Public Key Distribution with Secure DNS

James M. Galvin
CommerceNet, Glenwood, MD

Public Key Distribution with Secure DNS

James M. Galvin <galvin@commerce.net>
CommerceNet, PO Box 220, Glenwood, MD 21738

Abstract

Recently, many protocols in the Internet are proposing the use of public key cryptography in support of integrity and authentication security services. However, each of these protocols lacks a globally available public key distribution and management system. A secure version of the Domain Name System (DNS) is being developed which, conveniently, provides an infrastructure ideally suited for the distribution and management of public keys. We propose how this infrastructure of the secure DNS could be exploited by today's users of the Internet to distribute and manage their personal public keys.

1. Introduction

The use of public key cryptography in the Internet was first proposed by Privacy Enhanced Mail (PEM) [1,2,3,4]. PEM acknowledged the lack of a global key distribution and management system and, therefore, included an ad hoc mechanism for the distribution of public keys (embedded in X.509 certificates). Ideally, the X.500 Directory would have been become a technology of choice in the Internet, which would have provided a solution to the key (certificate) distribution and management problem. However, global deployment of the X.500 Directory has been problematic.

Recently, other protocols in the Internet are proposing the use of public key cryptography in support of integrity and authentication security services. However, similar to PEM, each of these protocols is in need of a globally available public key distribution and management system. In order to avoid the proliferation of ad hoc solutions it is essential that a single, unifying, integrated solution be proposed and deployed.

Fortunately, a recent protocol includes the specification of a global infrastructure that could be used to distribute and manage public keys for other protocols: the secure Domain Name System (DNS) [9]. As of this writing, it has been submitted for consideration as a Proposed Internet Standard. It is an enhancement of the DNS [5,6,7,8], an existing global infrastructure.

A global infrastructure for public key distribution and management must include a solution for the following objectives.

global availability - This is self-evident since a global infrastructure is not useful unless it is globally available. However, a critical aspect of global availability is scaleability; it must be possible to globally deploy a solution.

globally unique and unambiguous names - Since users prefer to be known by their name as opposed to their public key (a value that appears to be little more than a random sequence of bytes when displayed), each name must be globally unique and unambiguous. If the names are not unique and unambiguous, then the two users who share the same name would be indistinguishable by other users.

real-time access to public keys - Although other access methods are useful in some contexts, the real-time availability of public keys ensures that the global infrastructure is available to the broadest possible community.

cryptographically verifiable bindings between names and public keys - The introduction of names requires a mechanism for binding the names to public keys. The binding must be both unforgeable and verifiable by any other user. If the binding was forgeable it would be possible for an adversary to masquerade as the owner of the public key; similarly true for unverifiable bindings. In addition, it must be possible to revoke the binding.

We believe these four objectives represent a minimum set of criteria against which all global infrastructures should be evaluated. The DNS meets 3 of the 4 objectives, all but cryptographically verifiable bindings for its data. The proposed security enhancements add the functions and services necessary to meet the last objective. A global infrastructure based on the secure DNS is proposed that provides a solution to these objectives and can be used to distribute and manage the public keys of Internet users.

The Domain Name System (DNS) as deployed today is described first, followed by a description of the proposed secure DNS. The proposal is described last.

2. Domain Name System

The Domain Name System (DNS) is a distributed system for storing and retrieving resource records, a basic component that associates domain names with resources. Its most common use in the Internet today is to associate host names and IP addresses permitting the retrieval of one given knowledge of the other. Other uses include mapping old host names to new host names, identifying email gateways for hosts not directly connected to the Internet, and specifying other ancillary information necessary to the correct operation of the DNS.

The fundamentals below emphasize elements of the DNS that are relevant to the included proposal. An example is provided that can be easily contrasted with the example in the secure DNS section. This section ends with a discussion of trust issues, revocation, and how the DNS meets or does not meet the criteria stated above.

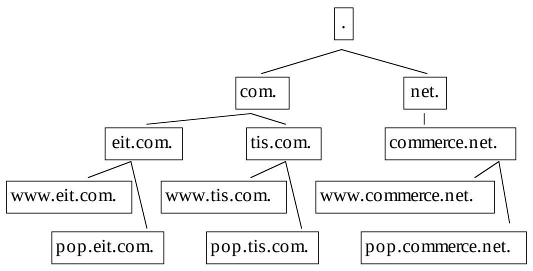
2.1 Fundamentals

Host names and IP addresses are two examples of resources of the DNS. Resources are accessed by a user or user application interacting with a resolver. The interaction includes the indication of a domain name and a resource associated with the name that is desired. The resolver interacts with one or more servers to obtain the requested resource. The interactions between a user and a resolver are a local implementation issue and will not be discussed further. The DNS specifications define the protocol used between the remaining elements.

Resource records are comprised of a domain name, a type field indicating what resource is contained within it, the data that is the resource, and other ancillary information. A domain name is comprised of an ordered sequence of labels which when displayed are usually separated by a dot (.). The type field implicitly indicates the syntax of the resource record and permits many kinds of resources to be associated with a domain name. The data is interpreted according to the type field. Some ancillary information will be considered later.

Domain names are chosen from a tree-structured name space. A domain name is either a leaf or an interior node of the tree space. Each leaf node holds a set of resource records. An interior node also holds a set of resource records, some of which will provide information about other nodes in the tree. Servers hold information about the tree structure and resource records.

The tree begins with a root designated by a single dot (.). Labels are added to the left, separated by dots, adding depth (a new level) to the tree and indicating nodes further away from the root. All labels at the same level in the same branch of the tree are required to be unique. Each node in the tree is named by concatenating the labels of the nodes in the tree along the path to the root. This is depicted below.



Some servers know that they hold all the resource records for a part of the tree; these servers are designated authoritative for that set of resource records. Although any server may respond to any query it receives for which it has the requested resource record, only authoritative servers may return authoritative resource records in response to a query.

Authoritative data is organized into zones. Zones are managed by servers. A primary server is always pre-configured with its zones. There is always at least one and may be several secondary servers that will automatically download the zone from the primary.

2.2 Example

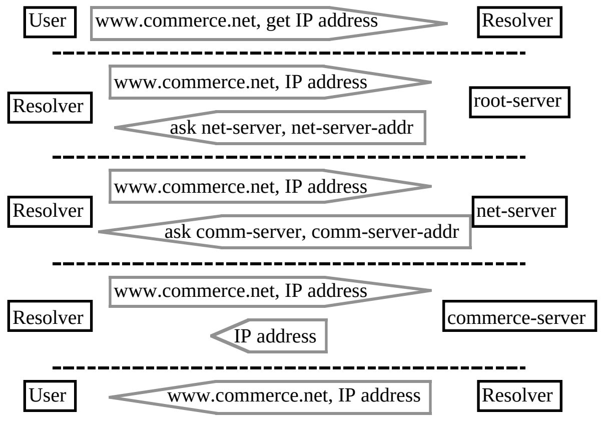
By way of example, suppose a user application needs the IP address of the host “www.commerce.net”. The user application would invoke a local resolver that accepts responsibility for obtaining the IP address.

The local resolver would have been pre-configured with the IP address of the root-server. It would begin by asking the root-server for the IP address of “www.commerce.net”. Since the root-server is authoritative for the root, it knows that it does not have an authoritative response for resources in the net-domain. It returns the hint telling the local resolver to ask net-server. It includes glue in the form of the IP address of net-server so that it can be contacted directly.

The local resolver then asks net-server for the IP address of “www.commerce.net”. Since net-server is authoritative for net, it knows that it does not have an authoritative response for resources in the commerce-domain. It returns the hint telling the local resolver to ask commerce-server. As was done by the root, it includes glue in the form of the IP address of commerce-server so that it can be contacted directly.

Finally, the local resolver asks commerce-server for the IP address of “www.commerce.net”. Since commerce-server is authoritative for the commerce-domain it knows the IP address of domain names in its domain. It returns the requested IP address.

The basic algorithm is depicted below.[1]


User
www.commerce.net, get IP address
Resolver
Resolver
www.commerce.net, IP address
root-server
ask net-server, net-server-addr
Resolver
www.commerce.net, IP address
net-server
ask comm-server, comm-server-addr
Resolver
www.commerce.net, IP address
commerce-server
IP address
User
www.commerce.net, IP address
Resolver

2.3 Trust

The DNS functions today principally because all parties to the system cooperate and agree to function according to the specifications. In particular, this means that all servers respond truthfully to all queries and do not return any misleading information. As a result the DNS can be exploited in several ways [12,13], although the example above demonstrates one of particular interest that, in fact, is required by the DNS specification.

An authoritative server may delegate branches of the part of the tree it owns, however it is not authoritative for any of the data in the delegated branch. In particular, a server will know it does not have the answer to a query but it can only provide a hint as to where the correct answer may be found. The hint it provides is the name of one or more servers that are more likely to have the authoritative data requested. Interestingly, the complete set of authoritative servers for the requested data is only available from the server that has the authoritative data. Although the set in the authoritative server and its parent server are supposed to be equal, there is no enforcement mechanism and no existing implementation checks or cares.

[1] This example is for expository purposes only and does not represent a recommended implementation strategy.

The issue is compounded by the fact that knowing the name of a server more likely to have an answer is insufficient information to proceed. A resolver needs to know the address of the server in order to contact. However, if the server is within the domain to be contact, a typical configuration, the authoritative address exists only within the server itself and thus is unattainable. As a result, servers returning hints are required to provide the glue necessary to facilitate contact by the resolver, with which the servers must have been pre-configured.

The issue is the elements of the DNS architecture are not tightly-coupled. There is no explicit or enforced relationship between a branch in the tree and its parent.

2.4 Revocation

Consistent with the DNS trust model, a revocation mechanism exists that works if all participants cooperate and function according to the specifications. In principle, a resource record binds a domain name to the data contained within it. This binding is defined to be valid until the time-to-live (TTL) field specified within the resource record expires.

When an authoritative server returns data, it specifies a TTL value in the resource record. The value is the number of seconds the recipient of the data may cache the data and use it to respond to future queries it may receive. When the TTL expires the receiver is required to destroy its local copy and re-query the authoritative server for it.

An authoritative server does not expire the data for which it is authoritative. When responding to queries it always responds with the TTL set to its initial value. To remove data from an authoritative server it must be removed from its configuration files.

2.5 Criteria Evaluation

The DNS as deployed in the Internet today meets three of the previously stated criteria.

global availability - Sites on the Internet must have a DNS server on the Internet binding their host's names to their IP addresses in order to effectively use the services available to them. The DNS is an infrastructure protocol that most, if not all, sites connected to the Internet support. As a result, the DNS is both globally available and scaleable.

globally unique and unambiguous names - By definition, the DNS defines tree structured name spaces, each with a unique root, therefore each guaranteeing that a name within it is globally unique and unambiguous. If DNS names can be mapped onto the names users prefer to associate with their public keys, the requirement for globally unique and unambiguous names would be met.

real-time access to public keys - The DNS provides real-time access to the resources it manages for the IP-connected Internet. Although the DNS is not currently used to distribute and manage public keys, it is independent of the resources that it manages. Therefore, by defining a resource record for storing public keys, it could be used for distributing and managing public keys.

However, the binding of names and objects in the DNS is based on cooperating peers. All peers are assumed to be honest and to respond to all queries with the correct answer. Thus, although the DNS is ideally suited to the needs of a public key distribution and management infrastructure, it does not meet the requirement of providing a cryptographically verifiable binding between names and objects.

3. Secure Domain Name System

Security enhancements for the DNS [9] have been drafted and submitted for consideration as a Proposed Standard in the Internet. The enhancements include the security services of data integrity and data origin authentication, noting that a digital signature mechanism could support both services. The objective of the enhancements is to cryptographically bind domain names to their resources, i.e., digitally sign the resource records managed by the DNS. Of course, backward compatibility with the existing, deployed DNS is supported.

The fundamentals below emphasize elements of the secure DNS specification that are relevant to the included proposal and assume familiarity with basic DNS terminology as defined in [5]. The example from the DNS section is enhanced to show how the addition of security affects the operation of the DNS. This section ends with a discussion of trust issues, revocation, and how the secure DNS meets or does not meet the criteria stated above.

3.1 Fundamentals

The specification defines two additional resource records that are relevant to this discussion: signature (SIG) and key (KEY).

The SIG resource record stores the signature calculated over a set of other resource records and other ancillary information, including the domain name of the signatory who created the signature and a footprint of which of potentially many signatory keys was used to create the signature.

The KEY resource record is used for storing public keys, which initially will be used by the secure DNS itself to distribute and manage the public keys it needs in support of its security services. The ancillary information stored with the public key includes an indication of the purpose for which it may be used, e.g., in support of a specific application.

A zone supported by a DNS server that creates, verifies, distributes, and manages SIG and KEY resource records is called a secure zone. All other zones are insecure. There is ancillary information in the KEY resource record that permits a server to authoritatively and securely indicate that a zone is insecure.

In a secure zone, a SIG resource record is created for each type of resource record in each domain. In addition, a SIG resource record is created for all the resource records in a domain, to be used when responding to an ANY query so that a resolver can check that all records are present, and a SIG resource record is created for all the resource records in a zone, to be used to validate zone transfers.

The authority for the public key corresponding to the private key for a secure zone is the super zone, i.e., a secure zone is not authoritative for its own key, although it may be authoritative for the public keys of domains within it.

All the security enhancements to the DNS are algorithm independent. However, to enhance interoperability among deployed systems, the specification has chosen an initial set of algorithms that must be supported by all implementations: RSA and MD5.

The proposed security enhancements will function automatically if the public key(s) of the root zone is ubiquitously known and manually configured. The specification also allows servers to decide locally which sub-trees they believe are security conscientious as opposed to believing the entire name space.

An overview of each of the basic operations of secure DNS is described below.

3.1.1 Secure DNS Signature Creation

The basic signature creation operation is as follows.

All the resource records within a domain name within a secure zone are grouped according to their type, e.g., all the IP addresses of a given host are grouped together.

Each group of resource records is canonicalized. The canonicalization involves at least expanding any compressed names in the resource record and sorting the records. The objective of the canonicalization step is to ensure a unique and unambiguous representation of the data to be signed.

The MD5 hash of each group of canonicalized records is computed.

The private key of the secure zone in which the domain name appears is used to create a digital signature for each group by signing its computed hash value. This signature is stored in a SIG record for the type over which it applies, along with the name of the signatory.

Conceptually, if the resource record being signed is a KEY record, the KEY and SIG record combination represent a certificate that cryptographically binds the domain name of the KEY resource record to its public key. Retrieval of the certificate requires retrieving both the KEY resource record and its corresponding SIG resource record

3.1.2 Secure DNS Signature Validation

The basic signature validation operation is a two step process. First the resources records of interest are verified as follows.

All the resource records of a given type for a given domain name are retrieved, along with the relevant signature record. Normal operation of the DNS would always return all records of a given type when queried for a type. Normal operation of a secure DNS server will also return the signature resource record.

The public key of the signatory indicated in the signature (SIG) record is retrieved and used to verify the signature of the resource records.

The resource record is considered valid if the signature can be verified using a valid public key.

Second, the public key of the signatory used in the second step above must itself be validated. The KEY resource record is a distinct record type and therefore has its own corresponding signature record. Consequently, validating the public key of the signatory is accomplished by invoking the signature validation process on the KEY resource record in which it is stored. This process is repeated recursively until the public key of a trusted point is used to verify a signature, as described in Sections 3.2 and 4.2.

A detailed discussion of various implementation optimizations and issues associated with validating signatures can be found in [10].

3.1.3 Secure DNS Server Operation

The behavior of security aware servers is enhanced as follows.

When responding to a query for data in a secure zone, by default only data for which the signature has been verified by the server is returned. Security aware clients may request that unverified data be included in a response, which servers must honor.

When responding to a query for data in a secure zone, both the resource record and its corresponding signature record must be returned.

The default behavior requirement that security aware servers may only respond with data for which they have verified the signature supports a very important service for non-security aware clients. It is likely that there are many (orders of magnitude) more DNS clients than there are DNS servers and that the transition to security aware servers will progress more quickly than the transition to security aware clients. If this is true, non-security aware clients will be used for quite some time after the initial deployment of security aware servers. Having security aware servers return only records for which they have verified the signature allows local[2] non-security aware clients to take advantage of the security services without any changes.

3.2 Example

The same example from the previous section is expanded here to include the additional security functionality.[3] To simplify identification of the signatory, in this example the secure zone is served by a server named within the zone itself.

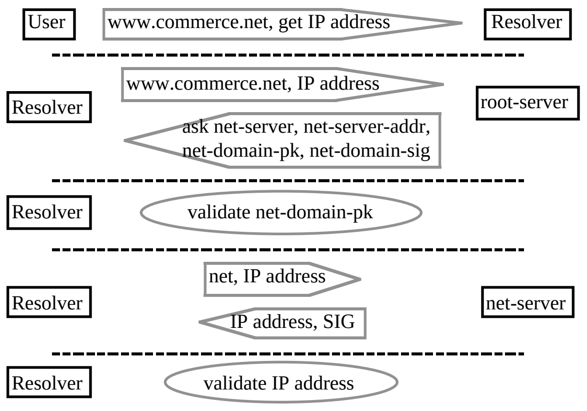
Recall, suppose a user application needs the IP address of the host "www.commerce.net". The user application would invoke a local resolver that accepts responsibility for obtaining the IP address.

The local resolver would have been pre-configured with the IP address and public key of the root-server. It would begin by asking the root-server for the IP address of "www.commerce.net". Since the root-server is authoritative for the root (.), it knows that it does not have an authoritative response for resources in the net-domain. It returns the hint telling the local resolver to ask net-server. It includes glue in the form of the IP address of net-server so that it can be contacted

[2] Use of the word local is intended to remind the reader that the client may be vulnerable over its link to the server. Presumably, a client believes its local environment is not vulnerable to adversaries and therefore it can believe responses received from local servers without verifying them.

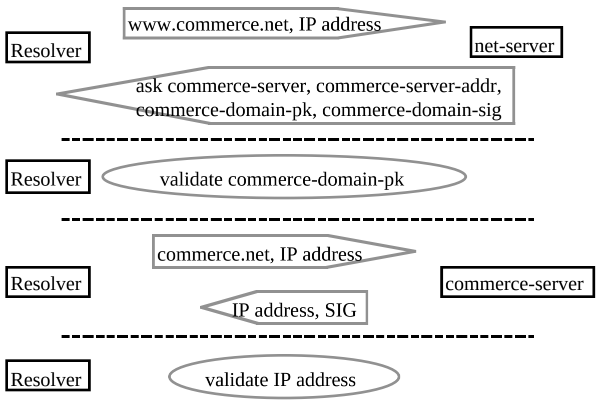
[3] As before, this example is for expository purposes only and does not represent a recommended implementation strategy.

directly. In addition, the public key for net-domain and its corresponding signature resource record is included in the response.



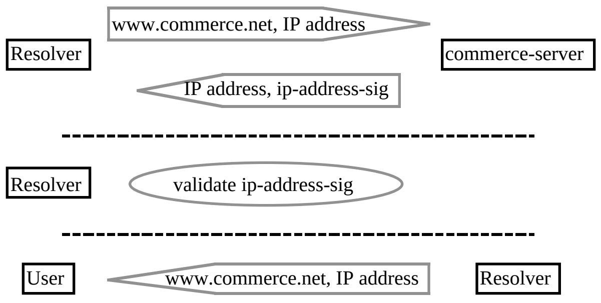
The local resolver must first validate the public key for net-domain using its pre-configured copy of the root's public key. If the public key is valid it must then proceed to validate the glue provided by the root-server. To do this it must first ask net-server for its address records. Net-server will return the complete set of address records for itself and the corresponding signature record. The local resolver must use its copy of net-domain's public key to validate the signature on the address records.

The local resolver then asks net-server for the IP address of "www.commerce.net". Since net-server is authoritative for net-domain, it knows that it does not have an authoritative response for resources in commerce-domain. It returns the hint telling the local resolver to ask commerce-server. As was done by the root, it includes glue in the form of the IP address of commerce-server so that it can be contacted directly, as well as the public key of commerce-domain and its corresponding signature record.


Resolver
www.commerce.net, IP address
net-server
ask commerce-server, commerce-server-addr, commerce-domain-pk, commerce-domain-sig
Resolver
validate commerce-domain-pk
Resolver
commerce.net, IP address
commerce-server
IP address, SIG
Resolver
validate IP address

Again, the local resolver must first validate the public key for commerce-domain using its copy of net-domain's public key. If the public key is valid it must then proceed to validate the glue provided by net-server. To do this it must first ask commerce-server for its address records. Commerce-server will return the complete set of address records for itself and the corresponding signature record. The local resolver must use its copy of commerce-domain's public key to validate the signature on the address records.

Finally, the local resolver asks commerce-server for the IP address of "www.commerce.net". Since commerce-server is authoritative for the commerce-domain it knows the IP address of domain names in its domain. It returns the requested IP address and its corresponding signature record. The local resolver must validate this one last signature by using its copy of commerce-domain's public key.


www.commerce.net, IP address
Resolver
commerce-server
IP address, ip-address-sig
Resolver
validate ip-address-sig
User
www.commerce.net, IP address
Resolver

3.3 Trust

Whereas the currently deployed DNS functions principally because all parties agree to be truthful at all times, when following authoritative data with secure DNS it is only possible to lie about the data for which a server is authoritative. Therefore, although a server can mislead clients attempting to contact it or any of its sub-zones, it is not possible to mislead a client about any other zone.

The DNS manages tree structured, hierarchical databases with single, globally unique roots. Just as the delegation of authority in the DNS passes down the tree from the root, so does the trust in the secure DNS. In the secure DNS, the trust begins at the root zone and is passed down to each sub-zone when its parent signs the sub-zone's KEY resource record. The delegation absolves the parent of all responsibility for the sub-zone except to indicate its existence and where to find it. The digital signature provides undeniable proof of the parent's intent to delegate, although it says nothing about the sub-zone's acceptance of the delegation.

A feature of having a secure zone be authoritative for the KEY records of its sub-zones is that the secure DNS architecture is tightly-coupled. From the root in the secure DNS, it is always possible to move securely to any other zone. Of course, a zone could respond with false information about itself and its sub-zones, but doing so only prevents legitimate access to itself. However, the trust relationship is not the same when moving up the DNS tree.

According to the secure DNS specification, a secure zone is required to have a KEY record with the public key of its superzone in its zone signed by itself. The purpose of this record is to make it possible to securely move up the DNS tree. This permits sites to pre-configure the public keys of arbitrary secure zones instead of the root, presumably on servers "close" to them, and still be able to move securely to any other zone.

Unfortunately, authority in the DNS, in particular the delegation of zones, passes down the tree not up the tree. Although it may be possible to trust the "close" servers of the pre-configured zones to be truthful about the public key of their parent zones, prudence suggests that the transitivity of that trust be limited. Whereas moving down the tree limits the damage a malicious server can inflict to data for which it is authoritative, a server that lies about its parent could mislead a client about a far greater portion of the tree. In the extreme, it could masquerade as a root to clients moving up the tree and subvert the entire name space.

Another feature of being able to pre-configure the public keys of arbitrary secure zones is to permit full use of the security features between disjoint fragments of the tree. Until secure DNS supplants the currently deployed DNS, there will be portions of the tree that implement security but whose parents do not. These portions of the tree could advertise their public keys, via several different mechanisms to enhance validation, thus permitting other secure portions of the tree to pre-configure the public keys and be guaranteed of obtaining correct information.

3.4 Revocation

The secure DNS specification does not include an explicit mechanism for revoking the bindings created by digitally signing resource records. Instead the DNS revocation mechanism was enhanced by having the digital signature of a set of resource records cover the initial value of the TTL field. Only the initial value is protected because protecting the values decremented by servers who have cached the resource record for a while would require those servers to keep a private key on-line at all times to be able to sign resource records in real-time.

The normal operation of the DNS requires clients to re-query for data for which the TTL field has expired. Protecting the initial TTL value guarantees that a client will re-query for data at least that frequently. This is functionally equivalent to an explicit revocation mechanism.

When using an explicit revocation mechanism, validation of a binding requires that a list of revoked bindings be retrieved and inspected to see if the binding of interest is listed on it. The revocation list is updated on a regular basis, at a minimum frequency usually indicated in the list and commensurate with the perceived significance of the binding. To leverage the DNS TTL field as a revocation mechanism, its value should be set to the minimum frequency an explicit revocation list would be updated if it were used.

During normal operation, a server would check to make sure the TTL on a resource record had not expired prior to using it in a response. Checking the TTL is analogous to checking for membership on an explicit revocation list.

In order for a secure zone to revoke a binding it either removes the invalid resource records from its configuration or it replaces them with valid records. All other servers will acquire the new bindings no later than the value of the TTL from time the valid records are inserted.

Since the security of the entire system is dependent on the use and management of valid public keys, the key bindings are more significant than other bindings. As a result, the TTL for KEY resource records should probably be substantially less than that used for other records, e.g., if 30 days is used for most records, 7 days would be a good choice for KEY records.

It is important to remember that the expiration of the signature is distinct from the TTL. The secure DNS specification includes a maximum lifetime for each signed resource record. During normal operation a signature could be valid for 1 or more years in contrast to the 1 or more months of the TTL.

3.5 Criteria Evaluation

Since secure DNS enhances or adds to the existing DNS, as opposed to changing or removing from the existing DNS, it inherits the meeting of the same criteria as the existing DNS. In particular, global availability, globally unique and unambiguous names, and real-time access to public keys.

One potential issue with respect to global availability is whether or not secure DNS will scale for global use. The DNS database (actually each of several) is predicated on the existence of a single global root domain. On the one hand, PEM may epitomize the likelihood that the Internet community will embrace an infrastructure structured in this way, i.e., it will fail.

On the other hand, there are at least two, more positive outcomes. The first is that the DNS has been functionally exceptionally well with a single global root. The space is managed by the Internet Assigned Numbers Authority and, setting aside the various trademark, copyright, and other related legal issues, there is acceptance and support for the process. It is entirely possible that secure DNS may succeed with a single global root in spite of the failure of PEM.

The second possibility is that secure DNS could be deployed with multiple “roots”. Instead of a single root domain, setup each top-level domain name to be the root of its hierarchy. This would require the maintenance of multiple root public keys, which would require the establishment of a distribution mechanism. However, given that the DNS handles the problem of multiple servers for the root domain, it is likely that a similar kind of solution will suffice for the management of multiple root keys.

In any case, this issue will be resolved as soon as we begin to see global usage of secure DNS. As of this writing, at least one reference implementation is being developed and it is currently in beta test. In fact, there are two significant implementation issues that are relevant to the scaleability of the secure DNS.

First, the addition of public keys and their signatures to the DNS will stress the robustness and reliability of current DNS implementations. For example, the most popular DNS implementation maintains its entire database cache in memory. For 1024-bit RSA keys, 500 user keys will require at least 1-megabits of memory just to store the public keys and their signatures in the cache, which does not include the memory needed to store the 500 domain names and header information of each of the key and signature records. While it is true that hardware and memory are getting cheaper all the time, will it be cheap enough that the “com” domain, for example, can be managed by one server?

Second, as is always the case with public key cryptography, the protection of the private key is the cornerstone of the security provided by the entire system. The secure DNS specification recommends that the signing of resource records (the creation of the SIG resource records) be completed off-line, and that the database file with the original resource records and their signatures be transferred manually to the on-line primary server. Although this process may appear cumbersome, in principle, it should not be a frequent occurrence. Alternatively, other technologies could be explored for protecting the private key of the zone, e.g., a trusted system, which provides guaranteed access control, and PCMCIA cards, which keep the private key in a physically secure location.

Finally, secure DNS meets the last criteria that was not met by the existing DNS: cryptographically verifiable bindings between names and public keys. By definition, secure DNS cryptographically binds domain names to the remaining data in the resource record. Since there is a resource record expressly for the purpose of managing public keys, the secure DNS meets this requirement.

4. User Public Keys and Secure DNS

The secure DNS uses public key cryptography in support of its security services. It provides for itself a global key distribution and management system, which is exactly what users and user applications need to support their security services.

If each user was assigned a domain name, then each users' public key could be stored as a KEY resource record in their domain. From the point of view of the secure DNS, retrieving and verifying these KEY resource records is no different than retrieving and verifying the KEY resource records it uses during its normal operation. User applications could be modified to query the secure DNS with the user’s domain name whenever the user's public key is needed.

4.1 User Domain Names

In the Internet, we already have a well-understood, globally deployed, tree-structured name infrastructure for users: RFC822 [11] email addresses. Many users are already commonly known by their email addresses. Internet-connected users include their email address on their business cards. Many companies are known by their domain name and some provide a variety of well-known email addresses through which they may be contacted.

To use email addresses there must exist rules for mapping them to DNS names. For a large fraction of Internet users this will be a straightforward process. RFC822 email address all have the syntax “local-part@domain-part". By definition, domain-part is a domain name so it is easily appended to the result of mapping local-part to a legal sequence of one or more domain name labels.

In the simplest case, the local-part conforms to the syntax of a domain name label, i.e., it contains only the letters A-Z and a-z, the digits 0-9, and the hyphen, e.g., galvin@commerce.net. The at-sign (@) character could be replaced with a dot (.) resulting in an email address that becomes the user's domain name.

It is incrementally more complicated to permit dots and other punctuation characters to appear in the local-part. Other punctuation marks could be replaced with dots, collapsing multiple adjacent dots to a single dot. This will require the creation of additional branches in the DNS tree, but that is a one time administrative exercise for each user, easily completed as part of creating a new user account.

It is incrementally more complicated again to provide mapping rules for other printable yet illegal characters, e.g., parenthesis and quotation marks. An approach that works is to replace illegal characters with strings of legal characters, similar to the approach taken in [14].

This simple set of rules will serve to provide automatic processing for a large fraction of the email user community. For more complicated local-parts, for example X.400 addresses, more complex mapping rules would have to be developed.

4.2 Validation Issues

Believing cryptographically verifiable bindings requires that a trusted path exist from the binding of interest to a trusted point. By default, the most trusted point in the secure DNS is the root. Conceptually, the root signs the public key for each of its delegated zones, and makes each key and its respective signature resource record available via the DNS. Similarly, delegations at other points in the tree would have their key records signed by their super zone at the point of delegation.

Validating a signed resource record constructs a sequence of KEY resource records, the first of which contains the public key of a trusted point. This public key is known *a priori* to be valid and trusted. It is used to verify the signature in the SIG record of the next KEY record in the sequence. Similarly, this next key is used to verify the signature in the SIG record of the next KEY record in the sequence, until the sequence terminates with the KEY record containing the public key needed to verify the signature of the signed resource record. Consider the example detailed in Section 3.2.

Since the secure DNS is a homogeneous system, all of the public keys created exist for a single reason and are implicitly signed according to a uniform policy. However, when keys for other purposes are added to the DNS, there is no implicit policy. In the absence of an explicit policy, it is not possible for an application to automatically evaluate whether a retrieved key is suitable for its intended use, irrespective of the existence of a trusted path. The only evidence a trusted path provides is that the name associated with public key is correct.

As a result, the DNS provides a mechanism suitable for assisting in the management of globally deployed public keys, but it is insufficient to support in and of itself the knowledge of whether a public key is suitable for a particular use. Additional ancillary information is necessary, e.g., a policy that defines the valid uses of the public key.

However, where human interaction is possible and likely, e.g., with secure email applications, a user could evaluate in real-time whether or not a public key was appropriate for use. In such an application, the secure DNS is a necessary and ideal enabling technology to the global deployment of the application.

5. Conclusions

The continued standardization of the use of public key technology in the Internet demands a globally available public key distribution and management system. The Domain Name System (DNS) is an infrastructure protocol which provides an ideal base on which to build such a system. The secure DNS specification being designed and implemented includes a public key distribution and management system that could be used to manage users' public keys if users were assigned domain names.

With a domain name users could store their own public keys as resource records where they would be quickly and easily accessible by others. This is straightforward to do for the vast majority of Internet users by taking their email addresses and changing the at-sign (@) to a dot (.). Additional suggestions were proposed for incrementally more complicated email addresses.

The use of the secure DNS as a public key distribution and management system for users will require changes to application programs. However, this transition can not proceed until there exists a reference implementation. As of this writing, one is being developed and is currently in beta test. However, as indicated above, the scaleability of existing implementations is uncertain.

Finally, using the DNS to validate public keys for users begs the question of under which policy was the public keys are signed. This is an issue being addressed in the context of X.509 certificates by many different organizations, but no attention to date has been given to resolving the issue in the secure DNS.

6. References

[1] John Linn. Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures. RFC1421, February 1993. Obsoletes RFC1113.

[2] Steve Kent. Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management. RFC1422, BBN Communications February 1993. Obsoletes RFC1114.

[3] David M. Balenson. Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers. RFC1423, Trusted Information Systems, February 1993. Obsoletes RFC1115.

[4] Burton S. Kaliski. Privacy Enhancement for Internet Electronic Mail: Part IV: Key Certification and Related Services. RFC1424, RSA Laboratories, February 1993.

[5] Paul Mockapetris. Domain Names - Concepts and Facilities. RFC1034, ISI, November 1987. Obsoletes RFC973.

[6] Paul Mockapetris. Domain Names - Implementation and Specification. RFC1035, ISI, November 1987. Obsoletes RFC973.

[7] Paul Mockapetris. DNS Encoding of Network Names and Other Types. RFC1101, ISI, April 1989. Updates RFCs 1034, 1035.

[8] Bill Manning and Richard Colella. DNS NSAP Resource Records. RFC1706, ISI and NIST, October 1994. Obsoletes RFC1637.

[9] Donald E. Eastlake and Charles W. Kaufman. Domain Name System Security Extensions. Work in Progress.

[10] James M. Galvin and Sandra L. Murphy. Using Public Key Technology - Issues of Binding and Protection. INET'95, Internet Society, June 27-30, 1995.

[11] David H. Crocker. Standard for the Format of ARPA Internet Text Messages. RFC822, University of Delaware, August 1982.

[12] Steven M. Bellovin. Using the Domain Name System for System Break-ins. The Fifth USENIX UNIX Security Symposium, Salt Lake City, June 5-7, 1995.

[13] Paul Vixie. DNS and BIND Security Issues. The Fifth USENIX UNIX Security Symposium, Salt Lake City, June 5-7, 1995.

[14] S. Hardcastle-Kille. Mapping between X.400(1988) / ISO 10021 and RFC 822. RFC1327, ISODE Consortium, May 1992. Obsoletes RFC987, RFC1026, RFC1138, RFC1148. Updates RFC822.