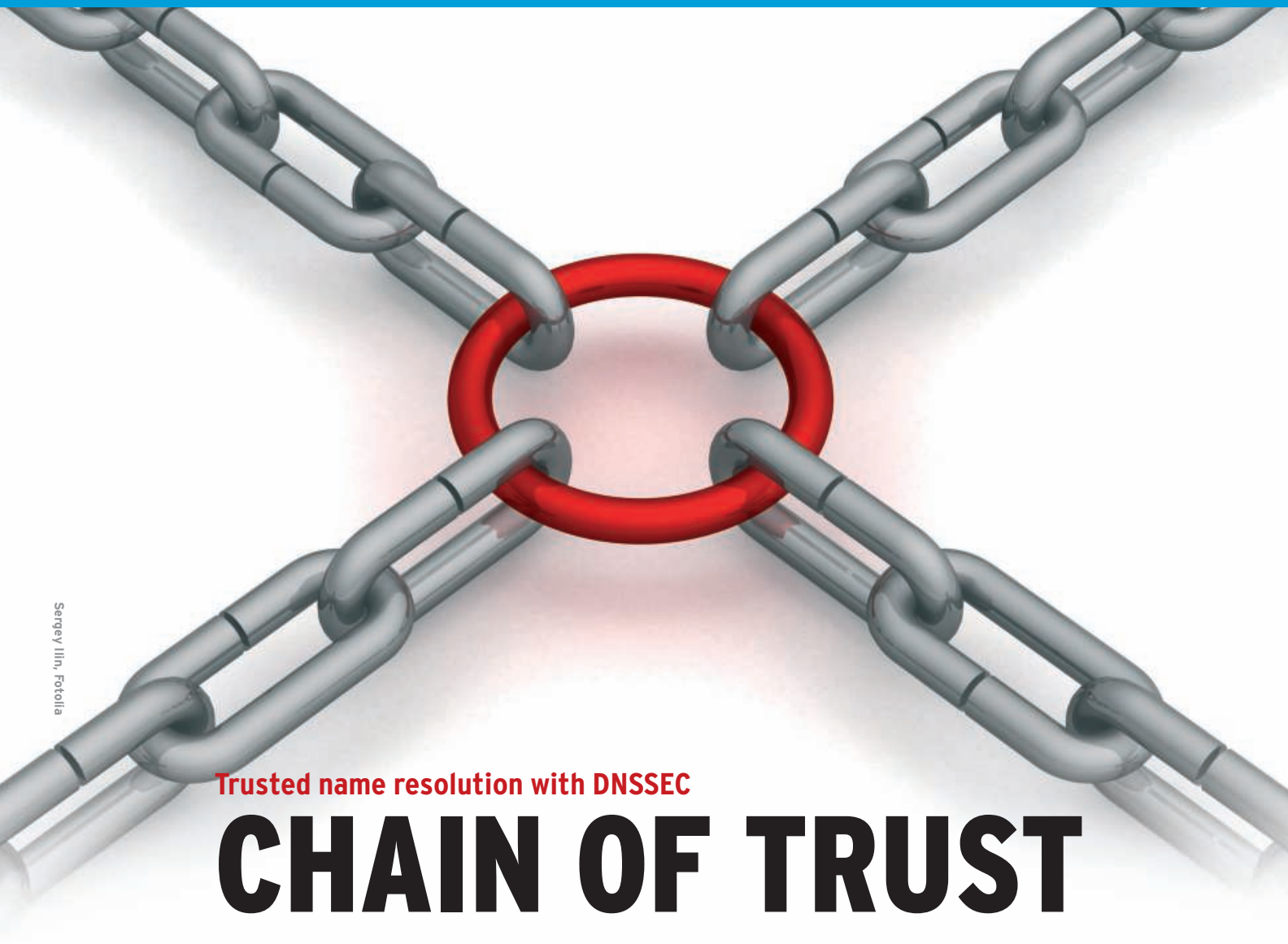

Sergey Ilin, Fotolia

Trusted name resolution with DNSSEC

CHAIN OF TRUST

Some Internet exploits target name resolution servers. DNSSEC uses cryptography to protect the name resolution service. **BY ERIC AMBERG**

System administrators and security consultants have devised elaborate strategies for protecting computer networks, but one very basic part of the Internet infrastructure is still surprisingly vulnerable: the name resolution system. Intruders have developed sophisticated techniques for spoofing DNS responses. Of course, the white hats have fought back with their own defensive maneuvers, but experts agree that a fundamentally different approach is necessary. The DNS Security Extensions (DNSSEC) system [1] offers a comprehensive solution for authentication and data integrity for DNS.

THE AUTHOR

Eric Amberg has worked for many years as a system engineer for IT networks in large companies. His special subjects are Linux and network security. In addition, Eric has published various articles and the book *Linux Servers with Debian GNU/Linux*.

DNSSEC adds cryptographic signatures to the legacy name resolution service. But a signature can't solve the problem alone (because an attacker can create a signature, too). DNSSEC also needs a method for authenticating the public key used in the asymmetric encryption, which means the system must provide its own form of Public Key Infrastructure (PKI).

Chain Reaction

Because the DNS system typically resolves names through a hierarchical chain of interacting name servers, DNSSEC can only guarantee authenticity if it operates at all levels of the chain. A complete solution therefore requires the adoption of DNSSEC on a massive scale. So far, the Swedish *.se* domain is the only top-level domain signed with DNSSEC, but many organizations have started implementing and experimenting

Teamwork

To help DNSSEC succeed, two groups must make a contribution: Users can only benefit from the system if network managers provide servers that use DNSSEC responses to validate their users. Name server managers must sign their zones and integrate them with the chain of trust in the superordinate zones [2]. The free ISC BIND name server, which many regard as being a DNS reference implementation, provides solutions for both these objectives [3].

DNSSEC name server extends its zone file. Besides administrative information in the SOA record, it mainly contains RRs that support mapping of DNS names to IP addresses or vice versa. DNSSEC uses signatures to protect the RRs. To do so, the DNSSEC introduces another series of RRs, as listed in Table 1.

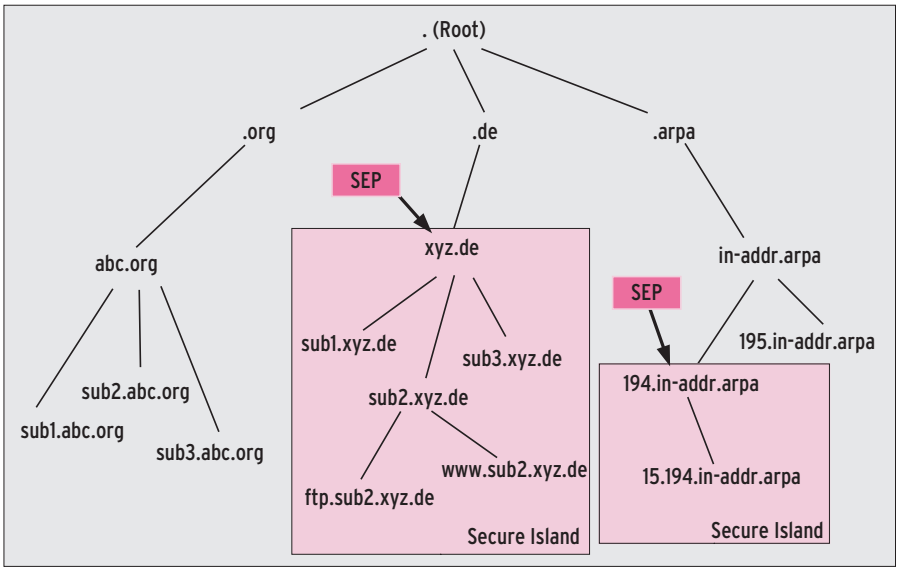

Figure 1: DNSSEC adds SEPs to the DNS domain hierarchy. Later on, these secure islands will grow together to form a large DNS continent.

with DNSSEC at lower levels. In this article, I will look at trusted name resolution with DNSSEC.

Public Keys for DNS

The first thing you will need is a resolver that supports DNSSEC. Because most stub resolvers can't do this – and the one in libc is no exception – administrators on enterprise networks will need to install a name server and enable its DNSSEC functionality.

Now, thanks to DNSSEC, when clients on the network ask the server for IP addresses, the name server is guaranteed to return reliable results. Of course, the hop between the client and the first server is not safeguarded and theoretically could be manipulated. If you are responsible for security on your network, you will need to decide on an individual basis whether to take this lapse seriously.

The DNSSEC resolver now checks to see whether the query is for a DNSSEC-secured zone. If the requested target is on a secure island, this is always true. The top nodes in these structures are referred to as Secure Entry Points (SEPs, see Figure 1). Admins must make these entries the top priority on the DNSSEC resolver. Thus, the list of SEPs is the functional equivalent of providing CA certificates to a web browser.

Lonely Islands

DNSSEC uses the same access mechanisms as legacy DNS. Because the resolver only requests Resource Records (RRs) from a server, the system is downwardly compatible. Additional security is provided by a DNSSEC-enabled resolver validating the signatures in the RRs. If a response is not correctly signed, it is discarded.

Because the user is never tempted to use a potentially compromised response, this is a very secure approach. However, users must get used to the server responding with *NXDOMAIN*, which means "this domain does not exist."

In contrast to this, PKI will pop up a window with web certificates in the same situation. The user can decide how to react to the invalid certificate; unfortunately, many users just ignore the warning.

If the response does not come from a secure island, the resolver will resort to legacy methods to resolve it and then return the response to the requesting client. Security admins should be aware that, if they use DNSSEC, the user will not be able to tell whether or not a response is authenticated by DNSSEC.

In the long term, the DNSSEC lobby seeks to have just a single SEP that points to the DNS root zone. A chain of trust links the signing key with all the zones below it in the hierarchy. This lets DNSSEC resolvers validate signatures. On the Internet today, this is not the case, in that it is still just interspersed with independent secure islands. Until the islands grow together, resolver administrators still need to manage multiple trusted keys as SEPs.

Chains of Trust

DNSSEC uses asymmetric key pairs – that is, pairs of private keys and public

Listing 1: DNS Config for SEPs

```
01 trusted-keys {
02  "example.com." 257  3  5
03   "AwEAAcDKu5Kqbk92caGeQ2GjQDucJ2t6jfUb
04    gdye+zyw6qS9PorViM5ViTifFt1JYgB5RnGf
05    wFwqEDm2eeopakOYnJdnVAgDJFd/4sEp7dJW
06    A4zPEvy8LYXCAqkbL5FqZcv9fbYHF2rKYlZJ
07    y5MbmEOk/X4nrxcjwSIcbpIe4/mhjWmRl+jA
08    AVlyODwko2edeilKuW5y/LpPvdZ3qXsw6mTU
09    pa39NcGbzDbHVyFZrQhnxjCD2cy6rWe5ZYck
10    c9VyQQafFLXx5h56AifOmi1i7f7uZjm6wAIc
11    iv+CkVUfKbcdqpoBThBWH67VqD8kljLRsEGt
12    wRWZbGfjhuGkm56MHZCfYTk=";
13
14 "tux.local." 257  3  5
15   "AwEAAa+z+JB9qd6Q9Kg7isg/DqJdqX9Kqxpu
16    One4zGlUWNJXAT5ivVva5N4l1YOPfq2M+dJH
17    Mxg9jmFZmrTLS8HYvuYzTVuBMh1u3hvS6UBr
18    SzEJdqWDpO/AJbWDUP+SIfryeWOZV7weHDX7
19    Xjqrrh2+8+Dc/k8LFxoocBeio9gljYMLdIvM
20    ddOUOhFxx6o4WvVNhuWF+i1HDoqGDOOWgRCk
21    9KOOfZpx8h/dwwyqL4/9ZkOMLF6KQaxgO+tQ
22    khhI6sq+7BYmnNBauJQlwLY8qrlA/gaajahU
23    PaHbJ1vzg+G5mLFI1vEt5FTGVXWJpOGWD6yK
24    uLdrYlLOoOapQ8FG9AqMrvk=";
25 };
```

Listing 2: Key Entry for a Zone File

```
01 cat Kexample.net.+005+18553.key
02   example.net. IN DNSKEY 256 3 5 (
03       ZUPI4+OM1VO+SQmFzHQtZMuzLH3UxWEOGmG5Gfj...
04       ijandHGG8lD3IO1azWN6DiVFEVzgrOotAdDonfY...
05       =oElkw== )
```

keys. The two-element system was devised by IETF (Internet Engineering Task Force) architects. A Zone Signing Key (ZSK) protects the individual RRs in a zone file and, in turn, is protected by the Key Signing Key (KSK) (see Figure 2).

Inside a zone, it is sufficient to know the public KSK to validate the ZSK and then the RRs. Between a parent and a child zone, DNSSEC uses a delegation signer RR (DS-RR). At the top of the chain of trust is a KSK, which specifies the SEP, or Trusted Anchor, and designates the zone hierarchy below it as a secure island. Adding these SEPs to the DNSSEC server configuration is the responsibility of an organization's admin.

To do so, the admin enters the KSKs for the secure islands to be supported to the *trusted-keys* section of the *named.conf* file (see Listing 1). The hierarchy should use the highest possible KSKs available and make sure that the keys were transferred in a trustworthy manner. The zone names are followed by three fields. The Flags field defines the key type; 256 stands for ZSK and 257 for KSK. The second value is the Protocol field, which must contain a *3*, in line with RFC 4034. The third value specifies the algorithm used, with *5* standing for RSA/SHA-1.

Client or Server?

The standard scenario uses a DNS server as the resolver on the local network that queries a forwarder on the provider's network. The DNS server validates the responses it receives from the provider-side DNS server. To be able to do so, the administrator first must enable DNSSEC. As of version 9.4.2, ISC recommends a BIND server. Enabling DNSSEC when building the program is imperative. Most distributions offer this option as part of their *bind9* packages. The *dnssec-tools* are useful for testing and debugging but are not necessary to run the system [4].

The *dnssec-enable yes;* option in the *named.conf* configuration file generically enables DNSSEC functionality. If at least one trusted key is defined as a SEP, you just need to reload *named*; the resolver in this example would validate the *example.com* and *tux.local* zones, as well as any zones below them in the chain of trust (e.g., *branch1.example.com*).

Signing a Zone

Name server operators first must generate and set up key pairs for their domains and zones, starting with ZSKs and KSKs on the primary domain server to sign the individual zone records. Authoritative zone servers must be enabled by setting *dnssec-enable yes;* for DNSSEC. The following command issued on the primary name server creates a key pair for the *example.com* zone:

```
dnssec-keygen -a RSASHA1 -b ➥
2048 -n ZONE example.com
```

The *-a* option specifies the algorithms RSA and SHA1. Although the developers typically recommend RSA with SHA-1, you can specify other algorithms, such as DSA or RSA/MD5.

The *-b* parameter specifies the key length, and *-n* is followed by the record owner, which is *ZONE* for a zone; however I will ignore other records (i.e., *HOST*, *ENTITY*, *USER*) for the purposes of this article.

The newly created key serves as the ZSK. To create a matching KSK, you need to add the *-f KSK* option to the command. This results in a file called *Kexample.net.+005+18553*, which is a concatenation of *K* for the KSK, the domain name, the encryption and hash algorithms, and a randomly generated key ID separated by plus characters. The algorithm designators are *1* for RSA/MD5, *3* for DSA, and *5* for RSA/SHA-1.

After generating the keys, listing the current directory should reveal the public key with its *.key* file extension and the private key with a suffix of *.private*. Now the public key (see Listing 2) can be added to the zone file with the *$include* directive, as shown in Listing 3.

Binding Keys to Zones

After completing this, the zone can be signed by issuing a *dnssec-signzone* command, modified as follows:

```
dnssec-signzone -o example.com ➥
-k Kexample.com.+005+42209 ➥
example.com.zone ➥
Kexample.com.005+42209
```

The *-k* option specifies the KSK.

The program now sorts the zone records, adds *NSEC* records, signs *DNSKEY* RRs with the use of ZSK and KSK, and then uses the ZSK to sign the other records. On top of this, the program cre-

Table 1: New Resource Records

Resource Record	Function Description
DNSKEY	Contains the public part of the Zone Signing Key (ZSK) and the public part of the Key Signing Key (KSK).
RDATA (R for right)	Contains both the key and details of the key type and the algorithms and protocols that DNSSEC uses. To support unique identification, the key ID is supplied; the file name includes the key. The original draft of RFC 2535 called this record *KEY*, and the ability to store any public key was supported, including user keys. This led to increased overhead client-side when following chains of trust. The new version restricts public keys to the ZSK and KSK.
RRSIG (Resource Record Signature)	Specifies the signature of a parent RR. The *RDATA* field includes the signed RR type, the algorithm used, the expiration date of the signature, and the signature itself.
NSEC (Next Entry)	Specifies the next entry in the zone file. When it reaches the last record in a zone file, *NSEC* points to the first entry. This ensures that nobody can delete entries unnoticed because the *NSEC* record is also signed by a *RRSIG*. *NSEC* has turned out to be a major issue preventing global implementation of DNSSEC because the RR supports zone walking – that is, successive requesting of all the records in a zone file.
DS (Delegation Signer)	Makes it possible to sign the validating key of a child zone, thus creating a chain of trust.

ates two new files: *dsset-example.com* and *keyset-example.com*. The new files have an extension of *.signed*. The resulting zone records are shown in detail in Listing 4, although the keys are curtailed to save space.

One *RRSIG* record for each of the original zone records is signed by the private ZSK. The server publishes the two public keys, the ZSK (256) and the KSK (257), in the *DNSKEY* RR. The key pairs sign each other reciprocally here, and the ZSK is used for all other signatures.

To prevent unauthorized removal of a zone record, the sorted RRs are linked to form a chain. Ironically, the *NSEC* RR is one of the biggest obstacles to more widespread coverage by DNSSEC. Some critics say that this leads to data protection problems because attackers can just query the chain to learn all the records in a zone in what is known as "zone walking."

After reloading the zone files, the server returns DNS responses with its own signatures. Zone signatures are valid for 30 days by default, but with the *-e YYYYMMDDHHMMSS* option, you can modify the validity period. After doing so, it is important to resign the zone manually or periodically with a script by calling *dnssec-signzone* with the required options. If you add entries to or remove entries from a zone, you must resign.

After saving the parent zone, it is possible to establish a chain of trust to extend protection to the child zones. A resolver can use a DS-RR to reference the delegated zone. A hash value in this record signs the KSK in the child zone.

Gaining Trust

The signature procedure uses *dnssec-signzone* to create two files: *dsset-example.com* and *keyset-example.com*. The administrator in the subordinate zone must send at least one of the two to the administrator in the parent zone. The DS set in *dsset-example.com* already contains a corresponding DS-RR for the parent's zone file.

After the administrator has run *dnssec-signzone* for the child zone *branch1.example.com*, a line like the following is added to the file:

```
branch1.example.com. ➥
IN DS 18890 1 1 ➥
AE9882AD0F80C91663A1A➥
DE3742B2BF2403A7283
```

In contrast, the key set in the *keyset-branch1.example.com* file has the *DNSKEY* zone file record for the KSK in the child zone.

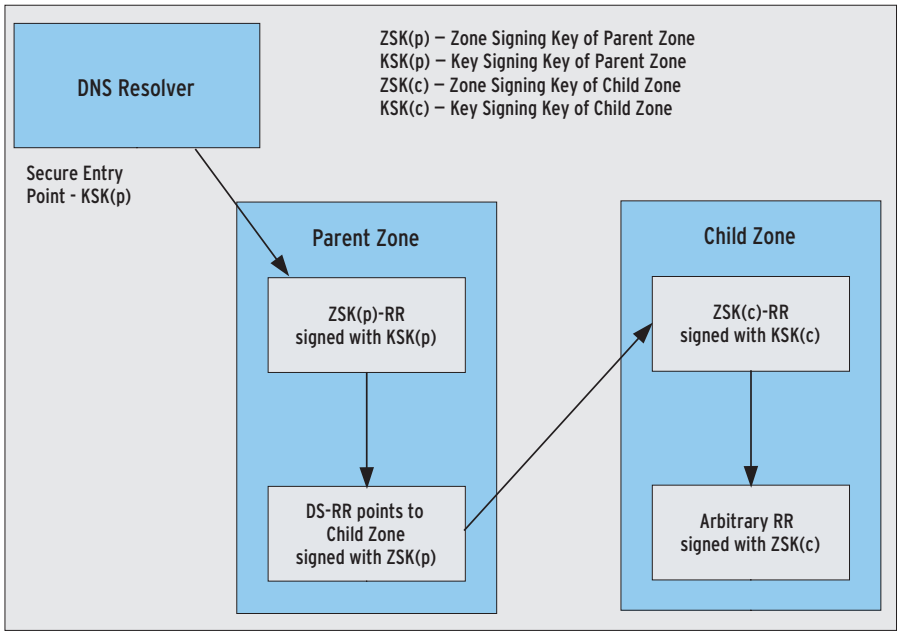

Figure 2: The KSK signs the ZSK, which signs all other zone file entries. The hierarchic PKI system is built on the same lines as the DNS structure.

This means that the admininistrator in the parent zone can set up the *DS* record by storing the key in a file with a *keyset-child* prefix; in this case, it would be *keyset-child-filiale1.example.com*.

All the files are stored in the zone file directory. Once the new files are in place, the provider needs to resign the parent zone to enable the links. Adding the *-d* option tells *dnssec-signzone* to create the corresponding *DS* record. As an alternative, you can *$include* the DS set and sign the parent's zone file.

Once the *DS* record has signed the KSK in the *branch1.example.com* child zone, and assuming a DNSSEC-enabled resolver has the parent KSK as a SEP, the resolver will now validate both the parent and the child zone. This validation can also occur for any other subordinate zones.

Listing 3: Zone File Before Signing

```
01 ; example.com zone
02 ;
03 $TTL 10
04 $ORIGIN example.com.
05
06 @            100 IN SOA ns1.example.com. (
07                  admin.example.com.
08                  2007112001
09                  100
10                  200
11                  604800
12                  100
13              )
14
15                  NS      ns.example.com.
16 ns1.example.com.  A       172.16.5.1
17 a                A       192.168.0.1
18 b                A       192.168.0.2
19
20 $include Kexample.com.+005+18553.key ; ZSK
21 $include Kexample.com.+005+42209.key ; KSK
```

If the parent zone is not secure, you can validate your own KSK through a DNSSEC Lookaside Validation (DLV) registry. ISC itself runs a DLV registry [5]. Administrators wanting to submit the KSK in their zones to the DLV registry would use the *-l* option and specify an address:

```
dnssec-signzone -l ➥
dlv.isc.org -o example.com -k ➥
Kexample.com.+005+42209 ➥
example.com.zone ➥
Kexample.net.+005+18553
```

This call writes the *dlvset-example.com* file, which the admin then mails to *dlv-registry@isc.org* along with the domain name and the administrator's name.

After the DLV registrar has verified the entry, a *DS* record that points to the applicant's zone is created. This means that the name server run by ISC is a useful SEP as long as you trust the company and its authentication processes.

To Be, or Not to Be?

DNSSEC is no panacea, but is can be a useful extension if you want to make sure that your communications partners are legitimate. Obviously, this is the case with things like online shopping and banking, but also if you need to transfer confidential data between computers that use DNS to locate each other and thus rely on name resolution.

On the client side, local clients typically do not support the protocol; therefore, widespread implementation of DNSSEC is prevented. Clients tend to rely on a local DNS server with this ability, and this is unlikely to change in the near future.

In highly sensitive environments, the use of DNSSEC on both local networks and the backbone is highly recommended. DNSSEC is used in healthcare scenarios, in which it authenticates communications partners such as doctors, pharmacists, and health insurance companies.

Additional Security

Of course, DNSSEC cannot replace other security measures, such as VPNs or public key infrastructures. Public PKIs manage certificates signed by acknowledged CAs. And if SSL/TLS use is based on this technology, the level of authenticity and trust is far more than DNSSEC can hope to offer. DNSSEC is particularly useful for protecting users who would accept untrusted certificates.

Strengths and Weaknesses

Setting up a chain of trust with DNSSEC is fairly easy, but managing one is more difficult. All the stakeholders – from the root to the last zone delegated by it – need regularly updated keys if the resolver is to work correctly.

The *NSEC* records make it possible to read all the records in a zone with zone-walking techniques. Because the developers of DNS designed the protocol to be open and freely accessible, they deliberately did not design DNSSEC for confidentiality. On the other hand, confidentiality is an unequivocal data protection objective.

Many registrars view zone walking as a data protection problem. The NSEC3 draft details a potential solution to the problem that relies on encryption. Skeptics question whether publicly resolvable DNS names are worth protecting; although they see the problem of unauthorized persons systematically listing zones, they object that other measures provide more in the line of data protection and trust. They include Access Control Lists and client authentication, but

Listing 4: Signed Zone File (continued on page 69)

```
01 ; File written on Wed Nov 20 17:02:12 2007
02   ; dnssec\_signzone version 9.4.1
03   example.com.  100  IN SOA  ns.example.com. admin.example.com. (
04                              2007112001 ; serial
05                              100        ; refresh (1 minute 40 seconds)
06                              200        ; retry (3 minutes 20 seconds)
07                              604800     ; expire (1 week)
08                              100        ; minimum (1 minute 40 seconds)
09                              )
10                 100  RRSIG   SOA 5 2 100 20070429180412 (
11                              20070330180412 17000 example.com.
12                              Q7QT/Y3MhD9Zx6/...= )
13                 100  NS      ns.example.com.
14                 100  RRSIG   NS 5 2 100 20070429180412 (
15                              20070330180412 17000 example.com.
16                              k4Dy4YRfMwTUsKt...= )
17                 100  NSEC    a.example.com. NS SOA RRSIG NSEC DNSKEY
18                 100  RRSIG   NSEC 5 2 100 20070429180412 (
19                              20070330180412 17000 example.com.
20                              fEnDtTdDyYrC7Dq...= )
21                 100  DNSKEY  256 3 5 (
22                              AQPI4+OM1VO55RS...=
23                              ) ; key id = 18553
24                 100  DNSKEY  257 3 5 (
25                              AQOzgs4qea+ImJ1...
26                              ) ; key id = 42209
27                 100  RRSIG   DNSKEY 5 2 100 20070429180412 (
28                              20070330180412 17000 example.com.
29                              hFcUzcQnsQbiOhn...= )
30                 100  RRSIG   DNSKEY 5 2 100 20070429180412 (
31                              20070330180412 49656 example.com.
32                              oyum/nlrNZ7Xdxi...= )
33 a.example.net.  100  IN A    192.168.0.1
34                 100  RRSIG   A 5 3 100 20070429180412 (
```

do not extend to freely available DNS records. Of course, the decision will be influenced by your company's security policies. Experts say that another issue preventing the introduction of DNSSEC is that a DNSSEC name server's cryptographic processes put twice as much load on the infrastructure as a normal server.

Conclusions

As is so often the case, politics also plays a role. The question of who manages the private key in the root zone is still open. On the one hand, RIPE and other registrars have asked the Internet Corporation for Assigned Names and Numbers (ICANN) to sign the root zone as soon as possible; on the other hand, some people worry about handing complete control of the private key to a US authority.

Many people regard the root zone server as the last line of defense against state intervention, and it is understandable that they do not want to place the root zone behind a private key. Global discussions have not prevented private zone administrators from testing and introducing DNSSEC. Most private zones are not affected by the NSEC data protection issue because they only contain *www*, *mail*, and other public records.

If they publish the KSK centrally – in a DLV registry, for example – third parties can use DNSSEC without any trouble.

Where personal data is at stake, as in online banking or shopping, providers can boost trustworthiness by creating a DNSSEC-protected zone. ■

Listing 4: Signed Zone File (continued page 68)

```
35                              20070330180412 17000 example.com.
36                              oN1QemG7B47dWBo...= )
37                  100  NSEC    b.example.net. A RRSIG NSEC
38                  100  RRSIG   NSEC 5 4 100 20070429180412 (
39                              20070330180412 17000 example.com.
40                              Kon6z25uqnHpGc9...= )
41 b.example.net.  100  IN A     192.168.0.2
42                  100  RRSIG   A 5 3 100 20070429180412 (
43                              20070330180412 17000 example.com.
44                              lWXfx2ebTpOBvCx...= )
```

INFO

[1] Multiple standards documents specify DNSSEC: RFC 4033 (Intro), RFC 4034 (Records), RFC 4035 (Protocol), and RFC 3658 (DS): *http://tools.ietf.org/html*

[2] DNSSEC HOWTO: *http://nlnetlabs.nl/dnssec_howto*

[3] ISC name server: *http://www.isc.org/*

[4] DNSSEC-Tools: *http://www.dnssec-tools.org/*

[5] ISC DLV registry: *https://secure.isc.org/index.pl?/ops/dlv/*