Cryptolog: A new approach to provide log security for digital forensics

Muhammet BAYKARA[1], Resul DAŞ[1], Gürkan TUNA[2]

[1] Department of Software Engineering, Fırat University, Elazığ, Turkey
[2]Department of Computer Programming, Trakya University, Edirne, Turkey
mbaykara@firat.edu.tr, rdas@firat.edu.tr, gurkantuna@trakya.edu.tr

Abstract: *As security vulnerabilities generated by the developments in information and communication technologies as well as emerging technologies can lead to severe loss in terms of individual and institutional aspects, the importance of information security has been increasing in recent years. Nowadays, digital information is considered as an important asset which must be appropriately evaluated and protected against all forms of unauthorized access, use, disclosure, modification, destruction, or denial. Since information security is more prominent and more important now than ever before, this growing awareness of digital information security has led societies to develop innovative ways of protecting their sensitive information. On the other hand, in today's digital world, keeping sensitive information secure is not as easy as it was in the past. In this regard, it is obvious that for all types of institutions there is a need for security software which provides the necessary security measures and policies for the protection and retrieval of sensitive digital information. To ensure information security, security software must have the ability to make logging of certain events. Through log files, some analysis can be performed to find out what kind of attacks were done by which users and when. In this respect, this study proposes a novel approach of recording traffic flow on the log files stored on a server to determine the changes made by unauthorized people/users on the log records, and this way ensures the security of the log records and contributes to digital forensics processes in terms of accuracy, integrity and confidentiality of the log records.*

Keywords: *Log analysis, log collection, log security, information security, digital forensics.*

1. Introduction

Due to the growing reliance on computers and communications networks, information security has become a must in the "digital information world". In this respect, all types of organizations are trying to cope with this challenge by employing appropriate tools in order to pro-actively establish, monitor and enforce information security measures and thereby limit access to sensitive electronic information. This challenge is sometimes viewed as the challenge of balancing the demands of users versus the need for data confidentiality and integrity and typically is a complex process which needs to be continuously updated and constantly be reviewed.

Basically, there are three main objectives for information security: confidentiality, integrity, and availability of data. Therefore, in the digital era, information security measures must be sufficient to ensure the confidentiality, integrity, availability, accountability, and auditability of important information [1]. In this regard, security software with logging ability must be in use to prevent potential threats and address challenges. In addition to dealing with any potential security threats from external sources, organizations must protect their sensitive data from internal threats too.

Since logging systems play a key role in information security, effective protection plans aimed at securing information systems must take the availability of logging systems into consideration and provide data security tools for generated logs [2]. Accordingly, methods or tools to verify the integrity, confidentiality and availability of the generated logs must be provided.

Logging systems can be classified into two main groups. Logging systems which only generate logs on the servers they have been installed on are in the first group. The second group consists of logging systems which can generate logs regardless of the servers they have been installed on. The first group can effectively use all the available features and thereby can carry out more detailed analysis. On the other hand, the logging systems in the second group are more flexible.

Currently the ability of providing data integrity is the common feature of all logging systems. However ensuring that the generated logs are unaltered is as important as data integrity. For this goal, hashing algorithms and the time-stamping technique is used. All generated logs are hashed

Received on: 18.03.2017
Accepted on: 09.072017

and stamped. A third-party certification authority provides the certificate used in timestamps. In addition to these, another important ability of logging systems is the protection mechanisms to prevent unauthorized access. Even if by times-tamping the hashes of generated logs data integrity can be provided, it is not a flawless solution when internal threats are taken into consideration [3]. When log security is ensured, tools to examine and verify the generated logs are needed. Such tools must only be accessible by authorized administrators and through an authentication system. If all these are provided, logs can be used as evidence in digital forensics.

In this study, the details of a software system which controls whether log files belonging to a specific traffic flow have been changed by unauthorized users, ensuring this way the security of the logs, are explained. The rest of the paper is organized as follows. Related work is given in Section 2. Section 3 explains the details of the proposed system. Finally, Section 4 concludes the paper.

2. Related Work

Digital Forensics is a novel interdisciplinary field, between digital technologies and law, and plays a meaningful role in the perpetual race with intelligent criminals [4]. Accordingly, digital evidence is the base of crime investigation and charge while it is easy to change, modify and even destroy [5, 6]. In this respect, a crucial problem is how to effectively maintain and detect the integrity [7] and authenticity of digital evidence. Therefore, monitoring systems provide worthy and particular records and traces of potential crime behavior and hence play a key role in digital forensics.

According to current legislation, in the US, EU countries and Turkey, Internet Service Providers and hosting companies have to log traffic flows destined to/sourcing from their networks for 1 year or more [8]. In this regard, log security is an essential issue which must be taken into consideration in designing communication architectures. Addressing this issue prevents any consequences resulting from the malpractice of authorized users and the changes made by unauthorized users. Accordingly, standardization efforts for log security have been initiated and are pursued in many countries. One of the proposed standards for log security is the Payment Card Industry Data Security Standard (PCI DSS). Although the PCI DSS is a widely accepted set of policies and procedures to optimize the security of credit, debit and cash card transactions, one of its objectives is related to logging [9]. Moreover, log security related legislations, regulations, laws and standards such as the law number 5651 in Turkey [8], FISMA (Federal Information Security Management Act) [10], HIPAA (Health Insurance Portability and Accountability Act), SOX (Sarbanes-Oxley) [11], and ISO 17799 are in force/use.

An important common feature of all types of security software is the logging ability which creates log files to record the activities carried out in a system in plain text files. For instance web servers record user activities and store them in log files which include fields related to the user requests. IP address, port number, state code, and access time are commonly used fields in this kind of log files [12, 13]. Different from the past when log files were mostly neglected except for urgent situations when system related problems were experienced, nowadays they give important information to system administrators and are efficiently used for detailed investigations. In addition to their uses for various purposes, intrusion detection and prevention systems and digital forensics analysis are the emerging application fields of log files [14].

To be presented as digital forensic evidence, log files need to be admissible, authentic, complete, reliable and believable [15] and the goals of digital forensics such as preservation, identification, extraction, documentation, and interpretation of data must be addressed [16], For this objective, many studies have been performed. In [16] Boeck et al. proposed a technique using a Trusted Platform Module and AMD's Secure Virtual Machine technology. The proposed technique establishes hardware-based trust and protects against manipulation of existing logs. However, the proposed technique requires vendor-specific hardware and may not be suitable for implementing on existing systems. A secure audit log using asymmetric key encryption was proposed by Waters et al. [17]. But it only addresses the problems of storing log files and making them searchable and does not verify whether they have been generated by validated systems or not. In [18] a secure logging scheme was proposed. Although the proposed scheme is fine for secure generation, storage and analyses of logs, it does not provide a verification method to identify if the logs have been generated by authenticated sources, similar to the previous one. A set of mechanisms which utilize a database management system (DBMS) and hashes to account for manipulation were proposed in [19]. On the other hand, the proposed mechanisms do not take verification of genuineness and authenticity into consideration. Different from the previous researches, current approaches mostly focus on securing logs and protecting them from unauthorized manipulations [16]. For this objective, validation of log entries is required. In [20], an authentication and validation methodology for analyzing syslogs was proposed. The proposed mechanism employs a modified version of the Needham Schroeder protocol [21] for key exchange and authentication.

Tao Q. et al. in their paper describes the integrity check module in which HBase is used to store the large volume of logs. Through the preliminary processing of collected data and the design of HBase tables, it implemented the integrity check of collected data. After the integrity check, missing data is found and retransmitted for the following analysis. The research contents of their paper were expected to provide beneficial reference for data integrity check of log collection system in the future. After retransmitted the missing file, the system need to ensure the integrity of the retransmitted file's content, then a program was designed to check the row integrity of the retransmitted file [22].

Wu S. et al. in their paper introduce the notion of secure logging monitor service, which is deployed in the cloud and

generates integrity proofs of cloud logs in real time. Once a proof entry has been produced, a dishonest cloud service provider (CSP) even colludes with the investigator, can't fake or remove the corresponding logs without being detected. Compared with related works, the proposed scheme can simultaneously meet the most major requirements of cloud forensics, including the integrity of log evidences, privacy protection and low computational burden [23].

Lin C. et al. discusses the issues of log evidence first in their paper. The framework of IEAAS (Automated Analysis System of Intrusion Evidences) was illustrated with LCA (Log Collection Agent) in network sensors and multiple modules in IEAAS. Analysis mechanism was discussed, particularly the improved aggregation algorithm and evidence preservation method were described. Then a series of experiments were performed to validate their method on actual attack network environments of CERNET. The results of experiments show that their approach is practical and effective for dynamic forensics to augment the computer crime investigators' efforts [24].

Zhang R. et al. describe the problem of checking the integrity of log in monitoring system for forensics investigation in their paper. Existing frameworks and solutions do not provide a satisfactory result to solve the problem. They either require a mass amount of storage overhead to store the hash values of the events or may not be able to match the situation in an effective way if some events have been modified. They propose an efficient hashing scheme with Shifted Transversal Design Group Testing algorithm to calculate hash values for all events in a log file as the integrity proof and precisely locate the events which have been corrupted. Experimental results show that the storage overhead can be significantly decreased by adopting the scheme [25].

Daci G. et al. considers the standard data verification methods, with the main goal to overcome one of its major limitations, low performance of File System check-summing in their paper. CLFS matches their performance expectations, as it performs close enough to non-cryptographic file systems. To improve the performance of the check-summing process they try to study and examine various design choices and propose metadata check-summing. Several tests were made to prove that this added functionality does not significantly affect performance [26].

Lin C. et al. provide a secure logging framework integrating with the cloud database in their paper. Log auditors here in can use the public key to validate the integrity of log data. The secret key can be use to generate signature of log and block data in this framework. They also provide an implementation for this framework and a performance evaluation of signing/verifying log data. Their study demonstrates a method to protect log data for log owners in the cloud database. Furthermore, the proposed secure logging framework can be easily deployed in a cloud computing environment [27].

Different from the abovementioned studies, in this paper, a novel approach which ensures log security is proposed. After completing a set of predefined steps to ensure that the generated logs have not been changed by unauthorized users, it encrypts and stores logs in a server which is responsible for storing them. In this way, the intruders who have gained access to the system are not able to find their fields of interest in the logs and read/write/change them. In this respect, to ensure data consistency and security, the data composed of the logs are stored in the server's main memory. At the end of the day, the encrypted logs stored in the main memory and those stored in the hard disk drive are decrypted and compared. This way the consistency of the logs is checked and the accuracy of the logs can be proven. If the logs compared are inconsistent, then the situation is reported. Otherwise, the decrypted log file is stamped using time-stamping method and is sent to the server responsible for storing the logs.

3. Proposed System

The proposed system mainly consists of two main modules, namely Timestamping Tool and Listening Tool. The traffic flow being monitored (or "listened to") is recorded on the server along with timestamps. By means of Listening Server, Listening Tool listens traffic on a set of ports and creates log files. At the end of the day, Cryptolog Database creates a report from these log files. Fig. 1 shows the proposed system's overall structure.

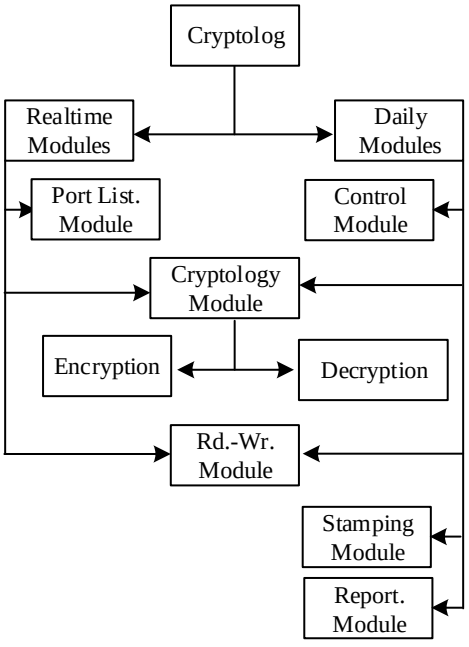


Figure 1. Overall structure of the proposed system

Administrator Panel which can be seen as a door to the proposed system stores access information and log records and sends them to Timestamping Server for time-stamping at the end of the day after checking the log data. It enables its users to view all the records. It provides several user-related functionalities too. During its operation, Listening Tool decrypts stored log records and compares data blocks. If there is inconsistency, the hard drive that stores those logs is marked in Report Status column (See Fig. 2). "Pass"

means that compared log records are consistent. “error_H” means that an error has occurred during reading or decrypting from the hard disk drive but the log records have been recovered using the data blocks stored in the main memory. “error_R” means that an error has occurred during reading or decrypting from the main memory but the log records have been recovered using the data blocks stored in the hard disk drive. “kill” means that an error has occurred during reading or decrypting from both the main memory and the hard disk drive and the logs have been lost. This case means that the attacker both has information about the pointers used and holds the administrator privileges. As shown in Fig. 3, Administrator Panel allows generating reports to have an overview of the stored logs.

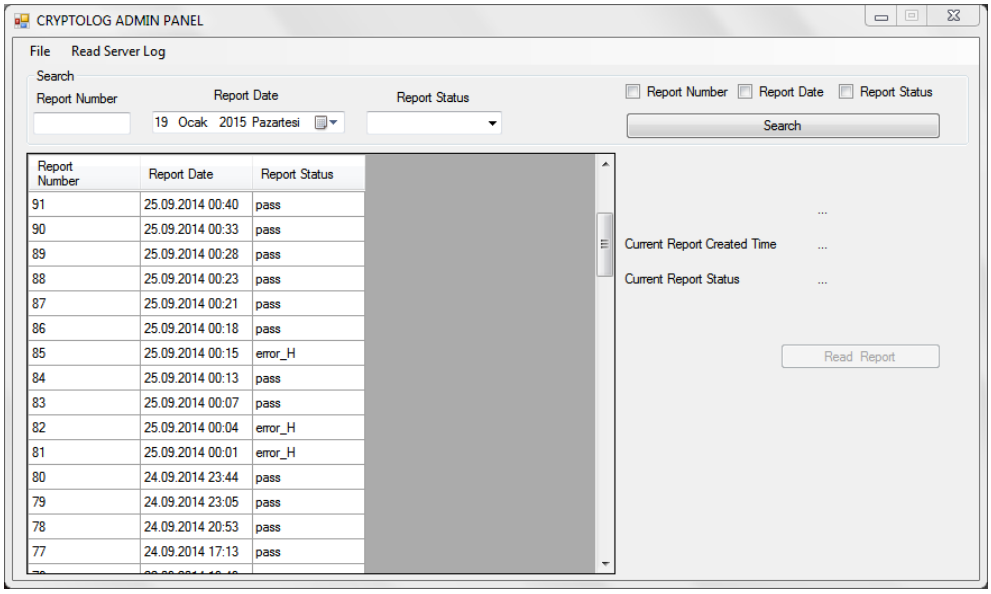


Figure 2. Administrator Panel’s graphical user interface

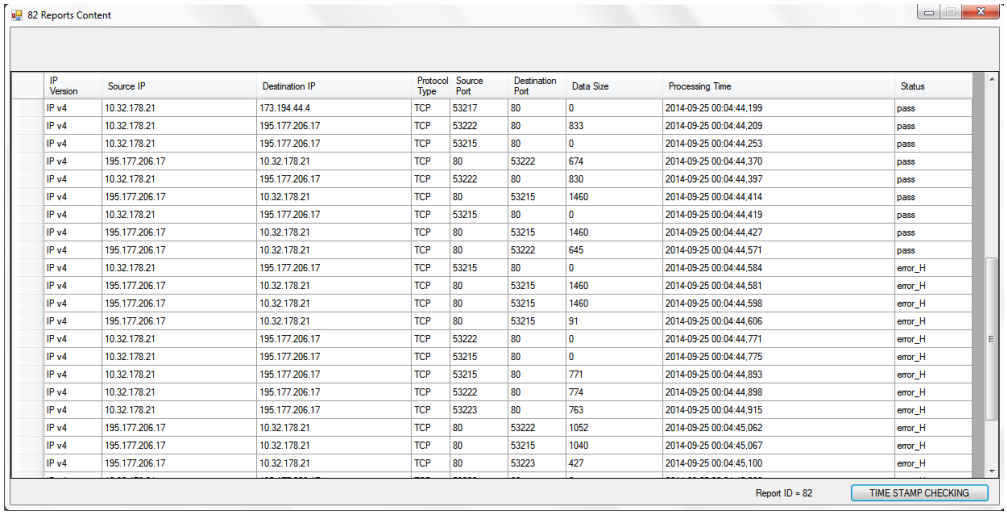


Figure 3. Creating reports using *Administrator* Panel

Timestamping Tool stamps the log data and records in Timestamp Database. As shown in Fig. 4, to connect to Listening Server and Timestamping Server, Administrator Panel uses Connection Settings file. After the connection is available, access information stored in the Cryptolog Database on the Listening Server is used to access the system. Encrypted connection information is stored in Connection Settings file and following connections do not require connection establishment. The user which has the access to Administrator Panel can check whether log files created in Cryptolog Database has been changed or not after having been stamped by Timestamp Database. To perform this, after Administrator Panel sends a checksum code, Timestamping Tool compares the checksum code with the information stored in the Timestamp Database.

Depending on the result of the comparison, a true or false value is returned. The following pseudocode explains the algorithm used by this tool.

Timestamping Tool:

1. Begin
2. Read log records whose numbers have been sent by Listening Tool
3. Serialize log records and create SHA-2 hash
4. Sign the hashed data using RSA and timestamp
5. Save the RSA parameter used, hashed data and the log's record number into the database
6. End

Basically, Listening Tool is responsible for generating logs and uses the functionalities provided by Listening Server. After Administrator Panel has received the connection information for Listening Server from Connection Settings file, Listening Server generates instantaneous data from the traffic flow. Fig. 5 depicts the overall process carried out by this tool. The following pseudocode explains the algorithm used by this tool.

Listening Tool:

1. Begin
2. While true do
3. Capture packets on the selected interface
4. If (the packets belong to the target IP)
5. Then encrypt the packets using AES or 3DES algorithm
6. Hash the packets and store them in the hard disk drive and the main memory
7. If (the time for times-tamping comes)
8. While (!end of log records) do
9. Decrypt the log records in the hard disk drive and the memory
10. If (the log records in the hard disk drive and the memory are consistent)
11. Then Report Status = "Pass"
12. Else if (the log records in the hard disk drive are corrupted)
13. Then Report Status= "error_H"
14. Else Control state="error_R"
15. End
16. End
17. Save the log records into the database and free used resources
18. Send the record number to *Timestamping Tool* for time-stamping
19. End
20. End
21. End
22. End
23. End

Time-stamping Tool is normally in waiting state and runs whenever it receives a command. The system normally receives two types of commands and therefore uses two different algorithms when it runs. One of the commands is responsible for stamping and is sent by the Listening Tool. The other one is responsible for checking stamps and is sent by the Administrator Panel. Whenever data are received from Listening Tool, stamping algorithm is triggered. The data processed by Listening Tool are reported and saved in the database at the end of the day. For each saved report a unique ID is created. After the report has been created, the report's ID and the stamping password stored on Listening Tool are sent to Stamping Tool. Using the ID sent by the Listening Tool, Timestamping Tool receives the date and time of the report and the report's content. Then by using the RSA parameters, Timestamping Tool stamps the data. Finally, it saves the report's ID and hash code generated during the stamping in Timestamp Database. The overall flow of this process is shown in Fig. 6. To perform a stamp check using Administrator Panel, the selected report's IP and stamping password is sent to Timestamping Tool. Then Timestamping Tool reads the report's content and generation date and time from Cryptolog Database. By stamping with the RSA parameter stored on Timestamping Tool, it generates a new hash code and compares it with the hash code of the previously stamped one stored in Timestamp Database. Finally, a true or false value is returned to indicate the result of the comparison.

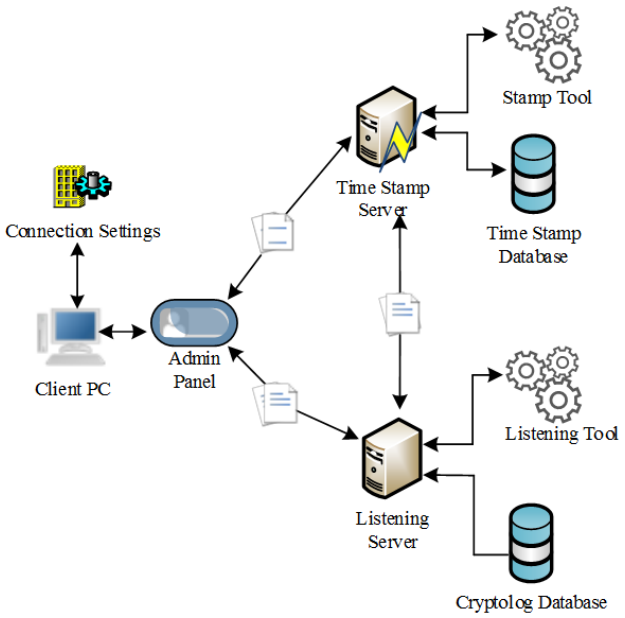


Figure 4. Proposed System

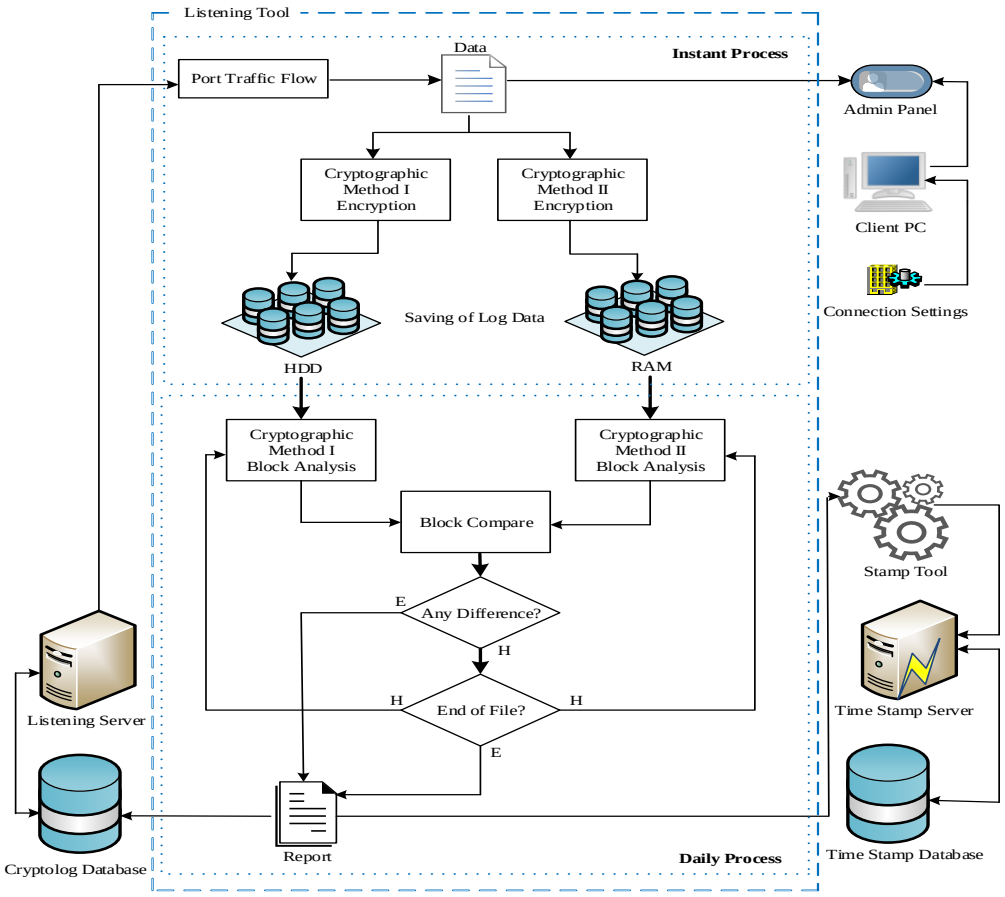


Figure 5. Listening Tool

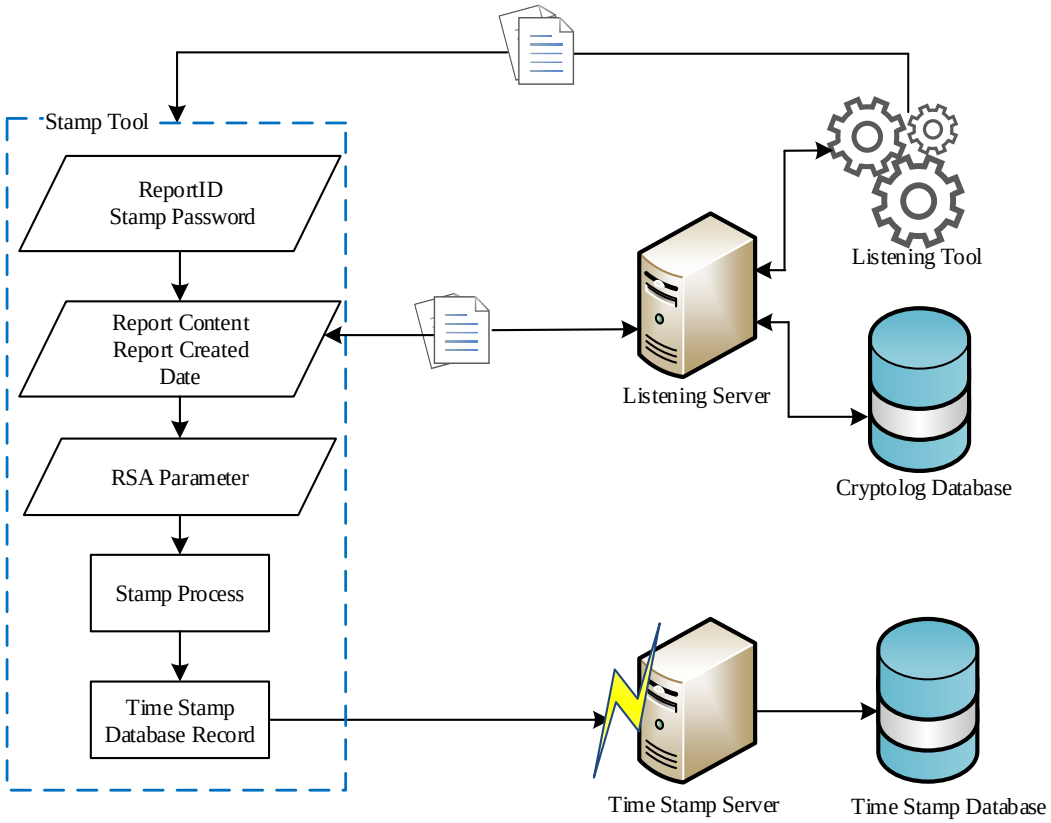


Figure 6. Timestamping Tool

As shown in Fig. 7, each log record consists of eight fields. Log records are encrypted using symmetric encryption algorithms such as the AES and 3DES. Then encrypted log records are saved in a log file in a hard disk drive and written in the main memory at the same time. At the end of the day, the encrypted log records in the hard disk drive and those in the main memory are compared after having been decrypted. If there is no inconsistency in the log records, it is reported that the log records are consistent. Then they are sent to the server responsible for storing the logs. If the log records are inconsistent, it means that the logs have been changed.

IP Version	Source IP	Destination IP	Protocol Type	Source Port	Destination Port	Data Size	Process Time

Figure 7. Log Format

As shown in Figs 8, 9 and 10, after entering the details regarding the proposed system, Listening Tool connects and starts to capture port traffic on the selected interface. Then the captured port traffic is listed.

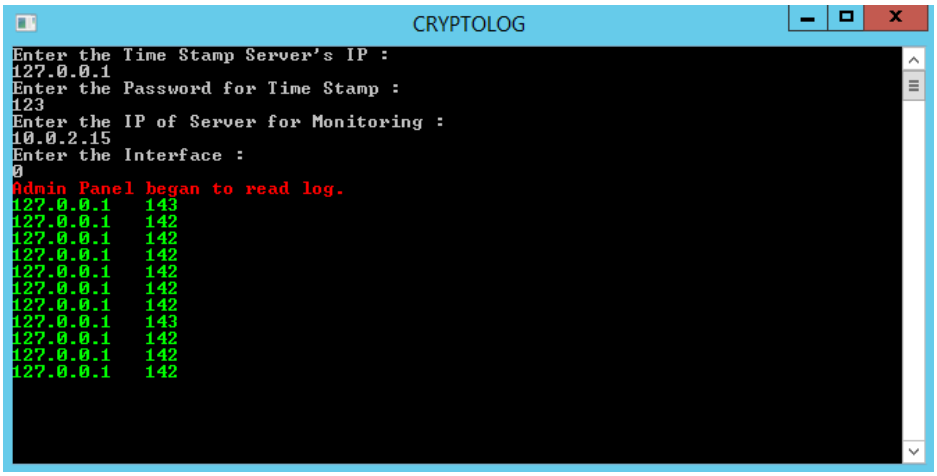


Figure 8. Startup of the proposed system

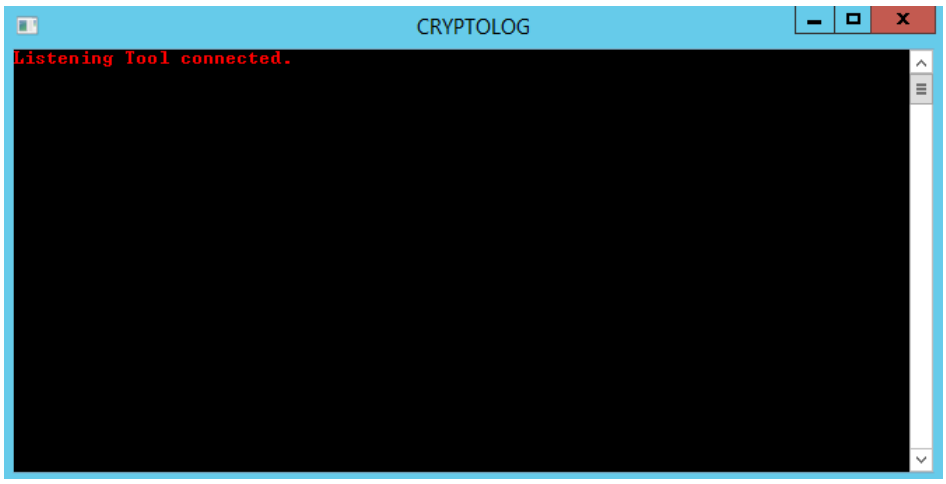
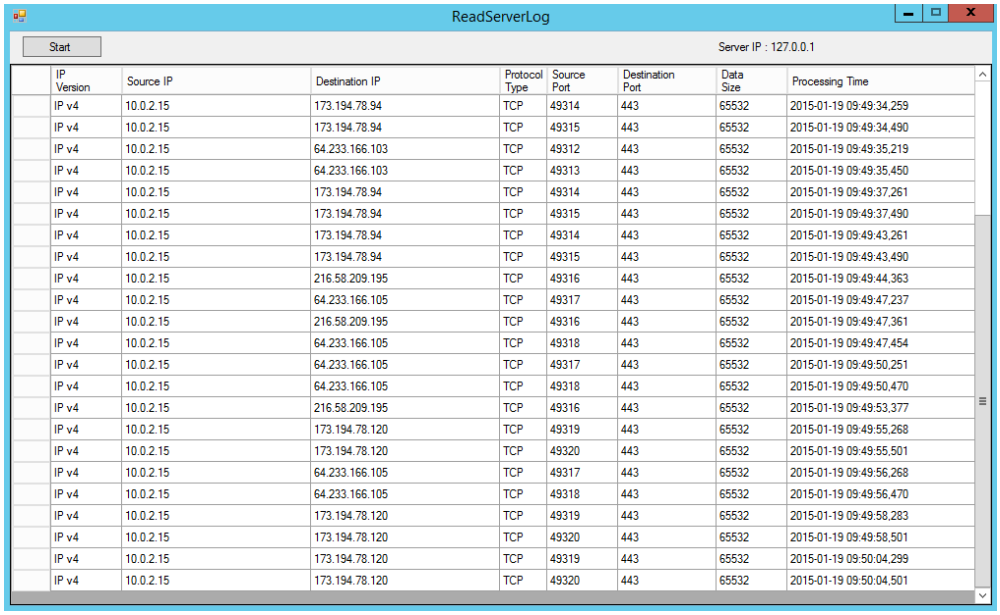


Figure 9. Listening Tool connects and starts to capture port traffic



IP Version	Source IP	Destination IP	Protocol Type	Source Port	Destination Port	Data Size	Processing Time
IP v4	10.0.2.15	173.194.78.94	TCP	49314	443	65532	2015-01-19 09:49:34,259
IP v4	10.0.2.15	173.194.78.94	TCP	49315	443	65532	2015-01-19 09:49:34,490
IP v4	10.0.2.15	64.233.166.103	TCP	49312	443	65532	2015-01-19 09:49:35,219
IP v4	10.0.2.15	64.233.166.103	TCP	49313	443	65532	2015-01-19 09:49:35,450
IP v4	10.0.2.15	173.194.78.94	TCP	49314	443	65532	2015-01-19 09:49:37,261
IP v4	10.0.2.15	173.194.78.94	TCP	49315	443	65532	2015-01-19 09:49:37,490
IP v4	10.0.2.15	173.194.78.94	TCP	49314	443	65532	2015-01-19 09:49:43,261
IP v4	10.0.2.15	173.194.78.94	TCP	49315	443	65532	2015-01-19 09:49:43,490
IP v4	10.0.2.15	216.58.209.195	TCP	49316	443	65532	2015-01-19 09:49:44,363
IP v4	10.0.2.15	64.233.166.105	TCP	49317	443	65532	2015-01-19 09:49:47,237
IP v4	10.0.2.15	216.58.209.195	TCP	49316	443	65532	2015-01-19 09:49:47,361
IP v4	10.0.2.15	64.233.166.105	TCP	49318	443	65532	2015-01-19 09:49:47,454
IP v4	10.0.2.15	64.233.166.105	TCP	49317	443	65532	2015-01-19 09:49:50,251
IP v4	10.0.2.15	64.233.166.105	TCP	49318	443	65532	2015-01-19 09:49:50,470
IP v4	10.0.2.15	216.58.209.195	TCP	49316	443	65532	2015-01-19 09:49:53,377
IP v4	10.0.2.15	173.194.78.120	TCP	49319	443	65532	2015-01-19 09:49:55,268
IP v4	10.0.2.15	173.194.78.120	TCP	49320	443	65532	2015-01-19 09:49:55,501
IP v4	10.0.2.15	64.233.166.105	TCP	49317	443	65532	2015-01-19 09:49:56,268
IP v4	10.0.2.15	64.233.166.105	TCP	49318	443	65532	2015-01-19 09:49:56,470
IP v4	10.0.2.15	173.194.78.120	TCP	49319	443	65532	2015-01-19 09:49:58,283
IP v4	10.0.2.15	173.194.78.120	TCP	49320	443	65532	2015-01-19 09:49:58,501
IP v4	10.0.2.15	173.194.78.120	TCP	49319	443	65532	2015-01-19 09:50:04,299
IP v4	10.0.2.15	173.194.78.120	TCP	49320	443	65532	2015-01-19 09:50:04,501

Figure 10. Listing of the captured port traffic

Table 1. Literature comparison

Comparison of the Work	**Evaluation Criteria**				
	Type	**Applied Area**	**Specifications**	**Performance**	**Specifications that can be added**
Qi Tao et al. [22]	Application	Subway Logs	Log Collection / Integrity Check	✓✓	Data loss control module should be added
Songyang Wu et al. [23]	Review / Suggestions	Cloud Service Providers	Log Collection / Integrity Check	✓	Verification module should be added
Chen Lin et al. [24]	Application	CERNET, Network Sensors	Log Collection / Analysis	✓✓✓	Algorithm, integrity, accuracy should be enhanced
Ruoqing-Zhang et al. [25]	Application	Network Monitoring	Storage / Integrity Check	✓✓✓	Algorithm should be enhanced
Genti Daci et al. [26]	Application	CLFS	File Integrity Check, Secure Environment and Encryption File	✓✓✓	Data audit and collection module should be added
Chung-Yi Lin et al. [27]	Application	Cloud Service Providers	Verifying Data in the Cloud	✓	Timing and multitasking module should be added
Cryptolog	Application	Network Traffic (IP, TCP and UDP Traffic)	Log collection, Privacy, Integrity, Security Check, Parallel Programming and cryptography	✓✓✓	Should be tried for big memory size

Table 1 gives a comparison of our study with similar studies in the literature. These studies are the references numbered 22-27. The benchmarking criteria in the table are the type of study, the field of application, the characteristics, the performance and the features that can be added. The attributes that can be added are the missing parts in the applications and the attributes that need to be improved. The real-time retention of log files is advantageous and security is provided instantaneously (with very short periodic checks). For this reason, studies 22, 23, 24 and our work (Cryptolog) are in the foreground. The integrity of the logs maintained by the system, and the provision of correctness and security, will also be important for the system. In Cryptolog, encryption of the log file with cryptographic methods has been performed in order to satisfy the basic criteria of information security. In addition, Cryptolog is one step ahead because of the parallel programming ability, than the studies compared in the literature.

Although there is no “one size fits all” solution to the information security needs of both governmental and non-governmental organizations and humans, security software providing appropriate security measures should be in place. A common feature which is essential for all types of security software is logging ability which plays a key role for the overall security.

4. Conclusion

In this paper, an alternative logging approach which offers safe and consistent logging by ensuring that the logs have not been changed by unauthorized users is proposed. The approach provides information security measures for log files and thereby enables the log files to be objective evidence for digital forensics processes. It also provides benefits to digital forensics processes in terms of performance and coverage.

The proposed approach compares log files at predetermined time intervals. Since the size of the log files collected during the comparison process is not known in advance and in parallel storing the log files in main memory can result in memory overflows, memory range can be used instead of time intervals. This way, the system runs only using the allocated memory, the load on the server decreases, and the comparison process is executed when the allocated memory is full. After the comparison process, the system runs again using the previously allocated memory. For maximum efficiency, the verification of the log files can be performed using parallel processing technique. Although in the performance evaluation of the developed system, the logs were collected from IP, TCP and UDP traffic, the system allows using other protocols. In sum, the novelty of this study lies in proposing a novel approach to ensure that the collected log files have not been changed by unauthorized users so that the logs are accurate and consistent and can be used as evidence in digital forensics.

5. References

[1] Vassilios Stathopoulos, Panayiotis Kotzanikolaou, and Emmanouil Magkos, "Secure log management for privacy assurance in electronic communications", Computers & Security, vol. 27, no. 7–8, pp. 298-308, December 2008. DOI: 10.1016/j.cose.2008.07.010.

[2] Olof Söderström and Esmiralda Moradian, "Secure Audit Log Management", Procedia Computer Science, vol. 22, pp. 1249-1258, 2013. DOI: 10.1016/j.procs.2013.09.212.

[3] Antonio de la Piedra, An Braeken, Abdellah Touhafi, and Karel Wouters, "Secure event logging in sensor Networks", Computers & Mathematics with Applications, vol. 65, no. 5, pp. 762-773, March 2013. DOI: 10.1016/j.camwa.2012.06.019.

[4] Eoghan Casey, "Experimental design challenges in digital forensics", Digital Investigation, 9 (3-4), pp.167-169, 2013, DOI: 10.1016/j.diin.2013.02.002

[5] Ruoqing Zhang, Zhiwei Chen, Yatao Yang, and Zichen Li, "An efficient scheme for log integrity check in security monitoring system", Proc. IET International Conference on Smart and Sustainable City 2013 (ICSSC 2013), 19-20 Aug. 2013, pp.198-202.

[6] M.T. Goodrich, M.J. Atallah, and R. Tamassia, "Indexing information for data forensics", Lecture Notes in Computer Science, vol. 3531, pp. 206-221, 2005.

[7] Junbin Fang, Zoe L. Jiang, S. M. Yiu, and Lucas C.K.Hui, "An Efficient Scheme for Hard Disk Integrity Check in Digital Forensics by Hashing with Combinational Group Testing", International Journal of Digital Content Technology and its Applications, 5(2), pp.300-308, 2011.

[8] http://www.tib.gov.tr/en/en-menu-47-information_about_the_regulations_of_the_content_of_the_internet.html

[9] Payment Card Industry Security Standards Council, "Payment Card Industry (PCI) Data Security Standard," Payment Card Industry Security Standards Council, Tech. Rep., 2010.

[10] United State Government, "Federal Information Security Management Act (FISMA)," 2002. [Online]. Available: http://csrc.nist.gov/groups/SMA/fisma/index.html

[11] Hemantha S.B. Herath and Tejaswini C. Herath, "IT security auditing: A performance evaluation decision model", Decision Support Systems, vol. 57, pp. 54-63, 2014. DOI: 10.1016/j.dss.2013.07.010.

[12] R. Daş and İ. Türkoğlu, "Creating meaningful data from web logs for improving the impressiveness of a website by using path analysis method", Expert Systems with Applications, vol. 36, no. 3, pp. 6635-6644, 2009.

[13] R. Daş, İ. Türkoğlu, and M. Poyraz, "Web Kayıt Dosyalarından İlginç Örüntülerin Keşfedilmesi", Fırat Üniversitesi Fen ve Mühendislik Bilimleri Dergisi, vol. 19, no. 4, pp. 493-503, 2007.

[14] B. Boeck, D. Huemer, and A Min Tjoa, "Towards More Trustable Log Files for Digital Forensics by Means of "Trusted Computing", 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), 20-23 April 2010, pp. 1020-1027. DOI: 10.1109/AINA.2010.26

[15] D. Brezinski and T. Killalea, "Guidelines for evidence collection and archiving", United States, 2002.

[16] P. D. Dixon, "An overview of computer forensics", IEEE Potentials, vol. 24, no. 5, pp. 7-10, 2005.

[17] B. R. Waters, D. Balfanz, G. Durfee, and D. K. Smetters, "Building an encrypted and searchable audit log", Proc. the 11th Annual Network and Distributed System Security Symposium, 2004.

[18] N. Kawaguchi, S. Ueda, N. Obata, R. Miyaji, S. Kaneko, H. Shigeno, and K. Okada, "A secure logging scheme for forensic computing", Proc. the Fifth Annual IEEE SMC Information Assurance Workshop, June 2004, pp. 386-393.

[19] R. T. Snodgrass, S. S. Yao, and C. Collberg, "Tamper detection in audit logs", Proc. the Thirtieth international conference on Very large data bases, 2004, pp. 504-515.

[20] S. D. S. Monteiro and R. F. Erbacher, "An authentication and validation mechanism for analyzing syslogs forensically", ACM SIGOPS Operating Systems, vol. 42, no. 3, pp. 41-50, 2008.

[21] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers", Communications of the ACM, vol. 21, no. 12, pp. 993-999, 1978.

[22] Q. Tao, L. Yuan and L. Youxun, "Design and implementation of data integrity check of subway log collection system," 2016 IEEE International Conference on Big Data Analysis (ICBDA), Hangzhou, pp. 1-4, 2016.

[23] Songyang Wu and Yong Zhang, "Secure logging monitor service for cloud forensics," 2015 IEEE 16th International Conference on Communication Technology (ICCT), Hangzhou, pp. 757-762, 2015.

[24] C. Lin, L. Zhitang and G. Cuixia, "Automated Analysis of Multi-Source Logs for Network Forensics,", First International Workshop on Education Technology and Computer Science, Wuhan, Hubei, pp. 660-664, 2009.

[25] Ruoqing Zhang, Zhiwei Chen, Zehui Li, Yatao Yang and Zichen Li, "An efficient scheme for log integrity check in

security monitoring system," IET International Conference on Smart and Sustainable City 2013 (ICSSC 2013), Shanghai, pp. 198-202, 2013.
[26] G. Daci and M. Shyle, "Improving data integrity and performance of cryptographic structured log file systems," 2011 3rd International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), Budapest, pp. 1-5, 2011.
[27] C. Y. Lin, M. C. Chang, H. C. Chiu and K. H. Shyu, " Secure logging framework integrating with cloud database," 2015 International Carnahan Conference on Security Technology (ICCST), Taipei, pp. 13-17, 2015.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the financial support from the Scientific Research Projects Administration Unit of Firat University for this study performed under project with grant no. TEKF.15.04.

Muhammet Baykara was born in Elazig, Turkey. He received his BS and MSc. in Computer Engineering Department from Firat University in 2006, 2009 respectively. He received his Ph.D. in Software Engineering Department from Firat University in 2016. Currently, he is a research assistant in the Department of Software Engineering at Firat University. His research interests are Information Security, Honeypots, Intrusion Detection and Prevention Systems.

Resul Daş received his B.Sc. and M.Sc. in Computer Science from Firat University in 1999, 2002 respectively. Dr. Das received his Ph.D. degree from Electrical and Electronics Engineering Department at the same university in 2008. He is currently an Associate Professor at the Department of Software Engineering of Firat University, Turkey. He has authored several papers in international conference proceedings and refereed journals, and has been actively serving as a reviewer for international journals and conferences. His current research interests include complex networks, computer networks, web mining, knowledge discovery, information and network security, software design and architecture and IoT/M2M applications.

Gürkan Tuna is currently an Associate Professor at the Department of Computer Programming of Trakya University, Turkey. Dr. Tuna has authored several papers in international conference proceedings and refereed journals, and has been actively serving as an Associate Editor for IEEE Access and Australian Journal of Electrical and Electronics Engineering journals. His current research interests include smart cities, smart grid, wireless sensor networks, underwater networks and M2M communications.