Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 684180, 7 pages
doi:10.1155/2012/684180

Research Article

Improving DNS Security Using Active Firewalling with Network Probes

Joao Afonso[1] and Pedro Veiga[2]

[1] *Foundation for National Scientific Computing, Avenue Brazil no. 101, 1700-066 Lisbon, Portugal*
[2] *Department of Informatics, University of Lisbon, Edifício C6, Campo-Grande, 1749-016 Lisbon, Portugal*

Correspondence should be addressed to Joao Afonso, joao.afonso@fccn.pt

Received 13 December 2011; Accepted 23 March 2012

Academic Editor: James Park

The security problems that outbreak network services today are increasing at a dramatic pace especially with the unceasing improvement of network transmission rates and the sheer amount of data exchanged. This translates not only more incidents but also new types of attacks with network incidents becoming more and more frequent. A significant part of the attacks occurs at Top Level Domains (TLDs) who have the assignment of ensuring the correct functioning of Domain Name System (DNS) zones. The proposed solution has been developed and tested at FCCN (Foundation for National Scientific Computing), the TLD manager for the .PT domain. The system consists of network sensors that monitor the network in real-time and can dynamically detect, prevent, or limit the scope of the attempted intrusions or other types of attacks to the DNS service, thus improving its global availability.

1. Introduction

The DNS is a critical application for the reliable and trustworthy operation of the Internet. DNS servers assume a pivotal role in the normal functioning of Internet Protocol (IP) networks today and any disturbance to their normal operation can have a dramatic impact on the service they provide and on the global Internet. Although based on a small set of basic rules, stored in files, and distributed hierarchically, the DNS service has evolved into a very complex system and critical system [1].

According to recent studies [2], there are nearly 11.7 million public DNS servers on the Internet. It is estimated that nearly 52% of them, due to improper configuration, allow arbitrary queries (thus allowing denial of service attacks or "poisoning" of the cache). About 31.1% of the servers also allow for the transfer of their DNS zones.

There are still nearly 33% of situations where the authoritative name servers of an area are on the same network, which facilitates the attacks of the type of Denial of Service (DOS), a frequent attack to the DNS. Furthermore, the type of attacks targeting the DNS is becoming more sophisticated, making them more difficult to detect and control on time. Examples are the attacks by Fast Flux (ability to quickly move the DNS information about the domain to delay or evade detection) and its recent evolution to Double Flux.

One of these attacks is the conficker [3] worm, first appeared on October 2008, but also known as Code Red, Blaster, Sasser, and SQL Slammer. Every type of computer, using a Microsoft Operating System can potentially be infected. Attempts to estimate the populations of conficker have led to many different figures but all these estimates exceed millions of personal computers. Conficker made use of domain names instead of IP address in order to make its attack networks resilient against detection and takedown.

The ICANN (Internet Corporation for Assigned Names and Numbers) created a list containing the domains that could be used in each TLD in such attacks to simplify the work of identifying attacked domains.

A central aspect of the security system that we propose and have implemented is the ability to statistically collect useful data about network traffic for a DNS resolver and use it to identify classes of harmful traffic to the normal operation of the DNS infrastructure. In addition to collecting data,

the system can take protective actions by detecting trends and patterns in the traffic data that might suggest a new type of attack or simply to record important parameters to help improve the performance of the overall DNS system.

The fact that the DNS is based on an autonomous database, distributed by hierarchy, means that whatever solution we use to monitor, it must respect this topology. In this paper, we propose a distributed system using a network of sensors, which operate in conjunction with the DNS servers of one or more TLDs, monitoring in real-time the data that passes through them and taking actions when considered adequate.

The ability to perform real-time analysis is crucial in the DNS area since it may be necessary to immediately act in case of abuse or attack, by blocking a particular access and notifying other cooperating sensors on the origin of the problem, since several types of attacks may be directed to other DNS components.

The use of a Firewall solution, whose triggering rules are dynamically generated by the network sensors, is a fundamental component of the system used to filter attacking systems in an efficient way and resuming to the initial situation when the reason to filter different traffic patterns has ceased to exist.

With this approach we aim to guarantee an autonomous functioning of the platform without the need of human intervention.

The use of network alarms can also help in monitoring the correct functioning of the whole solution. Special care has been taken to minimize the detection of false positives or also false negatives.

The remaining of the paper is structured as follows. In Section 2 we observe the DNSSEC approach. Section 3 provides background information regarding related work. Section 4 introduces our proposed methodology. In Section 5, we describe the solution. Section 6 presents a case study for validation of the proposal. In Section 7, the results gathered in the case study are analyzed. Section 8 describes the process of evaluation and treatment of false positives and negatives. Finally, Section 9 presents some conclusions and directions for further work.

2. DNSSEC

The use of DNSSEC (DNS Security Extension to DNS protocol) described in RFC 4033, 4034, 4035, and 4310, seeks to ensure backward compatibility with the DNS protocol providing additional security to the end user, solving some of the vulnerabilities existing.

It becomes possible to ensure the authenticity of the origin of DNS information (thus answering the vulnerability of the recursive or authoritative impersonation). The integrity of the information provided shall also be guaranteed by fighting in this way the modification of an answer given by a server, with the typical man-in-the-middle attacks, where data is modified from the information provider typically a DNS server and his client. Another advantage is the possibility to ensure the absence of a name or a given type.

Despite these gains, this extension to the protocol does not ensure the confidentiality of the information provided or mitigate security incidents which relate to the denial of service.

3. Related Work

One of the first studies that can be observed in this area has the authorship of Guenter and Kolar, with a tool called sqldjbdns [4]. Their proposal uses a modified version of the traditional BIND [5] working together with a Structured Query Language (SQL) version inside a Relational database management system (RDBMS). For DNS clients, this solution is transparent and there is no difference from classic BIND.

Zdrnja presented a system for Security Monitoring of DNS traffic [6], using network sensors without interfering with the DNS servers to be monitored. This is a transparent solution that does not compromise the high availability needed for the DNS service.

Vixie proposed a DNS traffic capture utility called, DNSCap [7]. This tool is able to produce binary data using pcap format, either on standard output or in successive dump files. The application is similar to tcpdump [8]—command line tool for monitoring network traffic—and has finer grained packet recognition tailored for DNS transactions and protocol options, allowing for instance to see the full DNS message when tcpdump only shows a one-line summary.

Another tool available is DSC-DNS Statistics Collector [9]. DSC is an application for collecting and analyzing statistics from busy DNS servers. Major features include the ability to parse, summarize and search inside DNS queries detail. All data is stored in an SQL database. This tool, can work inside a DNS server or in another server that "captures" bi-directional traffic for a DNS node.

John Kristoff also proposed an automated incident response system using BIND query logs [10]. This particular system, besides the common statistical analysis, also provides information regarding the kind of consultations operated. All information is available through the Web-based portal. Each security incident can result in port deactivation.

4. Methodology

4.1. Architecture. The architecture of the system that we have developed aims to improve the security, performance, and efficiency of the DNS protocol, removing all unwanted traffic and reinforce the resilience of a Top Level Domain. We propose an architecture comprising an integrated protection of multiple DNS servers, working together with several network sensors that apply live rules to a dedicated firewall, acting as a traffic shaping element.

Sensors carefully located in the network monitor all the traffic going to the DNS infrastructure identify potentially harmful traffic using an algorithm that we have developed and tested and use this information to isolate traffic that has been identified to have security threats.

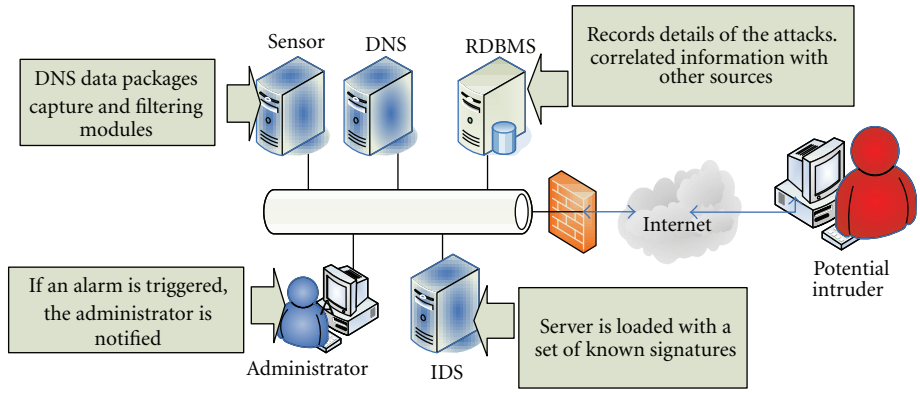

Figure 1: Diagram of the desired solution.

Several networks sensor monitor different parts of the infrastructure and exchange information related to security attacks. In this way, as shown in Figure 1, it should also be possible to exchange critical security information between the sensors. In addition to an increase in performance, this operation should prevent an attack on a server from a source, identified by another sensor as malicious. This scenario is relevant since some kinds of attacks are directed to several components of the DNS infrastructure.

4.2. Heuristic. One of the crucial parts of our work is the algorithm to identify traffic harmful to the DNS. In order to implement the stated hypothesis in the architecture and keep the DNS protocol as efficient as possible, it is necessary to apply a heuristic, which in real time evaluates all the information collected from different sources and applies convenient weights to each component and act accordingly.

The components that we have chosen to have impact in the security incidents of DNS are the number of occurrences, analysis of type of queries been made, the amount of time between occurrences, the number of probes affected, and information reported from intrusion detection systems.

Our system uses the following formula to evaluate a parameter that measures the likelihood of the occurrence of a security incident:

$$f(x) = O \cdot 0,2 + C \cdot 0,2 + G \cdot 0,15 + N \cdot 0,25 + I \cdot 0,20. \quad (1)$$

Are factors considered in applying this formula as the following.

(i) Occurrences (O)—represents the number of times (instances) that have given source was blocked, so that the distributed then depicted in Table 1.

(ii) Analysis (C)—real-time evaluation of the deviation of the values recorded in relation to the average observed statistics, based on the criteria and weights identified below in Table 2.

Note that the estimates apply the moving average, for the determination of reference values, given the ongoing development of data collected.

Table 1: Contribution of the number of occurrences of a source in malicious heuristic.

Occurrences	Weight
1	25%
2	50%
3	75%
4 or more	100%

Table 2: Contribution of events typified a potentially malicious source given in heuristic.

Event	Weight
Entire zone transfer attempt (AXFR)	100%
Partial transfer zone attempt (IXFR)	50%
Incorrect query volume, 50 to 75% on average per source	75%
Incorrect query volume exceeding 75%	100%
Query volume, up 50%, the average number of access by origin	50%

Table 3: Weight of different time between each occurrence.

Time	Weight
Less than 1 Minute	100%
Less than 1 Hour	75%
Less than 1 Day	50%
Less than 1 Week	25%

(iii) Time between occurrences (G)—time since last occurrence of a given source, distributed with the weights associated to the times below are obeisant, as observed in Table 3.

(iv) Incidence (N)—number of probes that report blocks in the same source.

For the calculation, we observed expression:

$$\frac{1}{\#\text{Total\_Sensors} - \#\text{Sensors\_Attacked}}. \quad (2)$$

Table 4: Interconnection with temporal data gathered from intrusion detection systems.

Metric: common vulnerability scoring system (CVSS)	Weight
Low level	34%
Middle level	67%
High level	100%

(v) Intrusion detection systems (I)—we considered the use of the Snort platform, being free to use, and gather a large number of notarized signatures of security incidents relating to the DNS service, as shown in Table 4.

For the activation of a rule in Firewall occurs will require the following.

(1) The formula shown above take values equal to or greater than 0.25.

(2) The combination of two or more criteria of the formula.

Exception: when receiving information from all the other sensors, in which case a single criteria is sufficient.

(3) It respected the existing White List in the repository, allowing considered privileged sources that are not blocked.

In this way we avoid compromising the Internet service, considering the key role played by DNS, the White List protects key addresses from being blocked in case of false positives events. This list is created from a record of trusted sources, allowing all addresses listed here to be protected from being added to the Firewall rules.

One example is the list of internal addresses, and the DNS servers of ISPs.

Instead, for the removal of a rule in the firewall will need to occur simultaneously on the following assumptions.

(1) Exceeded the quarantine period, based on the parameters in use.

(2) The expression of activation (heuristic) does not (still) check the referenced source.

5. Proposed Solution

5.1. Diagram. As shown in Figure 2, this solution is based on a network of sensor engines that analyze all traffic flowing into the DNS server in the form of valid or invalid queries, process the information received from other probes, and issue restrictions for specific network addresses. In case an abnormal behavior is detected or there is suspicious behavior from a certain network address, it will be blocked in the firewall and the other probes notified so they can act accordingly. The system can also calculate the response time for each operation to evaluate the performance of the server.

For each rule inserted in the sensor firewall, there will be a period of quarantine and, at the end of this time, the sensor will evaluate the behavior of that source, to decide what is needed to remove or keep that rule enabled, as shown in Figure 2.

5.2. Network Data Flow. According to our design, all data that flows through the probe heading for the DNS server is treated according to a standard set of global firewall rules, followed by specific local rules regarding to the addresses that are being blocked in real time. The queries are then delivered to the parser to be analyzed and stored in the RDBMS. At the top is the system of alarms and the Web portal.

All information collected is stored in a database implemented in MySQL [11]. Taking into consideration the need to optimize the performance of the queries and to reduce the volume of information stored, the data is divided into a number of different tables.

The conversion of the IP address of source and destination (DNS server) into an integer format has allowed for much more efficient data storage, and significant improvements in the overall performance of the solution.

The information regarding all queries made is stored daily into a log, and kept available during the next 30 days.

Two tables containing the set of rules that are dynamically applied—add or removed, based on situations that have been triggered—control the correct operation of the firewall. For auditing purposes every action is registered.

The information required for auditing and statistical tasks never expires.

5.3. Statistical Analysis and Performance Evaluation. The statistical information collected and stored in the database has a significant amount of detail. It is possible, for example, to calculate, for each sensor, the evolution of queries per unit of time (hour, day, etc.) badly formatted requests, DNS queries of rare types, and determine the sources that produce the larger number of consultations. It is also possible to see the standard deviation of a given measure so we can relate it to that is seen with the other hits [12].

The performance of the DNS protocol responses is permanently measured, regarding the response time per request. Data is constantly registered and an alarm is raised in case normal response times are exceeded.

6. Case Study

Our proposal have been under development since September 2006 at FCCN—who has the responsibility to manage, register and maintain the domains under the .PT TLD.

At present time, as shown in Figure 3, there are two sensors running attached to the DNS servers (one at the primary DNS and another working together with a secondary DNS server).

The network analyzer is tshark [13], and the firewall used is IPFilter [14]. The real time parser was programmed in Java, collecting the information received from the tshark. The Web server is running Apache with PHP.

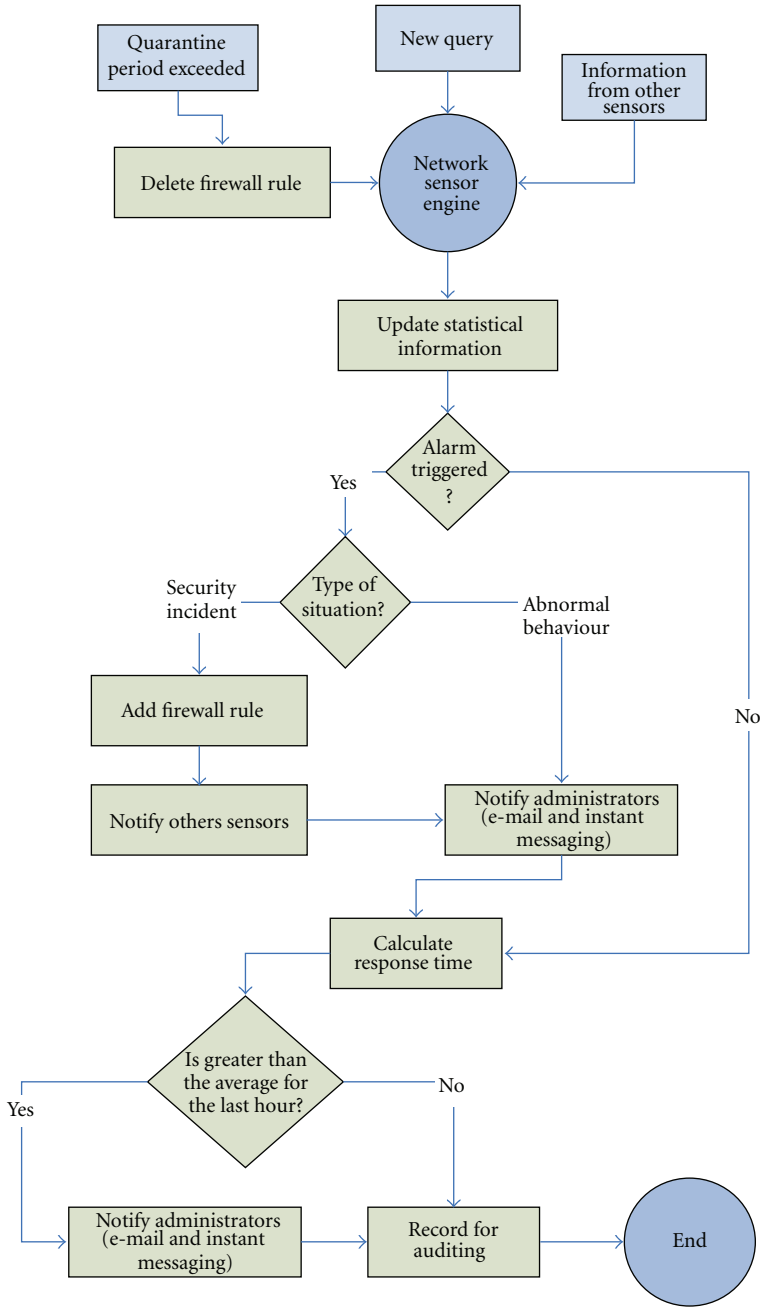

Figure 2: Block Diagram of proposed solution.

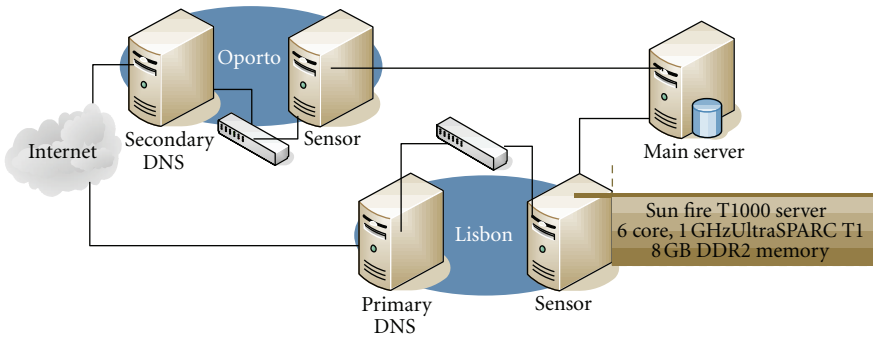

Figure 3: Working sensors coupled with DNS servers.

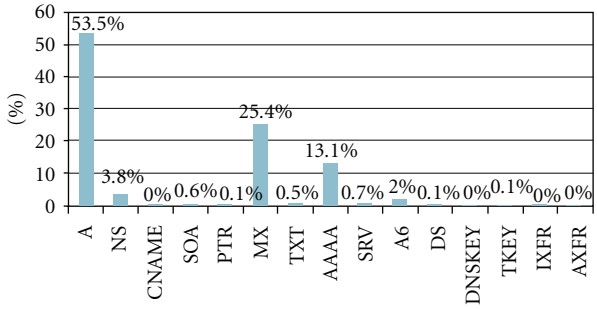

Figure 4: Statistical analysis by type of records accessed.

Regarding the Xmpp server [15], we choose the Jive messenger platform.

All modules are integrated together.

The entire sensor solution, as described above, as well as the web platform we developed went online on the 1st of January 2007, and the data from the various agents was collected from the 10th of May 2008 till now.

7. Results

We present here the results of the last 12 months of data collection (between 1st of May 2010 and 31st May 2011). The Average number of requests to the primary DNS server is up to 14,459,356 per day (167 per sec.).

The performance of the data analysis program is above 1240 requests processed per sec. (filtered, validated, and inserted in the database).

Using the data collected by the sensors, during this time period, we were able to do the following.

(i) Collect useful statistical information. For example, daily statistics by type of DNS protocol registers accessed (Figure 4).

(ii) Detect examples of abnormal use (that are not security incidents). For example we were able to detect that a given IP was using the primary .PT DNS server as location resolver.

The number of queries made was excessive when compared with the average value per source, reaching values close to some Internet Service Providers that operate under the .PT domain.

(iii) Detect situations of abuse, including denial of service attacks, with the execution of massive queries. In last 12 months of analysis there are 17 DOS attacks triggered.

They were instantly blocked, and addresses placed in quarantine (Table 5).

(iv) Improve DNS protocol performance repairing situations of inefficient parameterization of the DNS server.

On the DNS server side, considering the capacity of the probe to determine the processing time for each consultation, it is possible to detect cases of excessive delay, which was later confirmed to coincide with of moments of zone update.

Table 5: Examples when the sensor-detected situations that required the Firewall rules to change.

Source address	Date/time	Operation	Sensor
xx.xx.200.35	2011-07-12 01:03:12	Add rule	xx.xx.21.62
xx.xx.17.212	2011-07-12 03:25:19	Remove rule	xx.xx.32.63
xx.xx.117.51	2011-07-13 01:23:17	Add rule	xx.xx.21.62
xx.xx.94.139	2011-07-13 02:27:11	Add rule	xx.xx.31.63
xx.xx.13.231	2011-07-14 03:42:52	Remove rule	xx.xx.21.62

Table 6: Original decision versus midterm review.

Original decision (intrusion detection system)	Mid-term review
True positive	Correct
False Positive	Type I error
True Negative	Correct
False Negative	Type II error

Considering the daily progress of DNS queries, before and after applying shaping heuristic to the protocol we obtain an improvement between values of 5.3% (minimum) and 19.4% (maximum).

8. Reducing False Positives and False Negatives

For the treatment of false positives and negatives, it is important to evaluate the results obtained with each of the components in separate—from intrusion detection solution and the solution proposed in the hypothesis presented here, and after evaluating the results that accrue from the application of heuristic indicated above.

It is inevitable given the occurrence of false positives—type I error (when events are identified as security incidents that do not correspond to real situations), as well as false negatives, Type II error (when there are security incidents that are not detected by the solution in use).

And such an occurrence is the case both in traditional IDS solution, in this case the SNORT, as in the methodology implemented here in the form of prototype with the DNS server at .PT.

The way to mitigate such situations is to combine the values obtained by both mechanisms, thus seeking an end result, as close to reality as possible, as identified in Table 6.

Operated with the observation by the data obtained, for a period of 12 months, we observe that a reduction is achieved in 21% of false positives identified by the IDS solution when operated in isolation. The identification of type II errors is now possible in 8% of cases, which previously went unnoticed.

Table 7: Decision with versus without the methodology.

Original decision	Decision without the methodology	Decision with the methodology
True positive	Accept	Correct
	Reject	Type II error
False Positive	Accept	Type I error
	Reject	Correct
True negative	Accept	Correct
	Reject	Type I error
False negative	Accept	Type II error
	Reject	Correct

Given the mutual dependence on heuristic established between the two data sources—IDS and the proposed platform—we cannot verify the occurrence of false positives as a result of implementation of the method proposed here. Is it possible, however, as shown in Table 7, the occurrence of false negatives due to situations considered in the IDS solution, can be canceled later in the final assessment.

In any case given the relevance of the DNS service on the proper operation of the Internet, and the disorder caused by a blockage of a given origin, and there is no absolute certainty that this is a clear case of a security incident, makes it preferable in this particular case, the occurrence of a controlled number of false negatives.

9. Conclusion and Future Work

The solution presented here builds upon the existing solutions that collect statistical information regarding DNS services, by adding the ability to detect and control security incidents in real time. It also adds the advantage of operating in a distributed way, allowing the exchange of information between cooperating probes, and the reinforcement of its own security, even before it is threatened.

Currently, the solution presented does not allow the processing of addresses in the IPv6 format. The technical aspects that led to this situation are linked to the need to optimize the performance of the data recorder application making it possible to store the data from all consultations. Nevertheless, all queries made to IPv6 addresses are contained in this solution (AAAA types).

We are also working on extending the data correlation capabilities of the system by adding information collected from other sources (intrusion detection systems for instance). We anticipate that this could be a valuable approach to reduce considerably the number of false positives and negatives [16].

References

[1] P. Vixie, "DNS complexity," *Queue*, vol. 5, no. 3, pp. 24–29, 2007.
[2] D. Wessels, "A recent DNS survey," DNS-OARC, 2004.
[3] D. Piscitello, "Conficker summary and review," Tech. Rep., ICANN, May 2010.
[4] "SQLDNS website," http://home.tiscali.cz/~cz210552/sqldns.html.
[5] "BIND website," http://www.isc.org/products/BIND.
[6] B. Zdrnja, "Security monitoring of DNS traffic," May 2006.
[7] P. Vixie and D. Wessels, "DNSCAP—DNS traffic capture utility," in *Proceedings of the The Cooperative Association for Internet Data Analysis Workshop*, July 2007.
[8] D. Wessels, "Whats New with DSC," DNS-OARC, 2007.
[9] "Tcpdump website," Lawrence Berkeley National Laboratory, http://www.tcpdump.org/.
[10] John Kristoff, "An automated incident response system using BIND query logs," June 2006.
[11] "MySQL website," Open Source Database, http://www.mysql.com/.
[12] J. Afonso, E. Monteiro, and V. Costa, "Development of an integrated solution for intrusion detection: a model based on data correlation," in *Proceedings of the International Conference on Networking and Services (ICNS'06)*, Silicon Valley, Calif, USA, July 2006.
[13] "Tshark website," The Wireshark Network Analyzer, http://www.wireshark.org/.
[14] IP FILTER – TCP/IP Firewall/NAT Software, http://coombs.anu.edu.au/~avalon/.
[15] P. Saint-Andre, Ed., "Extensible Messaging and Presence Protocol (XMPP): core," RFC 3920, 2004.
[16] J. Afonso and P. Veiga, "Protecting the DNS infrastructure of a top level domain: real-time monitoring with network sensors," in *Proceedings of the 4th IEEE International Workshop on Wireless and Sensor Networks Security (WSNS '08)*, Atlanta, Ga, USA, 2008.