Vanishing Point: Resilient DNSSEC Key Repository

Bruno Garrancho, Eugénio Pinto, Luis Sousa, Nuno Loureiro

Carnegie Mellon University
Faculdade de Ciências da Universidade de Lisboa

{garrancho, goncalvespinto, lsousa, nuno} @cmu.edu

***Abstract*—The Domain Name Service is a critical part of the Internet infrastructure. If the service fails, the Internet stops working and if the service is partially or totally controlled by malicious users, the consequences can be devastating.**

This paper provides a security analysis of the Domain Name System (DNS), provides an overview of the DNS Security Extensions (DNSSEC) architecture and limitations, and highlights some of its problems: lack of resilience, multiple-root scenario, lack of isolation, legacy and Trust Anchor Management. DNSSEC Lookaside Validation (DLV) addresses most of these problems but not only it fails in providing resilience but also it devotes the root of trust of a zone into a unique trusted entity.

We propose Vanishing Point for solving the highlighted problems of DNSSEC. Vanishing Point is a resilient DNSSEC Key Repository Service that allows lookaside validation without relying solely on a PKI infrastructure. A set of notary servers is proposed, enabling a low cost and simple infrastructure, which independently collects and stores information of DNSSEC Public Keys. The keys can be requested by clients in order to make an informed trust decision about the received DNSSEC Public Key for a specific zone. This scheme uses temporal and spatial diversity, providing enhanced resilience, hierarchical independence and creating a trusted bridge for a multiple-root scenario.

***Index Terms*—DNS, DNSSEC, Security, Man in The Middle Attacks (MitM), Trust-on-first-use (Tofu), Trust Anchor**

I. Introduction

THE Domain Name Service (DNS) is the heart of the Internet infrastructure. Nevertheless, the security of the original DNS was far from being adequate to its critical role. This insecurity is a threat to one of the abutments of the Internet: if the service fails, the Internet stops working and if the service is partially or totally controlled by malicious users, the consequences could be devastating.

While it is a fact that security concerns were missing from scratch in DNS design process (e.g., lack of source authentication and data integrity protection that result in spoofing and corruption vulnerabilities), other specific characteristics of this service also impact its security. The fact that DNS architecture is hierarchical, provides confinement if the top level of the hierarchy is trustworthy, but it can only be as secure as its weakest link. Additionally, the use of caching name servers for addressing DNS scalability also results in specific vulnerabilities, namely, in polluted cached entries (the harmful effects of corrupted data may last longer and reach more victims).

After a brief overview of DNS and its Security Extensions (DNSSEC), we describe in detail some of the most important DNSSEC issues: lack of resilience, multiple-root scenario, lack of isolation, legacy and Trust Anchor Management. Although DNSSEC addresses source authentication and data integrity, these issues are mainly due to the hierarchical topology of DNS combined with the scenario of partial DNSSEC deployment.

DNSSEC Lookaside Validation (DLV) [1] addresses most of these problems but not only it fails in providing resilience but also devotes the root of trust of a zone into a unique trusted entity.

We present Vanishing Point as a resilient DNSSEC Key Repository that will allow lookaside validation without relying on a PKI-like infrastructure. A set of notary servers is used, forming a low cost and simple infrastructure, which independently collects and stores information of DNSSEC Public Keys. The keys can be requested by clients in order to make an informed trust decision about the received DNSSEC Public Key for a specific zone. This scheme uses temporal, data and spatial diversity, for enhancing resilience, achieving hierarchical independence, and providing a bridge for a multiple-root scenario.

The rest of this paper is organized as follows. Section II provides an overview of DNS and its Security Extensions. Section III elucidates the problem of managing trust anchors of DNSSEC. The Vanishing Point approach, which provides a solution for this problem, is described and analyzed in section IV. Section V addresses the possible decentralization of the DNS Hierarchy and its impact in the deployment of DNSSEC. We present related work in section VI and we compare our solution with other schemes that address similar issues in section VII.

II. DNS

A. Historical Overview

The notion of Internet names was introduced in 1971, short after the creation of the ARPANet, when the standardization of Host mnemonics was defined [2]. This concept provided an abstraction of a machine's numerical addresses by mapping them into human-legible names. A lookup table called "HOSTS.TXT" was created for this purpose and operators would install this file on their local server. The Stanford Research Institute (SRI) used to manage this file and make it globally available on an FTP server. While this scheme worked well for a number of years, it was not scalable.

The first general outline of the DNS structure was written in 1982 [3]. Within the period of two years, the concepts of delegation and authority were introduced and the initial top-level domain names were outlined [4], [5], [6], [7], [8]. At the end of 1984 the first UNIX implementation was written and, after some modification and renaming, resulted in the still dominant DNS software in use on the Internet today: Berkeley Internet Name Domain (BIND). For more information on the history of DNS please refer to the extensive article on the subject by Ross Rader [9].

In October of 2008, there was an estimation of about 11,900,000 name servers running on the Internet [10]. From these, around 36% were open resolvers (i.e., allow recursion, thus being vulnerable to cache poisoning and DoS).

Although the latest DNSSEC RFC was published in March 2008 [11], the initial feeling of straightforward deployment of DNSSEC was just an illusion. In 2008 only 0.007% of the servers had DNSSEC records, which is a surprisingly low adoption metric [10].

B. DNS Architecture

The Domain Name System is a hierarchical structure of nodes [8]. Each node has a zone, which is the administrative unit of DNS. The zone information, composed by DNS resource records, can be managed and stored by several servers.

There are three relevant components in the name resolution service:

- Authoritative name server: replies to DNS queries for a given zone.
- Cache Resolver: performs caching for indirect name resolution. If this server does not have an entry in cache that matches a query, or if the relevant information has expired, it forwards the query to one of the authoritative name servers for that domain. Then, the response is sent to the requestor, cached and used for subsequent queries.
- Stub Resolver: end-user mechanism that receives name resolution requests from applications and translates those requests into DNS queries. It is also responsible for translating the DNS reply into a meaningful application host resolution response.

The name resolution mechanism is illustrated in Figure 1. A DNS query implies that one of the root servers replies with the next authoritative server information for the levels down in the hierarchy, following a recursive process of querying the other servers, in order to resolve the full name. As the final step, the result is returned to the requestor.

C. DNS Security

The fact that security was not a primary concern in the original design of DNS resulted in this infrastructure being subject to a wide variety of attacks. The types of attacks range from Denial of Service (DoS) to Man in the Middle (MitM). However, most attacks that exploit DNS vulnerabilities are DNS spoofing and DNS cache poisoning.

1) DNS Spoofing: This attack consists in sending a fake DNS reply when a DNS request is made. It requires knowledge of the specific packet ID header that the requestor is expecting in the response. Therefore, the attacker is required to predict

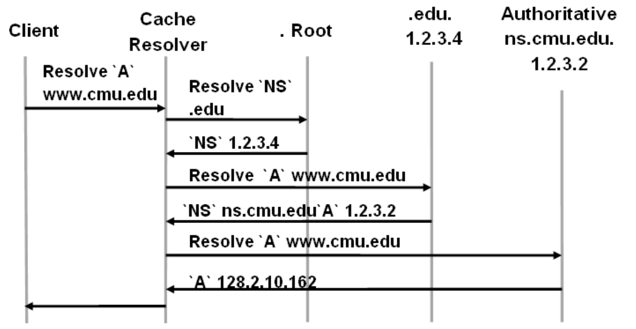

Figure 1. Name resolution

or eavesdrop the contents of DNS queries, allowing her to successfully redirect the requestor to a malicious host. The problem of DNS spoofing results from the lack authentication in the DNS name resolution mechanism. Guo et al. present spoof detection strategies based on a form of cookies for a DNS server to validate the origin of each incoming request [12].

2) DNS Cache poisoning: Cache poisoning is an enhanced version of DNS spoofing. If the client is a DNS cache resolver and an attacker can inject fake responses to queries, she can potentially poison the resolver's cache. This would cause an amplification effect, and allows extending the attack impact to numerous stub clients rather than just compromising individual clients. Dan Kaminsky exposed how fragile DNS cache implementations were to this vulnerability [13]. DNSSEC was defined to overcome these issues.

D. DNSSEC Architecture

The main focus of DNSSEC is source authentication and data integrity. In a scenario of full DNSSEC deployment, such security enhancements improve the security properties of DNS, namely making it more resistant to DNS spoofing and cache poisoning. However, full DNSSEC deployment is far from real and is not foreseeable in the near future.

DNSSEC relies on an infrastructure of asymmetric public keys. A zone signing key (ZSK) is used for signing responses and a key signing key (KSK) is used as the trust entry point for that zone, signing all the keys used. Each zone has different KSKs and ZSKs, which have expiration dates and are published in the zone as DNSKEY resource records.

The major differences between the DNSSEC infrastructure and a Public Key Infrastructure (PKI) are that in DNSSEC there is no global certification authority, no inherent management process of the certificates and no centralized directory to store the public keys.

DNSSEC guarantees integrity and authentication of data in DNS responses. The DNS Resource Records (RR) are signed with a private key, either KSK or ZSK and this signature is stored in RRSIG resource records. The verification of the digital signature is performed via public DNSKEYs.

Figure 2 illustrates the DNSSEC hierarchical Trust Chain. A crucial element in the DNSSEC hierarchical Trust Chain is the

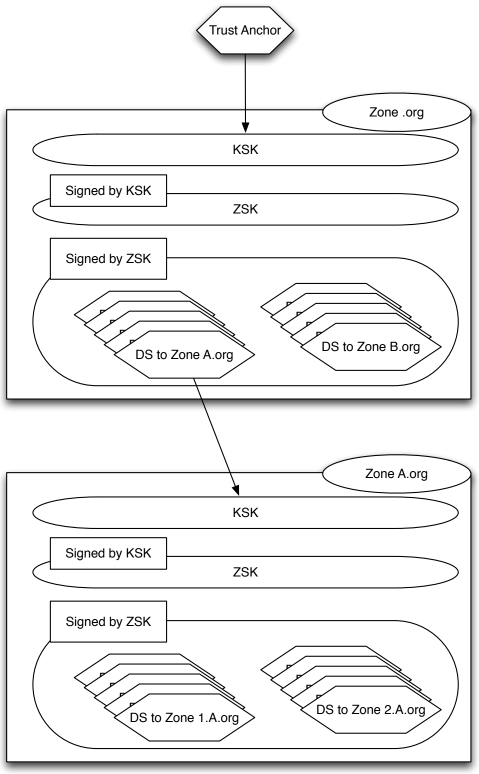

Figure 2. DNSSEC Trust Chain

Trust Anchor. The mechanism of one trusted key establishing the authenticity of other keys requires a public key that is configured as the entry point for the chain of authority (Trust Chain). A DNSSEC Trust Anchor is, precisely, that required public key. Due to the hierarchical topology of DNS, child zones sign their zones with their private key. Do note that the authenticity of that key is guaranteed by the signature of the Delegation Signer (DS).

Ideally, in a worldwide full DNSSEC deployment scenario, the Root Keys would be always the ultimate Trust Anchor.

E. Current DNSSEC Challenges

Several reasons justify the delay of worldwide full DNSSEC deployment. It is not our focus to describe in detail the ones that are not addressed by our solution, e.g., the inherent vulnerability of DNSSEC to: DoS attacks due to asymmetric crypto, the fact that DNSSEC does not protect confidentiality, additional complexity and performance degradation.

The issues of DNSSEC that we address in our research are the following:

1) Resilience: DNSSEC is not resilient in the sense that, if a DNSKEY is compromised, no mechanisms exist that enforce diversity for maintaining the correct functioning of DNSSEC.

2) Lack of Isolation: Lack of isolation is a direct consequence of DNS hierarchical topology. That yields, for example, DNSSEC not protecting the resolution of the example.com domain against an attacker controlling the .com authoritative name servers.

3) Legacy: A downgrade attack consists in intercepting a DNSSEC request from a client that is unsure whether DNSSEC is deployed and producing a fake unsigned response. This attack is similar to downgrade attack to http/https in the SSL/TLS context.

4) Trust Anchor Management: When a cache resolver needs to resolve a name for the first time, it may not hold the public key of the zone it is trying to resolve. The solution for this problem relies on the assumption that all the entities up to the root server should have the corresponding DS records, which will recursively point to the child's public KSK. The problem is that the root servers do not have DS records, neither the majority of the top level domains. DNSSEC administrators may have to retrieve the KSKs manually.

5) Multiple-Root: Due to governance impairments, there is no such globally trusted organization in which the hierarchical topology of DNS can be mapped into. This means that devoting the Root with the DNSSEC root of trust seems unfeasible for reasons that have nothing to do with technology. We discuss this topic in detail in section V.

A multiple-root scenario is a significant problem since it breaks the assumption of name resolution mechanism in which any authoritative server is reachable to any client.

III. Trust Anchor Management

As briefly described in the previous section, the trust anchor management is a present problem in DNSSEC deployment. Please refer to Figure 3 to better understand what is involved in the setup of a Trust Anchor. For the sake of simplicity, some technical details are not provided.

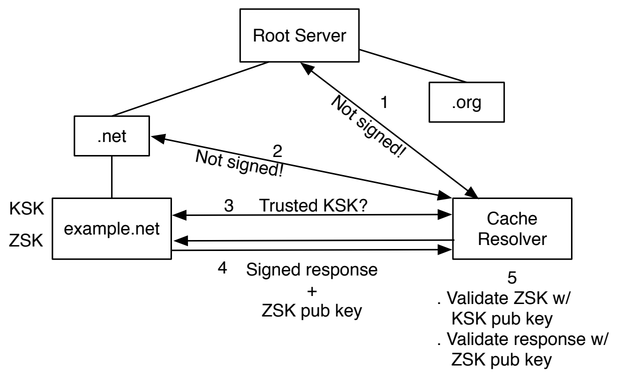
Root Server
.org
.net
1
Not signed!
2
Not signed!
KSK
ZSK
example.net
3 Trusted KSK?
Cache Resolver
4 Signed response + ZSK pub key
5
. Validate ZSK w/ KSK pub key
. Validate response w/ ZSK pub key

Figure 3. DNSSEC Trust Anchor

When a cache resolver needs to resolve a name using DNSSEC, it will end up contacting the authoritative nameserver for that domain. The data in the response is signed by a private ZSK Key, so that the cache resolver can verify the integrity and authenticity of the response. For this purpose, the cache resolver needs to hold the public ZSK of the domain. This key, which is distributed along with the response, can be trusted if signed by a private KSK of the zone. Therefore, the cache resolver also needs to hold the public KSK of the zone.

The problem is that on the first query, the cache resolver may not hold the KSK public key of the zone it is trying to resolve. The solution for this problem relies on the assumption that all entities up to the root server should have corresponding DS records, which will recursively delegate trust to the child's public KSK. However, neither the root servers nor the majority of the top level domains have DS records published in their ancestor's zones. Thus, most DNS security-aware resolvers are expected to have several Trust Anchors.

For some operations, manual monitoring and updating of Trust Anchors may be feasible, but many operations will require automated methods for updating Trust Anchors in their security-aware resolvers [14].

A related problem is the Trust Anchor Rollover. From a purely operational perspective, a reasonable key effectivity period for KSKs is 13 months, with the intent to replace them after 12 months [15]. DNS administrators should rollover KSKs periodically for security purposes, and may need to do it in case the current KSK is compromised. On the other hand, DNSSEC validators that have been offline or have missed an (emergency) rollover may need to restart the setup of the Trust Anchor.

We believe that these problems have a stringent impact in the DNSSEC deployment. DNSSEC Lookaside Validation was published to mitigate the Trust Anchor management problem [1]. The Internet Draft - DNSSEC Trust Anchor History Service - which was recently published (March 2009), proposes a trust history service to mitigate the problem of DNSSEC validators that have been offline or have missed a rollover.

IV. Vanishing Point

A. Design Goals

The deployment of secure DNS (DNSSEC) is hampered by the fact that a sub-domain (e.g., example.com) cannot protect its hosts until its parent domain (e.g., .com) publishes its own public key and signs the sub-domain public key. Unfortunately, to date, major top-level domains have shown little enthusiasm for deploying DNSSEC.

The idea of using diverse network vantage points for performing tolerant validation lookups is presented in Perspectives, in relation to SSL certificates for integrity validation [16].

The idea of using a Perspectives approach to DNSSEC, is leveraging the authoritative nameserver's capacity that for any sub-domain it can publish an unsigned key used to sign its own zone. Resolving name servers could then use multiple notaries to validate public keys prior to caching. This is a new approach to do DNSSEC Lookaside Validation that has not been studied.

B. Our Research

1) Main concepts: We present Vanishing Point as a resilient and reliable DNSSEC Key Repository Service that will allow lookaside validation without relying on a PKI-like infrastructure.

For the purpose of our solution we will consider recursive name servers (cache resolvers) as the main intended clients, however any other kind of application could benefit from it. In this sense, every proposal in this section is made in a way to facilitate the deployment of new client code on cache resolvers.

In our approach, a set of notary servers is used, forming a low cost and simple infrastructure, which independently collects and stores information of DNSSEC Public Keys. These keys can be requested by clients in order to make an informed decision about trusting a Public Key for a specific zone.

In this scheme, diversity is used in several domains in order to ensure several properties:

- Temporal diversity: the fact that the notaries store the history of DNSSEC Public Keys may be used by clients for inferring about the Public Key validity;
- Data diversity: implemented by notary quorums. The number of notaries used as the quorum is client-side configurable;
- Spatial diversity: geographical diversity ensures that notaries fail independently, either due to compromise or service failure;

2) Design: The Vanishing Point infrastructure uses two distinct components: notary servers and notary clients. Figure 4 illustrates the VP architecture.

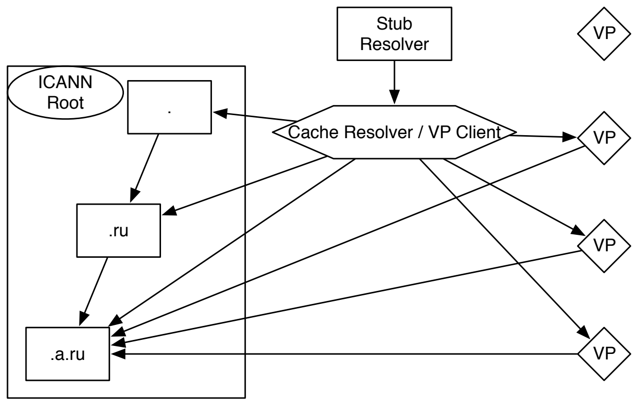

Figure 4. VP Architecture

The notary servers consist of two components: a probing module, which constantly monitors the known DNSSEC signed zones looking for new DNSSEC Public Keys, and a database storage module, which is basically a repository containing Public Key entries for each zone that the probing module is monitoring.

The notary client contacts the notary server for one of two reasons: a DNSSEC Public Key received in the response of a DNS query for a zone is not in its cache; or it does not match an existing cached public key. In both cases, the client requires the notary service for deciding on the trustworthiness of that key.

3) Notary Administration: We envision several network notary groups in a web of trust architecture. Each organization or company could run their own notary servers and establish trust relationships with other organizations. This decentralized approach, as opposed to a global and unique network of notary servers (or even the single DLV service approach), will also contribute for the scalability of the proposed solution.

Each organization should also manage its own list of notary servers, however, because of this web of trust architecture, each organization should publish the list of their notary servers in a way to facilitate other organizations to manage changes on this list. Several approaches could be used to automate this process, e.g. publishing the list in the DNS zone itself or a secure out-of-band mechanism.

4) Notary Server Key Monitoring Process: The monitoring process starts when a DNSSEC Public Key for a zone that is not in the notary's database is requested. It then ensures that the notaries periodically update their databases with fresh records. To update their records, notaries query DNS authoritative servers for the DNSKEY and RRSIG resource records, comparing the response with the stored entries for that zone. These entries contain the following fields:

- Domain name: zone identifier
- Serial Number (SN): zone serial number
- DNSKEY: zone KSK keys
- RRSIG: keys signatures
- Timestamp

If the notary server detects a new DNSKEY for a zone, it stores it in a new entry. The information stored in the notary servers does not require secrecy, since only public information is being stored.

5) Querying Notary Servers: There are two kinds of requests that a resolver can make: get current keys for a zone; get full history of keys for a zone.

When contacting an individual notary, the resolver specifies a (query-type, zone-name) pair. The notary finds the corresponding entry in its database and replies with observed key data consisting of current key(s) or the history of keys `{(ksk1, rrsig1, first seen, last seen), (ksk2, rrsig2, first seen, last seen)}`.

All communication must be authenticated, using for example a notary's private key. Using DNS as the base protocol for the transmission of this information, as proposed in the Internet-Draft "DNSSEC Trust Anchor History Service" [17], a scheme like DNSCURVE [18] could be used for secure communication between the cache resolvers and the notaries. The servers could instead make use of DNS TXT records for data exchange.

The process by which a resolver queries for and receives notary data should be as follows: the client's key-trust policy (next section) determines the number of notaries that the resolver should contact. The resolver then randomly chooses some entries from its list of notaries and queries these servers in parallel. The querying process is complete once enough notaries have replied for the resolver policy to make a trust decision, or when the resolver determines that all remaining notaries are unreachable. The resolver then validates the signature for each response using the appropriate key, discarding any invalid responses.

C. Operational Considerations

In the process of monitoring, notaries may or may not verify the authenticity of each signed key, depending on a policy. It may occur that the stored entries may contain bogus information but it is important to notice that the security of the whole solution does not depend on this design decision, since clients base the decisions on their own policies using a quorum of notaries. In fact, the verification by the notary must be seen as a mechanism for providing increased security while reducing the amount of data to be stored.

The separation of policies allows optimizing tradeoffs in both notaries and cache resolvers. The policy in the notaries allows optimizing their resources in order to ensure scalability. The policy in the cache resolvers allows optimizing the decision process, computation and bandwidth resources versus security.

1) Client Trust Policies: Once a client receives the response from the notaries, it should use a policy to validate its decision on either to accept or discard the received keys from the authoritative DNS server. Client code should provide gathering of data from several notaries, then the data should be validated against a set of policies that can be zone based and threshold filtered. For borderline cases we suggest using a quarantine mechanism for human validation. This approach can be justified given that this scenario is expected to be very uncommon. Upon policy evaluation the client decides either to trust or discard the keys it received from the authoritative DNS server.

2) Quorum: The role of a client policy is to validate the DNSSEC Public Key received from an Authoritative DNS server against a set of temporal and spatial measurements taken by the notaries. Adapted from the Perspectives quorum definitions we define threshold parameters that allow clients to take advantage of the stated properties:

Definition: For a set n of notaries, a set of keys S, and a threshold q $(0 \leq q \leq n)$ we say that a key K has quorum at time interval t iff at least q of the notaries report that K is in S at time interval t.

For the temporal realm:

Definition: For a set n of notaries, a set of keys S, we say that a key K has a quorum duration of d at time t iff for all t' such that $(t - d) \leq t' \leq t$ the key K had quorum with the threshold q at time t'.

3) Starting Points: The domain resolution starting points for recursive name servers are normally IP addresses for DNS Root name servers. Similarly, security-aware resolvers must have one or more starting points for building the authenticated chain to validate a signed DNS response. Instead of IP addresses, DNSSEC requires that each resolver trust one or more DNSKEY RRs or DS RRs as their starting point, also called the Trust Anchors.

In the Vanishing Point architecture, security-aware resolvers should also have a list of IP addresses of the notaries on which they trust. Although it is not as flexible as using DNS names, since a change of the IP addresses of notaries would require the resolvers to be reconfigured, this will increase the reliability of the solution which will not depend on the DNS service itself to resolve the notary's IP addresses.

D. Analysis

1) Authenticity vs Availability trade-off: DNS Cache Resolvers are responsible for delegated queries from numerous clients. Since they act as intermediaries, their decision to trust or not a given key has a deeper impact than a single stub client trusting a given key for a given zone on first use. In order to verify the response's authenticity for a given zone, a cache resolver must have a trusted anchor (DNSSEC Public Key in which it trusts) for that zone. If a cache resolver does not have a previous trust anchor, there is no way of assuring the authenticity of the received public key. So, upon receiving a new public key, a DNS Cache Resolver should contact a given number of notaries. The actual number depends on the policy being used, which in turn should depend on the zone being queried. The responses from the notaries vary according to the type of question, namely:

- The notaries, by default, respond with the last seen DNSSEC Public Keys.
- The Cache Resolver can, if the policy mandates, contact the notaries again for historical information - Public Keys, Signatures, first seen time, last seen time, etc.

With all the above gathered information the Cache Resolver can make an informed decision on the authenticity of the received public key. If the quorum information is not substantial, the given zone can be quarantined and subject to human verification, depending on the defined policy.

2) Improvements Over Single-Site Approach to DLV Service Management:

- All the procedures can be automated: there is no need for human-supervised publication or verification of public zone keys;
- Every Corporation can use its own set of notary servers, without depending on third-party services and without any single point of failure (without devoting the root of trust to a unique globally accepted entity);
- The solution scales along with incremental deployment of DNSSEC signed zones;

E. Scalability

Querying notaries should not present a scalability issue. In fact, caching is already the solution for the DNS scalability. However, one can argue that the potential bottleneck of VP is the monitoring process, in case there are millions of zones being monitored. The potentials bottlenecks could either be CPU or network bandwidth.

To mitigate the CPU bottleneck we suggest the usage of a proxy. Cache resolvers would query this proxy instead of querying notaries directly. This proxy calculates a hash of the requested zone name in order to make a decision about the backend notary to query. This means that the proxy server would always query the same notary for the same zone. Thus, if we have 10 notaries, the number of zones being monitored would be spread between the 10 notaries.

In regard to the bandwidth usage, considering that the automatic mechanisms used by zone administrators usually do not update the zone information more frequently than every 15 minutes, the notary servers do not require a zone update rate of more than once in every 15 minutes. If a notary service could afford a dedicated bandwidth of 1Mbps for its zone monitoring needs, it would be able to perform around 200.000 queries in 15 minutes (for an average of 600 Bytes per query). However, we believe that polling a zone every 12 hours would be an acceptable value. In that case, the number of zones in the monitoring poll could be increased to roughly 10 million. In order to handle more zones in the monitoring poll, the notary administrator could increase the time interval between verifications, increase the reserved bandwidth for this operation or distribute the load between several nodes in different networks.

F. Security Considerations

Diversity enforces dependability, which results in the Vanishing Point being tolerant to compromised notaries. The fact that notaries fail independently is crucial for the security of this scheme.

It is important to understand however, that all decisions will be based on a quorum and that notaries should be geographically dispersed, as this will prevent a MitM attack near one of the notaries from impacting the final result.

Finally, Vanishing Point increases availability in the sense that it increases the chances of a client not declining to use a new DNSSEC Public Key for security reasons.

In brief terms our solution aims to make more resilient the following attack vectors:

- Spatial Resistance: Multiple vantage points circumvent localized attackers.
- Temporal Resistance: Key history raises alarm even if all entries at the notary's are compromised at a given time. DNSSEC Key Roll-Over mechanism provides the necessary trust assurance.
- Bizantine Notary Failures Resistance: Only n correct notary responses are necessary to trust on new keys, where n is a value configured in the client.

However, our solution does not provide resistance over compromised authoritative DNS servers or clients.

G. Accordance to Trust Anchors Roll-Over Requirements

RFC 4986 defines Trust Anchors Roll-Over requirements [14]. Our research only addresses the most relevant requirements related to DNSSEC Public Keys management.

1) Initial Trust and Trust Anchor Management : Operators of security-aware resolvers must ensure that they initially obtain Trust Anchors in a trustworthy manner. RFC 4986 suggests that the correctness of the Root Zone DNSKEY RR(s) could be verified by comparing what the operator believes to be the Root Trust Anchor(s) with several 'well-known' sources such as the IANA web site, the DNS published Root Zone and the publication of the public key in well-known hard-copy forms [14].

Vanishing Point provides several trustworthy sources and is suited as a scheme for the automation of this process.

For Trust Anchors other than the root, operators must validate the DNSKEYs and/or DS RRs before using them Trust Anchors. Despite not guaranteeing the authenticity of the published information, Vanishing Point provides the means for trustworthy decisions through its spatial and temporal resistance capabilities.

2) Timeliness: Resource Records used as Trust Anchors must be able to be distributed to security-aware resolvers in a timely manner.

Security-aware resolvers need to acquire new and remove revoked DNSKEY and/or DS RRs that are being used as Trust Anchors for a zone such that no old RR is used as a Trust Anchor for long after the zone issues new or revokes existing RRs.

3) Non-Degrading Trust: The Trust Anchor Rollover solution must provide sufficient means to ensure authenticity and integrity so that the existing trust relation does not degrade by performing the rollover.

V. Governance Considerations

The Internet Corporation for Assigned Names and Numbers (ICANN) manages and coordinates the DNS root. ICANN oversees generic top level domains (gTLDs) and delegates authority over them. The root network of DNS is coordinated by ICANN, it consists of 13 Root-servers distributed among the entire world. ICANN is a private, non-profit organization based in the United States of America. The government of the United States of America can (in theory and in practice) control the global name resolution. A simple manipulation of the Root zone could cause a TLD, e.g. .ru, to be unresolved by the remaining world (outside Russia). Due to the hierarchical nature of DNS and current ICANN centralized operation of the network of Root servers, several countries and organizations are not comfortable with the current situation. ICANN's role in global communications is very limited, and yet very powerful. Alternative DNS roots are being independently operated by countries or organizations that desire to escape from the ICANN governance. In 2006, China launched a program for creating TLDs under name servers under their control [19]. Creating new gTLDs without ICANN's authorization, also known as,"Splitting the root", it involves creating a new root name server (possibly a network) and replacing the root zone file. Splitting the root scenarios could become more frequent as U.S.A. adversaries tend to deploy their own root of name servers network. This de-centralization of the DNS hierarchy could have a severe impact in deployment of DNSSEC, making solutions like Vanishing Point relevant for the future as they provide a bridge between multiple-roots according to a web-of-trust philosophy.

VI. Related Work

DNSSEC Lookaside Validation proposes an alternative and “outsourced” way for trust anchor configuration [1] . A DLV registry maintains all the trust-anchors in a dedicated domain (dlv.isc.org for example). The maintainers of secure zones, register their trust-anchors with the DLV registry. Whenever a validating resolver recognizes that a zone is signed, it will first try to validate it by assessing if it is within the island of trust configured by its local trust anchors. When the validated domain is not in a trusted island, the resolver will perform a lookup in the DLV domain and use the trust anchor from that zone if and when available. However, we believe DLV does not provide resilience and its major implementation devotes the root of trust to a single entity. Moreover, zone administrators can still face a tenuous process for registering a zone key in the DLV tree.

Some recent approaches intended to secure DNS make use of DNS server redundancy. Huang et al. proposes a rotation process to be used for periodically (proactively) setting a fraction of the DNS server replicas offline, for maintenance and cleansing [20]. By performing offline maintenance offline, the process itself is ensured to be trustworthy. Zhou et al. proposes an intrusion tolerant SDNS architecture, for tolerating Byzantine intrusions using consensus [21]. The major goal is to ensure the correctness of the responses with the voting system. However, if SDNS architecture integrates redundant proxy servers [22], the system may also enclosure enhancements in terms of availability.

Interim Trust Anchor Repository [23] is a very recent approach, intended to provide a temporary service, to be used in the legacy period until the root zone is signed. This approach uses Perspectives-like [16] diversity through the use of key repositories that act as a mechanism to disseminate "trust anchors".

VII. Discussion

In this section, we compare Vanishing Point to other schemes that address similar issues.

DLV provides a single point of measurement to infer trust over DNSSEC Public Keys [1]. Since this service only provides the current keys for a zone, DLV is useless in the case a resolver misses a key roll-over. This was actually the main concern in defining the DNSSEC Key History Service: if all the history of public keys for a zone is published in a trustworthy service, one can verify the trust chain between the key a resolver trusts and the key that is currently published in the zone [17]. Although these schemes provide a strong basis for DNSSSEC look-aside validation, they lack a fundamental security principle: the authentication of the reply messages. This issue allows an attacker to inject spoofed responses for the look-aside validation process, breaking the inherent trust assumptions.

Perspectives is a scheme that provides authenticated historic information about Public Keys along with spatial resilience [16], however not specifically designed with DNS issues in mind. Using these design principles, Vanishing Point provides DNSSEC adopters with a mechanism to strengthen the trust relations between all DNSSEC participants, both in achieving initial trust and during life-time operations. Moreover the web-of-trust philosophy inherent to Vanishing Point deployment provides a strong mechanism for the roll-out of a multi-root scenario in a sustained manner.

VIII. Conclusions

In this paper, we address some of the security issues of DNSSEC in the current scenario of partial DNSSEC deployment. All current approaches for enhancing DNSSEC security do not provide resilience. This is a major issue since only one compromised key in a DNS zone threats, e.g., all zones in the compromised zone's sub-tree. We refer to this problem as lack of isolation, since even a secure zone without any compromised key can be threatened if one of its ancestor zone is compromised. Moreover, the lookaside solutions proposed for overcoming the hierarchical topology of DNS, either rely on PKI-like entities or are impractical in terms of Trust Anchor Management.

Vanishing Point resilient Key Repository Service is an alternative lookaside approach based in a web-of-trust philosophy. We believe that this solution can solve, not only the problems of resilience, lack of isolation, legacy and Trust Anchor Management, but it is also an interesting solution for the multiple-root scenario. Since the source of the "Splitting the root" problem is the lack of trust between organizations that decide to control their separate root, our solution may be used for providing trusted bridges between those roots, following a web-of-trust philosophy (bridge-of-trust). As a result, we believe that Vanishing Point encourages the incremental deployment of DNSSEC, thus, contributing for improving the security of Internet users.

ACKNOWLEDGMENTS

We would like to thank Adrian Perrig for the motivation for this project, and Nuno Neves and João Antunes for their valuable input throughout this research.

REFERENCES

[1] Weiler, S., "DNSSEC lookaside validation (DLV)," IETF RFC 5074, Nov 2007.
[2] Karp, P., "Standardization of host mnemonics," IETF RFC 226, Sep 1971.
[3] Su, Z. and Postel, J., "The Domain Naming Convention for Internet User Applications," IETF RFC 819, Aug 1982.
[4] Mockapetris, P., "Domain names-concepts and facilities," IETF RFC 882, Nov 1987.
[5] Mockapetris, P., "Domain Names-Implementation and Specification," IETF RFC 883, Nov 1983.
[6] Postel, J. and Reynolds, J., "Domain Requirements," IETF RFC 920, Oct 1984.
[7] Mockapetris, P., "Domain names-concepts and facilities," IETF RFC 1034, Nov 1987.
[8] Mockapetris, P., "Domain names—implementation and specification," IETF RFC 1035, Nov 1987.
[9] Rader, R. W., "One History of DNS," Apr 2001, `http://www.byte.org/one-history-of-dns.pdf`.
[10] "DNS Survey," The Measurement Factory, Oct 2008, `http://dns.measurement-factory.com/surveys/200810.html`.
[11] Laurie, B., Sisson, G., Arends, R., and Blacka, D., "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence," IETF RFC 5155, Mar 2008.
[12] Guo, F., Chen, J., and Chiueh, T., "Spoof Detection for Preventing DoS Attacks against DNS Servers," *26th IEEE International Conference on Distributed Computing Systems*, 2006, pp. 37–37.
[13] Kaminsky, D., "It's The End Of The Cache As We Know It," *Black Ops 2008*, Aug 2008, `http://www.doxpara.com/DMK_BO2K8.ppt`.
[14] Eland, H., Mundy, R., Crocker, S., and Krishnaswamy, S., "Requirements Related to DNS Security (DNSSEC) Trust Anchor Rollover," IETF RFC 4986, Aug 2007.
[15] Kolkman, O. and Gieben, R., "DNSSEC operational practices," IETF RFC4641, Sep 2006.
[16] Wendlandt, D., Andersen, D. G., and Perrig, A., "Perspectives: Improving SSH-style Host Authentication with Multi-Path Probing," *Proceedings of the USENIX Annual Technical Conference*, Jun 2008.
[17] Wijngaards, W., "DNSSEC Trust Anchor History Service," Internet-Draft, Mar 2009.
[18] Bernstein, D. J., "DNS Curve," `http://dnscurve.org/`.
[19] Murdoch, S. J., "New Chinese TLDs," Mar 2006, `http://www.lightbluetouchpaper.org/2006/03/01/new-chinese-tlds/`.
[20] Huang, Y., Arsenault, D., and Sood, A., "Securing DNS Services through System Self Cleansing and Hardware Enhancements," *Proceeding First International Conference on Availability, Reliability, and Security (AReS 2006)*, Apr 2006.
[21] Zhou, W. and Chen, L., "A Secure Domain Name System based on Intrusion Tolerance," *Machine Learning and Cybernetics*, Vol. 6, Jul 2008, pp. 3535–3539.
[22] Saidane, A., Nicomette, V., and Deswarte, Y., "The Design of a Generic Intrusion-Tolerant Architecture for Web Servers," *IEEE Transactions Dependable and Secure Computing*, Vol. 6, No. 1, Jan-Mar 2009, pp. 45–48.
[23] Davies, K., "Interim Trust Anchor Repository," Feb 2009, `http://blog.icann.org/2009/02/anchors-aweigh/`.