermann. The official PGP user's guide, 1995.

[9] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS Security Introduction and Requirements, March 2005.

[10] Russ Cox, Athicha Muthitacharoen, and Robert T. Morri. Serving DnS using a Peer-to-Peer Lookup Service. In *International Workshop on Peer-to-Peer Systems (IPTPS '02)*, March 2002.