ELEKTROTEHNIŠKI VESTNIK 79(1-2): 47-54, 2012
ENGLISH EDITION

DNSSEC Key Management

Dušan Kozic[1], Benjamin Zwittnig[1], Janez Sterle[2], Andrej Kos[2]

[1] *Arnes, Jamova 39, 1000 Ljubljana, Slovenia*
[2] *University of Ljubljana, Faculty of Electrical Engineering, Tržaška 25, 1000 Ljubljana, Slovenia*
E-mail: dusan.kozic@arnes.si

Abstract. The DNS security extensions, DNSSEC, were standardized in 2005. Since the 2008 update, they have become available for general use. The implementation of the DNSSEC is a complex task, demanding software and hardware modifications throughout the entire DNS hierarchy. That is the reason why DNSSEC has only recently received more attention. The paper presents and compares current possibilities for DNSSEC implementation, which are available to DNS service providers. The authors believe that the currently accessible tools are powerful enough for widespread use.

Keywords: DNSSEC, public-key cryptography, validation, signed DNS zone, HSM

1 INTRODUCTION

Information and telecommunications systems or services are still one of the fastest growing economic sectors. The main reason lies in the rapid development of new technologies and business models that include core, aggregation, access networks, user equipment, servers, as well as a wide range of different services. Nowadays, the term "services" usually implies end-user services [9],[17],[19]-[22]. However, system and security services are also extremely important [8],[18]. One of the key system services of the IP (Internet Protocol) packet networks is the Domain Name System (DNS), which in principle did not support mechanisms for security services.

The DNS security mechanisms in use today were not primarily designed for ensuring security. All the applications chose a random source port, while the transaction ID was meant to connect DNS queries with DNS responses. The DNS system, which is one of the basic systems behind the Internet, is thus left vulnerable to a number of attacks: from eavesdropping on communications to spoofing, i.e. actively changing the content through an introduced attacker (DNS cache poisoning). [6]

In order to protect the DNS, Domain Name System Security Extensions or DNSSEC started being developed in 1995. The last specification was accepted in 2005 (RFC 4033-4035 [11][12][13]). The basic standard, tasked with introducing security extensions into the DNS system, contained a large security loophole; the attacker could gain access to the contents of the entire zone. When the latest standard (RFC 5155 [15]) was adequately fixed in 2008, it became suitable for large-scale introduction. The root zone was signed in July 2010 and most of the generic top-level domains have been signed recently. Top-level domains belonging to different countries are also in the process of being signed. However, there are currently not many signed domains owned by organizations and individuals. It is also difficult to foresee to what extent and if at all DNSSEC will catch on the latter.

DNSSEC adds an extra layer of complexity in the form of Public-Key Cryptography to an already complex DNS system. All the DNS responses are thus digitally signed, while the messages are not encrypted during transfer. DNSSEC consequently ensures authentication and integrity, but not private communication. With the implementation of DNSSEC, work has become significantly more intricate and complex. The possibility for mistakes is similarly greater. Even though there is no wide-spread demand for DNSSEC services yet, individuals have already opted for it. Internet service providers will thus have a hard time avoiding the implementation and use of DNSSEC in their environments.

The paper looks at the most difficult DNSSEC tasks, i.e. tasks related to the maintenance of zones and the keys used to sign them. The chapter 2 deals with the problems of zone and key maintenance. The chapter 3 offers a basic description of the selected tools for this task. A comparison is given of two commercial and one open-source tool. The chapter 4 presents results of a technical tool comparison, carried out by signing one large and 400 middle-sized zones and thus simulating the environment of a smaller internet service provider. Finally, the tools are compared from the point of view of user friendliness and the controlling system operations.

Received April 23, 2012
Approved May 7, 2012

2 CRYPTOGRAPHIC KEY AND ZONE MAINTENANCE

In the event of managing smaller zones, the DNS system implementation usually represented a one-time task. The system administrator needed to intervene only during changes. Consequently, the administrator sometimes never had to deal with certain zones during a period of several years.

The DNSSEC needs to ensure security services with an adequate security policy, meaning the zone file should no longer be static:

- The zone signature has a limited validity date, which means the zone needs to be resigned before the keys expire;
- The KSK (Key Signing Key) and the ZSK (Zone Signing Key) need to be periodically rolled over[1].

If the encryption keys are not regularly rolled over, they risk being discovered in the long run. Modern cryptography is based on the assumption that when using the all-possible-combinations approach to uncover a code, one needs only enough time and a large amount of processing power. With time, it consequently becomes more probable that the encryption keys will be discovered.

2.1 Cryptographic Key Rollover

Resigning a zone is relatively simple. The administrator needs only to resign a zone using valid keys – the existing DNS solutions usually enable this through one command or mouse click.

A key rollover is a more complex matter, as the keys need to be adequately generated before resigning the zone and the old ones need to be safely destroyed.

In principle, this is not enough because the DNS system is based on caching responses for a certain amount of time (Time to Live or TTL). When the recursive resolver receives the DNS response, it stores it in cache for a set time defined by the TTL. This means the server will only start a new query when the TTL expires. If the zone was signed only with a new key while the DNS system key was being rolled over, the following scenario could occur:

1) A user searches for www.dnssec.si on a recursive resolver.
2) The recursive resolver does not have the name in its cache and so continues with its queries.
3) It receives a response and a digital signature for www.dnssec.si.
4) The recursive resolver already has the DNSKEY for dnssec.si in its cache because the other users searched for the domain, so it does not start a new query.

The recursive resolver thus verifies the signature of the address www.dnssec.si using the old DNSKEY and the DNSSEC validation gives a negative result.

2.2 ZSK Rollover

There are two ways to rollover keys: the pre-publish method and the double-signature method.

Using the pre-publish method, the new key is published before it is used to sign the zone. The procedure is as follows:

- For at least twice the TTL[2] time before the zone expires, the new key is published together with the old DNSKEY;
- The zone is still signed with only the old key;
- After the TTL expires, the zone is signed with the new key only, however the old one still remains in the zone;
- After waiting for twice the TTL time, the old key is removed from the zone.

With the double-signature method, the zone is signed using the new and the old key at the same time. After the TTL passes, the old key is removed from the zone and the zone is then signed using only the new one.
When rolling over the ZSK, the pre-publish method is more practical, regardless of the required double TTL. This method ensures that the signed zone's file is twice as small as it would have been using the double signature method. The DNS messages are also twice as short. The double-signature method is imperative when also replacing the algorithm used to sign the DNS messages [1].

2.3 KSK Rollover

When rolling over the KSK, it is important to publish the DS record to the parent domain server. It is thus essential to take the TTL of the DS record into account, the former usually being longer than the TTL from our zone[3]. When doing a KSK rollover, the double-signature method is used, since the pre-publish method would be time consuming. The method also does not save much DNS response space, because only a ZSK is signed with a KSK. The procedure is as follows:

- The new KSK and its DS record in the parent zone are published for at least the TTL, before the zone or the DS record in the parent zone expire;
- The ZSK is signed with the new and the old key;
- After the TTL, the old KSK is withdrawn from the zone, together with the key's parent zone DS record. [1]

[1] The ZSK is used to sign all the records in the zone, while the KSK is used to sign the ZSK.

[2] The value of TTL is determined by the longest TTL in the zone.
[3] The longest TTL always needs to be taken into account.

3 METHODOLOGY AND THE TEST ENVIRONMENT

In order to facilitate the transition of service providers to the DNSSEC, the authors of the paper decided to compare different tools, which began to emerge in the course of DNSSEC development. The tools were compared according to their technical capabilities and user experiences. The technical capabilities aspect included:

- Operational reliability
- Working speed
- Possibility of operating in a high-availability mode
- Compliance with the FIPS (Federal Information Processing Standard) [5]
- Support for various DNSSEC algorithms
- Transition between the DNSSEC algorithms

The user experiences aspect included a look at the:

- User interface simplicity
- Amount of the DNSSEC knowledge expected from the user
- API support
- Monitoring the DNSSEC tasks
- Customer support

For domain administrators using the DNSSEC and their own DNS servers, zone maintenance will not present much of a problem. However, one can imagine the following DNS service provider scenario. DNS service providers will have to ensure a quality DNSSEC service to a number of users, including:

- Automatic zone resigning
- Automatic ZSK and KSK rollovers
- Automatic DS record messaging to the parent zone
- Possibility of transition between different DNSSEC algorithms
- Possibility of switching between different DNSSEC tools

3.1 Setting up the Environment

The authors set up a test environment comprising one large zone with 100,000 delegations on different DNS servers, as well as 400 middle-sized zones delegated on the same DNS server. The first zone stood at 100 MB, while an individual middle-sized zone was about 100 kB.

The authors used an open-source tool, OpenDNSSEC (version 1.3.0) [10]; among the commercial tools, they opted for the Infoblox solution (version 5.1) [7], the BlueCat Proteus (version 3.5), and the BlueCat Adonis (version 6.5) [4]. OpenDNSSEC is a dedicated solution for signing DNSSEC, while BlueCat and Infoblox are the so called IPAM solutions (Internet Protocol and Address Management), combining the functions of the DHCP server and a DNS server.

The logical schema of the test environment (Figure 1) comprised two units: the signing server and a DNS server serving the signed zones.

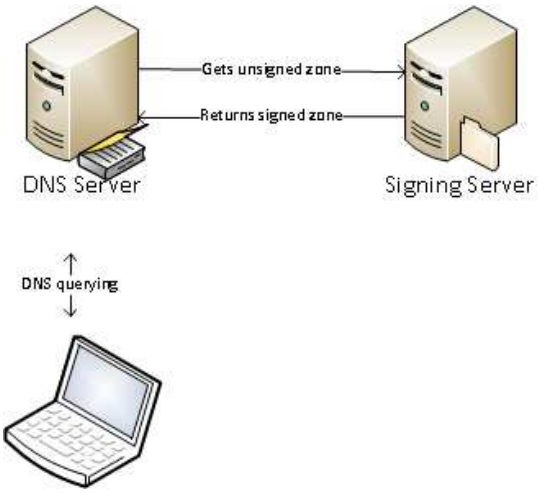

Figure 1. Logical schema of the DNSSEC test environment

The set up environment was somewhat different, depending on the tools used. When using OpenDNSSEC and Infoblox, the signing server and the DNS server were being run on the same computer with a Linux OS. The BlueCat tool required two servers: BlueCat Proteus was the central server monitoring the Adonis server, which was the active DNS server. The zones, the signing policy, and the keys were all located on the Proteus server, which redistributed to the Adonis server.

Both commercial tools, Infoblox and BlueCat, are based on Linux and use the most common open-source server Bind for the DNS server [3]. They also use this server's tools for signing zones. The latter also ensures the resigning of zones before they expire. No additional tools are used to verify the validity of the signed zones. They are only checked by the Bind server, which does this before loading them. Both tools monitor the DNSSEC policy themselves. This includes generating new keys and rolling over the ZSK and KSK. The data (zones, keys) are saved onto the database.

In order to compare performance, the trial versions of the tools were set up as virtual computers in the VMware ESX environment. The trial license for the Infoblox tool limited the processing power to 2000 MHz and the RAM to 2 GB. Consequently, other tools were given the same limits. BlueCat and Infoblox are *de facto* purpose-built appliances. However, the manufacturers enable testing on virtual computers. The authors of the paper chose to do so.

3.2 High Availability

It is essential that there is a redundant location to ensure the fastest possible continuation of DNSSEC zone maintenance in the event of a malfunction on one of the servers.

The BlueCat tool supports the XHA (Crossover High Availability) method. In this case, two identical BlueCat devices, communicating through an assigned network interface, are interlinked. In the event of a device malfunction, the switch to a secondary is automatic. BlueCat also supports the replication of a database onto BlueCat devices on standby. The database can be replicated onto several devices, which can be at different locations.

The Infoblox tool is based on the so called grids. One of the devices is the Master replicating its data onto the other grid members. Similarly to the BlueCat tool, we can prepare a high-availability pair for the Master. We can additionally define the remaining roles of the other devices within the grid, as well as the services running on them.

The OpenDNSSEC solution is essentially not designed to operate in the high-availability mode, however we can set up such an environment ourselves. The server settings can be saved in the MySQL database, which supports the replication onto other systems. When dealing with keys on the HSM, we can choose a physical HSM that works in a high-availability mode, while SoftHSM enables the periodic redistribution of the key database. OpenDNSSEC also allows the advance generation of keys for a set period (ex. five years). These can then be copied onto the secondary system. The latter is then ready to take over from the primary, should it go offline[4].

3.3 Security

In modern cryptography, data security depends on the length, randomness, and secrecy of the keys. When dealing with DNSSEC, the location of the saved keys is thus very important. It is never completely possible to discount the possibility that a hacker will gain access to the DNS server. Consequently, it is not advisable to save the keys there.

The secure saving of sensitive data is facilitated by the FIPS 140-X [5]. FIPS sets the standards for cryptographic modules. One criterion is the location where cryptographic keys may be kept, how they may be generated, and how destroyed. Devices, guaranteeing an adequate level of security, need to be properly certified. The BlueCat and Infoblox tool are not certified according to FIPS. However, both manufacturers claim that the design of the tools is in accordance with these standards. The two tools are questionable because the keys are not only saved in the database, but also in the Bind server files. Bind has the keys because for the dynamic DNS. It additionally needs to take care of the zone resigning on its own. In the case of the BlueCat tool, where zone signing takes place on the Adonis server, this means that private keys are periodically transferred onto the servers together with the zones, which from the security standpoint is far from optimal.

OpenDNSSEC supports the use of HSM modules (compatible with different FIPS 140-X) for generating and saving keys, as well as for signing zones. These modules can be locked, which means the attacker cannot gain access to the keys, even during a direct break into the system.

3.4 Data-Loss Prevention

Should an attacker get hold of our private keys, they need to be immediately revoked in accordance with the RFC 5011 standard [14]. The next step is to generate new ones and use them to resign the zone. This kind of key revocation is possible in BlueCat, while the other two tools do not support this function.

Should we wish to restore the system to its previous state, before data was lost, we need backup copies. BlueCat and Infoblox, which keep the entire configuration and the keys in the database, support backup copies. Infoblox also supports the export of individual keys through the API (Application Programming Interface). The format of the exported file is proprietary. BlueCat does not enable the export of individual keys through the API.

OpenDNSSEC does not have a central database. Consequently, it is not enough to only back up the files containing the zones, but also the configuration files, the configuration database, and the key repositories. When working with the SoftHSM repository, it is also possible to export individual keys into the PEM file, which is compatible with the PKCS#8 standard [16]. Exporting keys also depends on the HSM used.

3.5 Switching Between Tools

Exporting zones is relatively easy. All of the tools enable the replication of zones onto other DNS servers (Bind, NSD, and Microsoft DNS). However, as already mentioned, exporting keys is not possible in BlueCat, while Infoblox uses a proprietary format to export them. It is difficult to switch from these two DNSSEC tools to the third one. It is true that both tools also save the keys in the Bind server files. If we manage to successfully transfer these files, switching to the third tool becomes much easier.

OpenDNSSEC, together with the SoftHSM software repository, also offers a tool for migrating between exported keys in the PKCS#8 format and the files on the Bind server. If using another HSM security module, the export and import of keys depends on the manufacturer.

Generally, there are two ways of switching:

- By importing private keys from one tool to the other
- By importing the DNSKEY record on the first tool to an unsigned zone on the second

When importing private keys from the first tool to the second, the switch from the commercial solutions to the

[4] The secondary system does not take over automatically. The switch requires manual intervention from the administrator.

OpenDNSEC is possible, as long as we manage to export the private keys from them. However, the transition from OpenDNSSEC to the commercial tools is not possible, as it is impossible to import private keys. Switching from OpenDNSSEC to the Bind server tools and vice versa is quite easy. On the other hand, the switch from the commercial solutions to the Bind server tools depends on whether it is possible to transfer the Bind files containing the keys from the latter. Switching from the Bind tools to the commercial tools is not possible; neither is transferring between BlueCat and Infoblox.

Another approach would include adding the DNSKEY record into an unsigned zone. The procedure is as follows:

- We use the first tool.
- We generate a ZSK/KSK key pair in the second tool.
- We insert the DNSKEY records from the second tool into an unsigned zone version.
- We continue to sign the zone with the first tool for some time.
- The keys from the first tool are inserted into the unsigned zone of the second tool in the form of DNSKEY records.
- We begin to sign the zone with the second tool.

BlueCat, as well as Infoblox, do not allow the insertion of DNSKEY records into an unsigned zone. OpenDNSSEC on the other hand does, however it would need to be used in conjunction with one of the other tools.

4 Comparing Key and Zone Maintenance Tools

4.1 Defining Policies

The algorithms supported by different tools are presented in Table 1, Table 2, and Table 3. As can be discerned from the tables, OpenDNSSEC does not support DSA algorithms. Infoblox supports all of them, except the use of the opt-out with negative responses. BlueCat also does not support the use of the opt-out with negative responses. In addition, when using NSEC3, it does not use salt, it does not support NSEC3 with the newer RSA/SHA-256 algorithm, and neither does it support RSA/SHA-512.

Table 1. Algorithms supported by different tools

Tool	Supported Algorithm
BlueCat	RSA/SHA-1, RSA/SHA-256, DSA
Infoblox	RSA/SHA-1, RSA/SHA-256, RSA/SHA-512, DSA
OpenDNSSEC	RSA/SHA-1, RSA/SHA-256, RSA/SHA-512

Table 2. Support for NSEC, NSEC3

Tool	Negative Responses	Salt Use	Opt-Out
BlueCat	NSEC, NSEC3[5]	No	No
Infoblox	NSEC, NSEC3	Yes	No
OpenDNSSEC	NSEC, NSEC3	Yes	Yes

Table 3. Support for DS record algorithms

Tool	DS Record
BlueCat	SHA-1
Infoblox	SHA-1, SHA-2
OpenDNSSEC[6]	SHA-1, SHA-2

The tools enable the creation of different policies and the logical transfer of the zones into them. OpenDNSSEC and BlueCat require strict application of a zone into a policy, while Infoblox has a DNS view. Infoblox allows signing parameters to be changed on an individual zone[7].

The authors first tried to sign the zones using algorithm 8 (RSA/SHA-256) and to employ NSEC3 for the negative responses. However, it turned out that one of the tools does not support the use of NSEC3 for negative responses with this algorithm. It was thus decided to sign them with algorithm 7 (RSA/SHA-1 NSEC3). The following policy was used to sign the zones:

- Algorithm: 7
- Negative Responses: NSEC3 (no Opt-Out)
- KSK Size: 2048 bites
- ZSK Size: 1024 bites
- Signature Duration: 7 days
- ZSK Validity Period: 11 days
- KSK Validity Period: 14 days
- ZSK Rollover Method: key pre-publishing
- KSK Rollover Method: double signature

4.2 Zone signing

Before being signed, the zones needed to be imported into the individual tools. Importing into OpenDNSSEC was the easiest. It is run on Linux, which offers several ways to perform this task. In the test environment, they were imported through the SSH protocol. For this purpose, the authors wrote a script for Infoblox that employs the functions of the API tool. The script followed a zone list, initiating a zone transfer from the DNS server for each one. When using BlueCat, the zones were imported through the data-import function using an XML file. The XML file was prepared from the Bind zone files using a tool supplied by BlueCat.

[5] NSEC3 is not supported when using RSA/SHA-256.
[6] DS is manually inserted into the parent zone.
[7] None of the zones needs its own created policy.

The signing on OpenDNSSEC was initiated by adding the zone onto the zone list and allocating the corresponding policy. When working with Infoblox and BlueCat, we need to determine whether a zone is using DNSSEC or not. The authors again used the script on the Infoblox system, which followed the list and informed the system to sign each zone. BlueCat presented some difficulties in this regard, as only an individual zone can be signed. Additionally, this can only be done through a graphical interface (BlueCat API does not support DNSSEC-related tasks). The authors were thus left with no choice but to perform the task manually, using the graphical interface. The time spent generating 401 2048-bit KSKs and 401 1024-bit ZSKs is shown in Table 4.

Table 4. Key-generation times

Tool	Generating 401 Keys	
	2048-bit KSK	1024-bit ZSK
BlueCat	16 min	2 min
Infoblox	n/a	n/a
OpenDNSSEC	9 min	1 min 30 sec

It was impossible to estimate the time when using Infoblox, since the keys were being generated at the same time as an individual zone was being signed. With OpenDNSSEC, keys were generated when the service began running. BlueCat also generated keys as soon as an individual zone was marked as when using DNSSEC. However, it was subsequently possible to measure the times when the keys were being replaced and regenerated. In this case, the keys were generated before the zones were signed.

An important parameter when generating keys is the possibility of generating them in advance. This solution comes in handy, especially when we wish to lock the HSM. Writing onto the security module is impossible after it has been locked. We also have the option of generating the keys beforehand and transferring them to the tool on standby. After transferring the keys to a redundant location, the HSM needs to be locked. When not using a special FIPS-compatible module, the advance generation of keys is quite risky. When an attacker gains access, they also get hold of the future keys. OpenDNSSEC gives us the option of generating keys in advance. It is also the only solution that offers key sharing between zones (i.e. several zones can be signed with the same key).

Once the keys had been successfully generated, it was time to sign the zones. The time it took to sign and resign the 401 zones is presented in Table 5.

Table 5. Zone-singing times

Tool	First Signing of the 401 Zones	Resigning of the 401 Zones
BlueCat	1 h	1 h 11 min
Infoblox	n/a[8]	n/a
OpenDNSSEC	3 h 17 min	3 h 11 min
OpenDNSSEC without Auditor tool	1 h 55 min	2 h 14 min

BlueCat and Infoblox sign zones using the Bind tools. Zone signing is already integrated into OpenDNSSEC, which does not use external tools for this task. Signing in OpenDNSSEC also takes longer because the solution offers the Auditor tool. It checks whether the zones have been correctly signed. Only when the zone has been successfully verified can it be installed onto the DNS server. BlueCat and Infoblox do not additionally verify the signed zones. The zones are checked by the Bind server before they are installed. However, additional verification of the signed zones is not a shortcoming in itself.

When signing a zone, it is important to pass on the DS record of the KSK to the parent zone. Since the dnssec.si parent zone and its subdomains were loaded onto the same system, the transfer of the DS records of the subdomains to the dnssec.si parent zone can be automatic. BlueCat and Infoblox used the same approach to transfer DS records, while OpenDNSSEC did not. However, OpenDNSEC supports the use of scripts that enable the automatic transfer of DS records through the EPP (Extensible Provisioning Protocol).

Tests have shown that the resigning of the zones lasts approximately as long as the first signing. When using BlueCat and Infoblox, the resigning is taken care of by the Bind server, while OpenDNSSEC performs this operation on its own.

4.3 Key Rollover

OpenDNSSEC rolls over the ZSK according to the pre-publish method and the KSK according to the double signature method. Infoblox uses the double signature method to rollover the two keys. BlueCat on the other hand enables the user to choose the desired method. All the tools generate keys before rolling them over. OpenDNSSEC does not generate them if they have already been generated.

Key rollovers were made in accordance with policies defined in Chapter 4.1. When doing a preliminarily key rollover, BlueCat first writes it into the zone and signs with the KSK. The zone was not resigned, so such an operation took only a few minutes. During the next time

8 Our test data exceeded the Infoblox database-size limitation. It was thus impossible to measure the times for this tool.

interval, the zone is signed with a new ZSK, while the old one remains in the zone. This operation was consequently as time consuming as the zone-signing operation. When doing a KSK rollover according to the double signature method, the zone with the existing ZSK was resigned in addition to the ZSK. This represented an unnecessary and time-consuming operation. When rolling over the KSK, the DS record in the parent zone changed accordingly. We have to be careful when changing the DS records in the zone which are not being managed with the tool. The keys are rolled over automatically, regardless of whether they are adequately transferred and published in the parent zone. The authors of this paper were not informed that the DS record needed to be changed through the graphical interface.

OpenDNSSEC automatically did a ZSK rollover, while the KSKs can never be rolled over automatically. The administrator will have to manually rollover the KSK.

When using Infoblox, the ZSK is also rolled over automatically, while the administrator needs to perform the KSK rollover manually. When the time to do the key rollover draws near, the administrator is notified and given the option of initiating the KSK rollover through the graphical interface. The DS records in the zones on the local tool are updated automatically, while the zones outside our administrative area need to receive new records manually.

Changing the algorithms is not possible in OpenDNSSEC. Infobox allows the algorithms to be changed, however switching between NSEC and NSEC3 is not possible. BlueCat supports switching between NSEC and NSEC3.

OpenDNSSEC correctly signs the zones, however the OpenDNSSEC process is unstable. When it runs out of RAM, it crashes (it does not use disk cache). All the tools had problems with system resources when faced with the level of difficulty used in the test.

5 User Experiences

Using both tools with the graphical interface is easy. In order to set up the DNSSEC, practically no prior knowledge is required. When using Infoblox, DNSSEC turns on with just one click. BlueCat requires the user to define a policy beforehand and then connect it to the zone (an additional step is required). It should be noted that the pre-set policy on both tools is set to the RSA/SHA-1 NSEC algorithm, which means that using the default settings is not the safest option, since the possibility of “zone walking” exists (an indirect transfer of the entire zone). The solutions allow for easy modifications to the default settings and policies. Both tools have an excellent help interface and provide good documentations. It is not possible to browse through the signed zones on BlueCat, since the zones on the Proteus system (access is gained through the graphical interface) are unsigned. They are signed later on the Adonis server. On the other hand, working with OpenDNSSEC is complicated. Users need to be quite knowledgeable about DNSSEC and need to install the tool by themselves. There is a bonus, however: it is possible to adjust more specific parameters, something that is not possible on the commercial solutions.

Both commercial tools allow work to be carried out through the API, which was mostly used by the authors to automate repetitive tasks. The BlueCat API uses a standardized access through SOAP (Simple Object Access Protocol). However, it does not support all of the functionality offered by the tool through the graphical interface. Infoblox on the other hand enables all the tasks it supports through the API. For now, OpenDNSSEC does not support work through the API, but this is also not a prerequisite. The user is not limited by the graphical interface. The Linux shell allowed the researchers to automate many tasks.

Troubleshooting is more limited with the commercial products. Access to the log files is possible, however reviewing them in the graphical interface is somewhat difficult due to the large amount of data. BlueCat allows us to copy log files from the server, but not on the Adonis system, where it is difficult to access old log files. OpenDNSSEC entries into the log files are very precise, so the possibility for troubleshooting is greater. The commercial tools support user notification through the SNMP (Simple Network Management Protocol) and e-mail. The user is thus notified, for example, when the key rollover will take place.

During the tests, the authors received very good and quality support from BlueCat Networks. Infoblox also offered some support. In light of all the support received during testing, one can assume that customer support is even better. As with any other open-source solution, OpenDNSSEC support is available through various open-source communities (e-mail lists and forums).

6 Conclusion

The authors tested two commercial and one open-source tool for key and zone maintenance in order to facilitate the supply of quality DNSSEC services to DNS service providers. These will become indispensable in the future. The tests, performed on one large and 400 middle-sized zones, examined their technical characteristics, such as DNSSEC-standard compliance, operational speed and reliability, safety and high availability. The researchers also looked at the user friendliness and tool difficulty.

Owing to the limits imposed by the trial license, the operational-speed test was too difficult for one of the tools. The other two solutions fared well, despite the great system resource limitations. The tools implement most of the DNSSEC standards. All of them can work

reliably in high-availability mode, but there is no possibility of connecting external tools in order to check if the zones were correctly signed. In terms of security, it was discovered that commercial tools lack FIPS-standard support. The commercial tools are easier to use and have a friendlier user interface than the open-source solution.

Since DNSSEC system errors make Internet domains inaccessible, the authors of the paper will continue their research by analyzing and introducing tools that will check the adequacy of the signed zones and by discovering errors even before the zone is published on the Internet.

ACKNOWLEDGEMENTS

Special thanks go to Frey Khademi at BlueCat Networks for providing excellent support for the testing of their DNSSEC tool. Thanks are also due to the Infoblox technical support. We are greatly thankful to both companies for allowing us to test their DNSSEC tools.

REFERENCES

[1] P. Albitz, C. Liu, DNS and Bind, Sebastopol: O'Reilly Media, 2006, chapter 11.

[2] A Review of Administrative Tools for DNSSEC – Spring 2010, http://www.iis.se/docs/DNSSEC-Admin-tools-review-Final.pdf, accessed in August 2011.

[3] Bind Documentation, http://www.isc.org/software/bind/documentation , accessed in August 2011.

[4] BlueCat IPAM, http://www.bluecatnetworks.com/, accessed in August 2011.

[5] FIPS Publications, http://csrc.nist.gov/publications/PubsFIPS.html, accessed in August 2011.

[6] C. Florent, Security Issues with DNS, SANS Institute, 2003, http://www.sans.org/reading_room/whitepapers/dns/security-issues-dns_1069, accessed in July 2011.

[7] Infoblox Administrator Guide, http://ww2.infoblox.com/support/tech_lib/NIOS/NIOS_AdminGuide_5.1r2.pdf, accessed in August 2011.

[8] KOS, Andrej, BEŠTER, Janez. Evolucija hrbteničnih IP-omrežij v smeri MPLS. Elektrotehniški vestnik. [Slovenska tiskana izd.], 2001, year 68, N. 4, p. 200-206. [COBISS.SI-ID 2505556]

[9] KOS, Andrej, BEŠTER, Janez. Razvoj in uvajanje novih telekomunikacijskih storitev. Elektrotehniški vestnik. [Slovenska tiskana izd.], 2002, year 69, N. 3-4, p. 221-226. [COBISS.SI-ID 3318100]

[10] OpenDNSSEC, OpenDNSSEC Documentation, http://www.opendnssec.org/documentation/, accessed in August 2011.

[11] RFC 4033, http://www.rfc-archive.org/getrfc.php?rfc=4033, accessed in July 2011.

[12] RFC 4034, http://www.rfc-archive.org/getrfc.php?rfc=4034, accessed in July 2011.

[13] RFC 4035, http://www.rfc-archive.org/getrfc.php?rfc=4035, accessed in July 2011.

[14] RFC 5011, http://www.rfc-archive.org/getrfc.php?rfc=5011, accessed in August 2011.

[15] RFC 5155, http://www.rfc-archive.org/getrfc.php?rfc=5155, accessed in July 2011.

[16] RFC 5208, http://www.faqs.org/rfcs/rfc5208.html, accessed in August 2011.

[17] SEDLAR, Urban, ZEBEC, Luka, BEŠTER, Janez, KOS, Andrej. Bringing click-to-dial functionality to IPTV users. IEEE commun. mag. (Print). [Print ed.], Mar. 2008, vol. 46, no. 3, p. 118-125, illustr. [COBISS.SI-ID 6398548]

[18] STEGEL, Tine, STERLE, Janez, SEDLAR, Urban, BEŠTER, Janez, KOS, Andrej. SCTP multihoming provisioning in converged IP-based multimedia environment. Comput. commun.. [Print ed.], 2010, vol. 33, no. 14, p. 1725-1735, illustr. [COBISS.SI-ID 7960404]

[19] ŠTERN, Andrej, KOS, Andrej. Mobilni telefon kot orodje na področjih varovanja zdravja = Mobile phone as a tool in the areas of health protection. Zdrav Vestn (Tisk. izd.). [Tiskana izd.], nov. 2009, year 78, no. 11, p. 673-684, illustr. [COBISS.SI-ID 7486548]

[20] UMBERGER, Mark, HUMAR, Iztok, KOS, Andrej, GUNA, Jože, ŽEMVA, Andrej, BEŠTER, Janez. The integration of home-automation and IPTV system and services. Comput. stand. interfaces. [Print ed.], Jun. 2009, vol. 31, no. 4, p. 675-684, illustr. [COBISS.SI-ID 7093332]

[21] VOLK, Mojca, GUNA, Jože, KOS, Andrej, BEŠTER, Janez. Quality-assured provisioning of IPTV services within the NGN environment. IEEE commun. mag. (Print). [Print ed.], May 2008, vol. 46, no. 5, p. 118-126, illustr. [COBISS.SI-ID 6532436]

[22] ZEBEC, Luka, HUMAR, Iztok, BODNARUK, Darko, KOS, Andrej, BEŠTER, Janez. NGN service development - overview and Parlay X implementation. Elektrotehniški vestnik. [Slovenska tiskana izd.], 2005, year 72, no. 1, p. 45-51, illustr. [COBISS.SI-ID 4771156]

Dušan Kozic graduated in 2011 from the Faculty of Computer and Information Science, University of Ljubljana. He works as a DNS system administrator at Arnes (The Academic and Research Network of Slovenia). His research interests include the field of network security.

Benjamin Zwittnig graduated in 1990 from the Faculty of Natural Sciences and Technology – Faculty of Mathematics, University of Ljubljana. He has been employed with Arnes since 1992. During that time, he has been overseeing the technical development of domain registration in conjunction with managing top-level DNS servers for .SI.

Janez Sterle graduated in 2003 from the Faculty of Electrical Engineering, University of Ljubljana. He works at the Laboratory for Telecommunications (LTFE) at the same faculty. His research fields are next generation IP, network security, traffic planning and traffic analysis as well as development and implementation of integrated services into landline and next-generation mobile networks.

Andrej Kos received his PhD degree in 2003 from the Faculty of Electrical Engineering, University of Ljubljana. The same year, he was nominated assistant professor for the field of electrical engineering and in 2009 associate professor. He holds a combined post of professor-researcher, lecturing subjects from the field of telecommunications and multimedia. His research is focused on telecommunications, multimedia, Internet networks, systems on access, aggregation and backbone layer, testing, traffic analyses and sources optimization, driver protocols and development of convergent multimedia services.