Web Server Security on Open Source Environments

Dimitrios X. Gkoutzelis[1] and Manolis S. Sardis[2]

[1] University of the Aegean, Samos, Greece
icsd04126@icsd.aegean.gr
[2] National Technical University of Athens - NTUA, Athens, Greece
sardis@telecom.ntua.gr

Abstract. Administering critical resources has never been more difficult that it is today. In a changing world of software innovation where major changes occur on a daily basis, it is crucial for the webmasters and server administrators to shield their data against an unknown arsenal of attacks in the hands of their attackers. Up until now this kind of defense was a privilege of the few, outbudgeted and low cost solutions let the defender vulnerable to the uprising of innovating attacking methods. Luckily, the digital revolution of the past decade left its mark, changing the way we face security forever: open source infrastructure today covers all the prerequisites for a secure web environment in a way we could never imagine fifteen years ago. Online security of large corporations, military and government bodies is more and more handled by open source application thus driving the technological trend of the 21st century in adopting open solutions to E-Commerce and privacy issues. This paper describes substantial security precautions in facing privacy and authentication issues in a totally open source web environment. Our goal is to state and face the most known problems in data handling and consequently propose the most appealing techniques to face these challenges through an open solution.

Keywords: Open Source, Security Practices, Authentication Management.

1 Introduction

In today's electronic world thousands of application layer software is being designed and shared throughout the Open Source (OS) communities. Utilized solely by volunteering work which produces pioneering outcome when combined with world wide projects sharing resources and human personnel, projects in this area often win the trust of the vast majorities of website developers and website owners with its easy-to-use and free of charge benefits [1], [2], [38]. But do those solutions come with a security-cost or not [3], [36]?

This work examines the ways in which the new trend in global programming produces solutions from non commercial products in an absolutely secure and safe way - compared to close source. It is common logic to assume that when it comes to data security and handling sensitive transactions it is not always a clever idea to have your code open, or even worse, shared with millions of people. Today, people are able to get educated, trained and specialized in certain programming areas of their likes without the

A.B. Sideridis and Ch. Z. Patrikakis (Eds.): e-Democracy 2009, LNICST 26, pp. 236–247, 2010.

need to pay for extra education fees or joining a closed group of professionals. Writing open source software has enabled thousands of people to look closely at professionally-written code, learn the ropes of advanced software engineering methods and in that way excel the way we read, write and understand code to our own advantage. It would certainly not be an exaggeration to state that the open source movement has influenced the security of modern networks and large computing compounds. One of the most well known examples is the LAMP (Linux, Apache, MySQL and PHP) architecture used to minimize security risks and provide a totally open source solution to handling security on your web server [4]. LAMP software consists of the main tools security professionals most commonly use for business systems.

A recent survey [5] performed by Netcraft, a data collection and analysis of hosting companies organization, showed that 46,35% of the servers on the market use Apache as their web server software leaving the second position to Microsoft with 29,47%. What is more worth mentioning is that on the survey on the 1 million busiest servers in the world that percentage reaches 66,6% leaving IIS (Internet Information Server) with a 18,6% (Fig. 1).

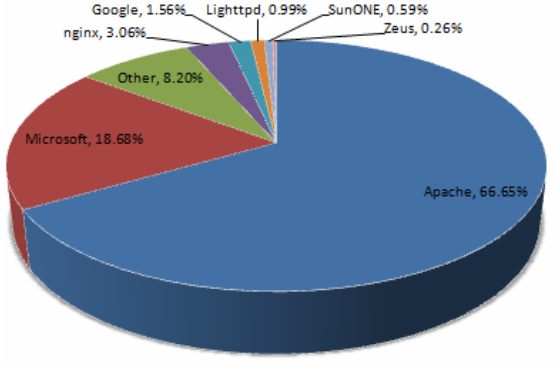

Fig. 1. Server Share amongst the Million Busiest Sites (March 2009)

This paper analyzes separately each of the main problems a server administrator has to address, when facing security issues and special care will be given on the meaning of open source tools to aid that goal. The paper is composed of three parts. In the first we present the basic ground of server security measures that every administrator should start with. The second part contains a new approach in how we face security with some advanced practices that have begun to gain administrators trust around the world. In the third and final part, the future work in the field of security is described and the limits that still need to be crossed in order to maximize efficiency.

2 Facing Security Issues on the Front Line

When talking about security over the Internet, from an administrator's point of view, there are really a lot to talk about. In this section we analyze the most crucial techniques

to be established by a web server administrator to protect the box from the most common and popular attacks.

2.1 File Permissions

Every folder or directory our website contains has its own permissions. Those permissions are defining who is allowed to do what on our files. On Unix-based operating systems there are three kind of permissions: read-write-execute and three kind of owners: owner-group=other. Permissions are set by using a three-digit number: for example with 000 being the most restricting as it takes away all rights on the file and with 777 being the most liberal as it provides all permissions to all groups. The first digit represents what is allowed to be done by the owner of the file, the second represents what the rest authorized users can do and the third everybody else's permissions on it. The command with which we change or set the permissions of a file is chmod. To keep balance between usability and security all directories should be set to 755 and all files to 644 unless a certain file has specific configuration that demand otherwise [7].

It is a spread and good practice to keep the files and directories non-writable in the extent that it is possible for the normal function and to make them writable only in the need of significant content-changing such as configuration files or where manual editing is the only choice available. What should drive our actions is the Least Privilege Approach: if someone has no reason to have access to a specific resource or to able to perform a certain action on it then he should be banned from doing, or in a more polite rephrase "one should only be able to do what he is needed to do and nothing else". All the typical recommendations, to set any files on status 777, are given by people who have no concern about security or at least it is not their first priority and should be avoided at all costs. One other thing to be considered is to avoid setting as server-writable files under the document root. A large percentage of the attacks occurring in small websites are caused because the web server could write under the document root, so all an attacker had to do was to upload a php script which he called through his/her browser.

2.2 Shielding Your Server with SSL

First and foremost when talking about building something secure is to keep the administrator's keys to a safe place. If fallen to the wrong hands then all efforts about security go down the gutter. All user data should be handled exclusively through SSL (TLS) [8], a technology that is used to encrypt the data over the channel you sent it on Internet to avoid eavesdropping, tampering or message forging and your user's data to end up on the wrong hands. TLS runs on many layers below protocol layer such as:

- HTTP (Hypertext Transfer Protocol): which could be used for application users' login through HTTPS (http secure).
- FTP (File Transfer Protocol): in order to secure file transfer procedures through SFTP mechanisms.
- SMTP (Simple Mail Transfer Protocol): for your email accounts protection.

Today all the algorithms and libraries required for those cryptographic protocols to work are open and unified under the OpenSSL [6] project which is an open source implementation of the SSL and TLS protocols. The core library implements the basic cryptographic functions and provides various utility functions.

2.2.1 How It Works

The client and the server to be contacted negotiate a connection by using a handshaking procedure (Fig. 2). In the meaning of handshaking we mean the agreement on using certain parameters as common in order to establish the connection's security.

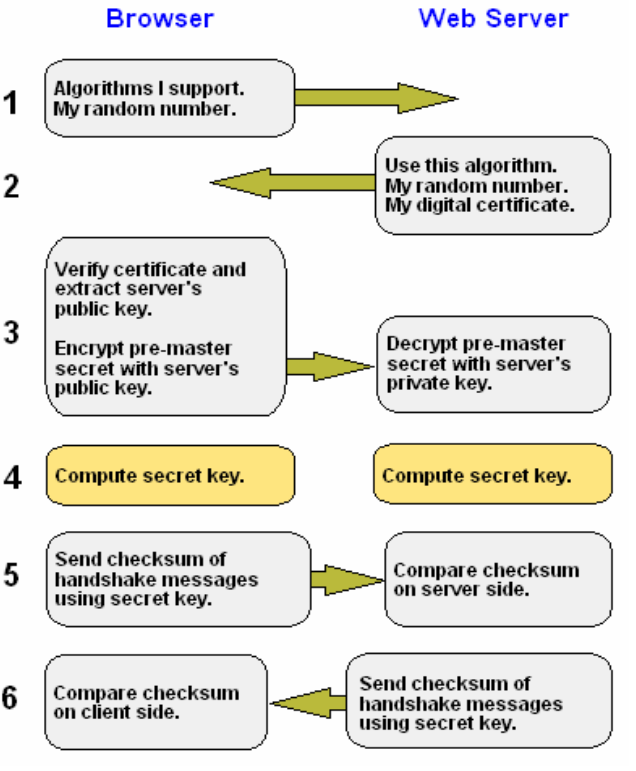

Fig. 2. Description of the TLS handshaking

The procedure is initiated by the client's request to connect to a TLS-enabled web server after it presents it to a list of supported ciphers and hash functions. In turn, the server picks the strongest ones that it also supports and sends a message to the client about his decisions in the form of a digital certificate which usually holds valuable information such as the server's name, trusted certificate authority, and the server's public key.

In cases of strict security measures, the client will contact that authority to verify the validity of the server's before it establishes a connection. After the validation, the

client encrypts a random number with the server's public key and sends it to the server to be decrypted with its private key.

2.2.2 Advantages of Using SSL

The success of SSL in e-commerce is a testament to the advantages of the technology. Using the common web browser as the primary client software, simplifies support requirements by eliminating the need for additional software applications. Authentication of the target SSL Web server is transparent to the end user and is fairly reliable.

Encryption algorithms have evolved with the technology to provide a high level of security with the availability of 256-bit and higher keys. Authentication with SSL is achieved with the identification of the server by the client via a Digital Certificate that is issued and signed by a Certificate Authority (CA) and stored on the server. Identification and authentication of the client accessing the server, although possible, is not practical for the purposes of e-commerce since the vast percentage of clients do not have Digital Certificates that are signed by a registered CA. Without a certificate signed by a CA, reliable identification of the client to the server is not possible. This can lead to a situation where an anonymous client on the Internet can connect to the SSL server, establish an encrypted session and then use this session as a secure channel for attacking the specific Web server associated with the session. The encrypted SSL connection has traditionally prohibited network security, or management personnel from inspecting the contents of the session prior to its termination at the SSL concentrator, or the Web Server that terminates the SSL session.

2.3 Logging Critical Information

If the server is unrecoverable, remote logs allow to see what happened prior to the crash, even without the system running. If the crash is related to an intrusion, any information that can describe how the system was compromised can help determine the cause of the problem. Because of the importance of the log files to the administrator's duties, the location of those files should not be adjacent to the logged system. A very common approach is to use a remote logging server with tight security, where the logs can be kept safe from crashes or attacks to our system.

One of the main logging tools on every Linux distribution is *syslogd*, which is very easy to configure and use. Different software stores its logs to different places, for example with Apache [9] you set where you store your logs through *httpd.conf* and if we use Red Hat [10] the *httpd.conf* is usually stored in */etc/httpd/conf/httpd.conf*.

There are countless tools and applications out there today that enable detailed and user-defined logging [11] of all actions and data that are characterized as important for the system. A good system is one that keeps track of all actions performed so as to revive in case of a disaster [12].

2.4 Password Management and Backups

2.4.1 Passwords

Due to cost and compatibility with legacy systems, the most popular form of user authentication continues to be a secret password. Passwords are simply secret words, or at best secret phrases. They can be compromised in many ways:

- Users may write them down or share them, so that they are no longer really secret.
- Passwords can be guessed, either by a person or a program designed to quickly try many possibilities.
- Passwords may be transmitted over a network either in plaintext, or encoded in a way which can be readily converted back to plaintext.
- Passwords may be stored on a workstation, server or backup media in plaintext, or encoded in a way which can be readily converted back to plaintext.

Users in a large organization frequently have many passwords, each protecting their access to a different computer system. Users have some basic limitations, which limit what can be done in the context of secure password management. One of the primary weaknesses of passwords is that they may be guessed. While a human may give up after trying guessing ten or a hundred possible passwords, software will happily try millions of combinations. To combat password guessing attack, users should pick hard-to-guess passwords. One way to do this is to ensure that the set of all possible passwords is too large to search thoroughly, and then to eliminate probably guesses.

To limit the usefulness of passwords that have been compromised, a best practice is to change them regularly.

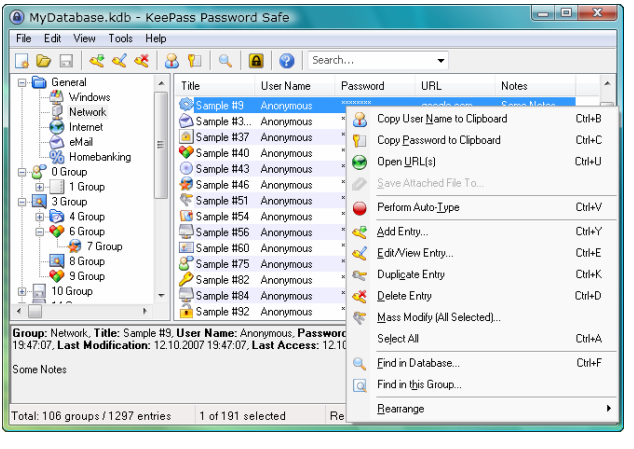

Fig. 3. Open Source Password Manager for multi user environments (KeePass [39])

A common rule on many systems is to force users to change their passwords when they log in, if they have not been changed for an extended period (e.g., 60 or 90 days). In general, users should be required to change their passwords regularly, at most every 90 days, and preferably more frequently. For the same reasons, users should not reuse old passwords, as they may already have been compromised. Many systems support this (Fig. 3) by recording some representation of old passwords, and ensuring that users cannot change their password back to a previously used value.

2.4.2 Backups

In an information environment, an organization's success is tightly coupled to its ability to store and manage information. Storage systems provide a critical part of an organization's network infrastructure. With the amount of data growing at an incredible rate, your storage strategy must keep pace. In designing a storage strategy for your organization, you must select the right technology for your primary storage system, implement solid backup procedures and ensure ongoing management of the system.

When you start thinking of backing up the data, first thing comes to be considered is what data to backup and how much storage is available for the backup to take place. This depends on what data is important. So before backing up your data, the capacity of your data places an important role, identifying the capacity of the backup medium to the quantity of data you propose to backup is very important. An optimized solution is to maintain storage devices able to store a mirror copy of your server's data so as not to make any cutbacks on the amount to be rescued in case of a disaster.

2.5 VPN Usage for Extra Security

A Virtual Private network (VPN) is a network that uses a public network (usually the Internet) to connect remote sites or users together. Instead of using a dedicated, real-world connection such as leased line, a VPN uses "virtual" connections routed through the Internet from the company's private network to the remote site or employee. A well-designed VPN uses several methods for keeping your connection and data secure (Fig. 4):

- Firewalls
- Encryption
- IPSec
- AAA Server

A firewall provides a strong barrier between a private network and the Internet. We can set firewalls to restrict the number of open ports, what type of packets are passing through and which protocols are allowed through. We should already have a good firewall in place before we implement a VPN, but a firewall can also be used to terminate the VPN sessions.

Encryption is the process of taking all the data that one computer is sending to another and encoding it into a form that only the other computer will be able to decode. Most computer encryption systems belong in one of two categories:

- Symmetric-key encryption [14] each computer has a secret key (code) that it can use to encrypt a packet of information before it is sent over the network to another computer.
- Public-key encryption [15] uses a combination of a private key and a public key. The private key is known only to our computer, while the public key is given by our computer to any computer that wants to communicate securely.

Internet Protocol Security Protocol (IPSec) provides enhanced security features such as better encryption algorithms and more comprehensive authentication. IPSec has two encryption modes: tunnel and transport. Tunnel encrypts the header and the payload of each packet while transport only encrypts the payload.

AAA (authentication, authorization and accounting) servers are used for more secure access in a remote-access VPN environment. When a request to establish a session comes in from a dial-up client, the request is proxied to the AAA server. AAA then checks the authentication, the authorization and the accounting. By effectively administering the server through a VPN network and granting access to the users only through it, we maximize security precautions. As it adds an extra layer of protection on the server and the data, a VPN installation deters hackers from performing an assault on the network. It is today a very well known practice to use VPN as an extra layer of security in many large corporations [35].

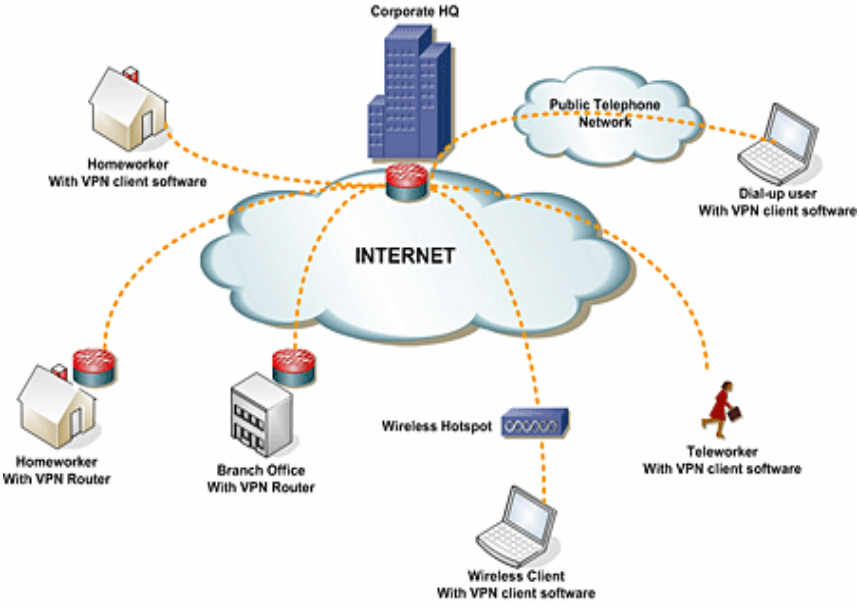

Fig. 4. Representation of a VPN network [13]

3 Pioneering Security Mechanisms

3.1 Building Your Own Honey Pots

Sophisticated hackers that have the means and knowledge to hack into your system will eventually succeed in their task. That is the reason why preventing unauthorized access is not adequate any more. Defenders have to be as smart and pioneering as their intruders in order to successfully face a serious hacking attempt. Honeypots [16] can help prevent attacks in several ways. The first is against automated attacks, such as worms or auto-rooters. These attacks are based on tools that randomly scan entire networks looking for vulnerable systems. If vulnerable systems are found, these automated tools will then attack and take over the system (with worms self-replicating, copying themselves to the victim). One way that honeypots can help defend against such attacks is slowing their scanning down, potentially even stopping them. Called sticky honeypots, these

solutions monitor unused IP space. When probed by such scanning activity, these honeypots interact with and slow the attacker down.

Honey pots [17] appeared on the security scene as the means to cover the hole in creating an "ambush" to the attacking parties. It is a very complex security mechanism designed to act as a trap for the intruders luring them to exploit a resource intentionally kept unprotected giving time to the defender to collect information on his attackers. Its diversity in usage is what makes it unique in its purpose: a honey pot can either act as distraction from more valuable machines on a network, serve as an early warning mechanism about unauthorized access into the server or as a research mechanism that observes the intruder and collects valuable information about his steps and system information. There are two types of Honeypots:

- *Production Honeypots* are easy to use, capture only limited information, and are used primarily by companies or corporations; Production honeypots are placed inside the production network with other production servers by organization to improve their overall state of security. Normally, production honeypots are low-interaction honeypots which are easier to deploy. They give less information about the attacks or attackers than research honeypots do. The purpose of a production honeypot is to help mitigate risk in an organization. The honeypot adds value [18] to the security measures of an organization.
- *Research Honeypots* are run by a volunteer, non-profit research organization or an educational institution to gather information about the motives and tactics of the Blackhat [19] community targeting different networks. Research honeypots are complex to deploy and maintain, capture extensive information, and are used primarily by research, military, or government organizations.

3.2 Password-Less Login Using RSA Keys

There eventually comes a point in technologies where current mechanisms are outdated and surpassed. In the presence of such needs new ideas emerge? Password orientated authentication has been for ages the most popular of all login procedures but the number of hacking tools for intercepting or cracking those passwords have multiplied as well [20]. The new trend in authentication processes is password less logins to the system using public key cryptography.

With the use of OpenSSH [21] which fully supports the RSA [22] algorithm, remote users can authenticate themselves to the server using its public key to encrypt data and their own private key to decrypt it. With the use of a secret master password to protect our private key we maximize the security measures of loss or interception of our data. Even if our master password gets stolen the intruder will still not have our private key and same reasons hold if he has only our private key without the master password. SSH clients like Putty [23] need the master password to authenticate that we are the owner of the private key. So in effect, combining the two technologies leads to the best results.

3.3 Port-Knocking

Port scanning [25] is one of the main tools in the hands of the attacker that troubles the security community trying to find a permanent solution. Such a solution is known

as *port knocking*, [26] a very sophisticated mechanism to prevent attackers from discovering the open ports of your server.

The procedure is a very simple yet quite intelligent packet sending sequence which imitates the known "Open Sesame" story from the fairy tales. In order for the remote host to be able to connect to a specific port he must first send packets to pre-specified closed ports in a predefined sequence so as to inform the Firewall that it must open the port. Unless the correct closed ports have been "pinged" the wanted port will not appear open. That way even ports that are really open will appear closed to potential port scanning attacks wishing to exploit services of the web server [37]. This method is used to isolate critical resources from the Internet and grant access and awareness of existence only to those really intended to. The best way to keep a door locked is if people do not know how to find the door at the first place, except of course the owners.

3.4 File System Encryption

Sometimes there is really nothing we can do and even the best administrator of the world can not save a violated private property that had its hard drives stolen from their server. Physical security is a big chapter in the Security community but one popular and widespread technique is *file system encryption* [27]. By encrypting the data kept on the web server's hard drivers we prevent unauthorized access in case of theft from the datacenter. Documents of great corporate, military or financial importance often worry enterprise and government officials about their fate should they fall on the wrong hands. There are a variety of tools for open source usage in Linux file system encryption [28] that fit the needs of every IT professional.

4 Conclusions and Future Work

Undoubtedly, open source ideology has contributed to the growth and evolution of computer software security [29] in every aspect. Its ability to reach inside thousands of programmer's eyes before hitting the market makes the product solid against a series of attacks the developers couldn't humanly have noticed themselves. Vincent Rijmen [30], one of the inventors of the known AES algorithm, which today is the basis of Internet security claims that the open source (OS) culture will gradually improve security and make vulnerabilities easier to spot:

"*Not only because more people can look at it, but, more importantly, because the model forces people to write more clear code, and to adhere to standards. This in turn facilitates security review*" [40].

Although making your code transparent certainly improves your readiness to respond to programmer's suggestions and bug fixes this alone is not enough as it can "*lull people into a false sense of security*" [31]. Among the biggest opponents of open source ideology there is the argument that the construction process of code is absent in large open source projects [32] whereas proven in practice those projects are lead by one or few version with owners with reputation at stake.

Security professionals agree that sharing a code and exposing it to the eyes of possible attackers actually makes it safer. Safer to spot, track and fix. When the question

of who might spend hours to figure out bugs in someone else's code the answer is simple: people who use it often, people in the enterprise business that depend their data and money on it and also volunteers from anywhere simply wanting to help. Closed Source software that keeps the code secret away from the eyes of the attacker does not guarantee that is safer by no means [33]. *Security through transparency* is the new trend to substitute *Security through obscurity*.

There are dozens of security techniques pending exploration; new mechanisms introduced by OS communities and IT professionals every day and all part of volunteering open source code writing from the people for the people. The importance and magnitude of this trend can be measured in the scale that governments and large organizations turn their back on the unaffordable, expensive and badly maintained closed source applications and adopt open source platforms and infrastructures [34] for their critical data handling operations. The global community is now mature enough to proceed to a new model of Systems and Information Security based on open procedures and no one can foresee what can be achieved in the future of computer software. Business, E-Commerce, Education, Defense, Science and all important aspects of our everyday encounters will evolve around the ability to share and improve each other's life through open and participatory procedures.

References

1. Lawton, G.: Open source security: opportunity or oxymoron? Computer 35(3), 18–21 (2002)
2. Spinellis, D., Szyperski, C.: How is open source affecting software development? IEEE Software 21(1), 28–33 (2004)
3. Witten, B., Landwehr, C., Caloyannides, M.: Does open source improve system security? IEEE Software 18(5), 57–61 (2001)
4. Shankar, K.S.D., Kurth, H.: Certifying open source - the Linux experience. IEEE Security & Privacy 2(6), 28–33 (2004)
5. Net Craft Secure Server Survey -, Web Server Survey (March 2009), http://news.netcraft.com/archives/2009/03/15/march_2009_web_server_survey.html
6. OpenSSL Project, http://www.openssl.org/
7. World writable/tmp, http://seclists.org/bugtraq/1998/Jul/0119.html
8. Transport Layer security, http://en.wikipedia.org/wiki/Transport_Layer_Security
9. Apache Software Foundation, http://www.apache.org/
10. Red Hat Entreprise, http://www.redhat.com/
11. Linux log Analyzers, http://www.linux.org/apps/all/Administration/Log_Analyzers.html
12. Schroder, C.: Enhance Security with a Linux Logging Server, http://www.enterprisenetworkingplanet.com/netos/article.php/3521481
13. VPN Image, http://www.lanos.co.uk/main/images/stories/diagram/vpn.gif%20
14. Toolbox for Information Technology, Symmetric key Encryption, http://it.toolbox.com/wiki/index.php/Symmetric_Key_Encryption

15. Online Encyclopedia: Wikipedia, Public-key cryptography, http://en.wikipedia.org/wiki/Public-key_cryptography
16. Tian, Z.-H., Fang, B.-X., Yun, X.-C.: An architecture for intrusion detection using honey pot, November 2-5, vol. 4, pp. 2096–2100 (2003), doi:10.1109/ICMLC.2003.1259851
17. Honey pots Intrusion Detection, http://www.honeypots.net/
18. Spitzner, L.: Honey pots: Definitions and Value of Honey pots (May 2003), http://www.tracking-hackers.com/papers/honeypots.html
19. Black Hat Homepage, http://www.blackhat.com/
20. Online Encyclopedia: Wikipedia, Password Cracking, http://en.wikipedia.org/wiki/Password_cracking
21. OpenSSH Project, http://www.openssh.com/
22. RSA Laboratories, http://www.rsa.com/rsalabs/node.asp?id=2146
23. Putty Homepage, http://www.chiark.greenend.org.uk/~sgtatham/putty/
24. OpenSSH Passwordless Connections, http://wiki.e-shell.org/OpenSSHPasswordlessConnectionsTheQuickWay#rsa
25. Introduction to Port Scanning, http://netsecurity.about.com/cs/hackertools/a/aa121303.htm
26. Linux Journal, Port Knocking, http://www.linuxjournal.com/article/6811
27. Linux journal, Encrypt your file system, http://www.linuxjournal.com/article/7743
28. Linux.com, Enhance Security with file encryption tools, http://www.linux.com/feature/59932
29. Secure Programming for Linux and Unix, 'Is Open Source Good for Security? http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/open-source-security.html
30. Biography of Vincent Rijmen, http://www.nist.gov/public_affairs/releases/biovince.htm
31. Viega, J.: The Myth of Open Source Security, http://it.slashdot.org/article.pl?sid=02/02/15/1846214
32. Schneider, F.B.: Open Source in Security: Visiting the Bizarre, May 14-17, pp. 126–127. IEEE CNF (2000)
33. The Risks of Closed Source security, http://www.ibiblio.org/oswg/oswg-nightly/oswg/en_US.ISO_8859-1/articles/alan-cox/risks/risks-closed-source/risks.html
34. Linux Adoption worldwide, Online Encyclopedia: Wikipedia, http://en.wikipedia.org/wiki/Linux_adoption#Government
35. Khanvilkar, S., Khokhar, A.: Virtual private networks: an overview with performance evaluation. IEEE Communications Magazine 42(10), 146–154 (2004)
36. Hissam, S.A., Plakosh, D., Weinstock, C.: Trust and vulnerability in open source software. IEE Proceedings Software 149(1), 47–51 (2002)
37. Ohmaki, K.: Open source software research activities in AIST towards secure open systems. In: Ohmaki, K. (ed.) Proceedings of 7th IEEE International Symposium on High Assurance Systems Engineering, 2002 High Assurance Systems Engineering, 2002, pp. 37–41 (2002)
38. Sarkinen, J.: An open source(d) controller. In: Telecommunications Energy Conference, 2007. INTELEC 2007, September 30-October 4, pp. 761–768 (2007)
39. KeePass password Safe, http://keepass.info/
40. Rijmen, V.: http://www.linuxsecurity.com/content/view/117552/49/