A New Real-Time File Integrity Monitoring System for Windows-based Environments

Dragi Zlatkovski, Aleksandra Mileva, Kristina Bogatinova and Igor Ampov

Faculty of Computer Science
University "Goce Delčev", Štip
Republic of Macedonia
`draganz87@gmail.com, {aleksandra.mileva, kristina.bogatinova, igor.ampov}@ugd.edu.mk`

Abstract. The ability to get real-time notifications about unexpected changes in files or directory structure occurred by unauthorized accesses is a necessity in the defense from hackers. This paper describes the design and implementation of a new real-time file integrity monitoring system, named WebSGuard. It is a client-server system intended for Windows-based environments. Client agents are installed as system services on the watched web servers and they monitor and report in real-time, while the server application is a desktop application for managing the clients, collecting data from clients, reporting, and sending alerts and notifications to system administrators. The communication between the clients and the server is secure and reliable. The existing prototype currently is used on the University Goce Delcev intranet, for monitoring university's web applications.

Keywords: File integrity checkers, file integrity monitoring, intrusion detection, Windows servers.

1 Introduction

In reality, there are many situations when system administrators find out that their system has been hacked days, weeks or even months previously. If an unprotected machine is compromised, a careful attacker's activity on the machine may never be detected, greatly increasing the amount of damage. The ability to get real-time notifications about changes in files or directory structure occurred by unauthorized access is a necessity for network and web administrators in the defense from hackers. This kind of tools does not prevent the attacks, but are helpful for detection of successful system intrusion.

Changes that occur in the file's or folder's properties (such as content or some attribute) are quite common and normal for a given file system, but also they are a basic part of most of the hacker attacks. System administrators need a way to distinct nonmalicious and authorized changes from malicious and unauthorized changes to the file systems. So, they employ file integrity monitoring (FIM) tools (also called file integrity checking or change auditing tools) to track different important files or folders, such as configuration files, registry files, executables, web site resources, file and directory permissions, tables, indexes, stored procedures, rules, etc. These tools monitor the changes of different properties, like credentials, privileges and security settings, file content, core attributes and size, hash values, configuration values, etc.

Most of the FIM tools operate as user-mode utilities and they first establish a known and trusted state of a system, after what they perform scheduled checks for detecting changes and alerting the administrators. At a minimum, an FIM solution should be able to establish a trusted state for protected files and folders, monitor for configuration change relative to the trusted state, determine if change is authorized or unauthorized, alert when unauthorized change occurs, and provide detailed information to help the administrators remediate any improper changes.

Importance of FIM can be seen also by the fact that today several well-established compliance standards and regulatory compliance acts indicate FIM to be implemented, such as:

- PCI-DSS - Payment Card Industry Data Security Standard (Requirement 10.5.5 and 11.5)
- SOX - Sarbanes-Oxley Act (Section 404)
- NERC CIP - NERC CIP Standard (CIP-010-2)
- FISMA - Federal Information Security Management Act (NIST SP800-53 Rev3)
- HIPAA - Health Insurance Portability and Accountability Act of 1996 (NIST Publication 800-66)
- SANS Critical Security Controls (Control 3).

This paper describes the design and implementation of a new real-time file integrity monitoring system, named WebSGuard. In Section 2 the basics and categorization of the file integrity monitoring tools are given. Section 3 describes the architecture of the new real-time FIM solution, with a special description of the server and the client components. In Section 4, some details of the implementation are given.

2 File Integrity Monitoring

File integrity monitoring is one of the most popular approaches to discover malicious behavior by detecting modification actions on protected files and folders, such as modifying different log files, inserting new files, etc. FIM tools are a host-based intrusion detection software, and they can help identifying which files or directories may have been damaged or manipulated, by which user, in what time, etc. The idea for FIM originate from a seminal paper by James Anderson [1].

Generally, there are two classes of FIM tools: periodic FIM and real-time FIM [5]. Periodic or pool-based FIM tools check periodically current file attributes, like file size or last modification time, and compare them with previously collected one. This process ensures that the files are not damaged or manipulated within a time interval that determines the comparison, usually by keeping track of cryptographic hashes of files at different points in time. The first such a tool is Tripwire [6], which takes snapshots first for all the files that need to be protected (by generating file signatures), and later detects if they have been tampered with. Similar tools are Unix-based Advanced Intrusion Detection Environment (AIDE) [8], Osiris [13] and Samhain [**?**], and cross-platform Verisys [11], OSSEK [4] and CimTrak [3].

Periodic FIM tools have several disadvantages, such as they are less effective in detecting attacks that happen between scheduled checks, they can be easily be compromised by attackers with root privileges, and they significantly degrade system performance during the checks. Most of them, like Tripwire, use SSH and SSL/TLS for securing the communication.

Real-time FIM tools detect changes in real time, and they can be divided in several groups. The first group is deployed as a kernel module in the OS, which means they are platform-dependent. They insert hooks into the OS kernel, to intercept read and write system calls, like XenFIT [10], and I^{3}FS [9]. Some of them, even immediately block access to the affected file before notifying the administrator. The problem with these real-time FIM tools is that their kernel module can be easily attacked or masked by rootkits. The second group is deployed as a module in the Virtual Machine Monitor (VMM), under the traditional OS, so cannot be accessed by attackers. One example is VMFence [5], where the real-time FIM tool is implemented in one privileged virtual machine, while it observes file operations in other monitored virtual machines, through the System call sensor module inserted in the VMM. Similarly, another virtual machine monitor, VRFPS [?], introspects all file operations of guest OS, and implements a virtual sandbox in privileged domain to prevent protected files in guest domain from modifying illegally. FSGuard [12] is another example for the Xen virtualization platform. There are also hardware based protection mechanisms for integrity verification which require Trusted Platform Module (TPM) chips embedded on the computer hardware and an additional software to make it efficient. For example, there are snoop-based kernel integrity monitoring tools that snoop the bus traffic of the host system from a separate independent hardware, like Vigilare system [7], which work by adding Snooper hardware connections module to the host system for bus snooping.

3 System Architecture

WebSGuard is a real-time FIM tool with client/server architecture. It consists of an administrator component with a graphical interface designed for administrators, and client agents as services of the operating system intended for supervision of web applications and reporting to the server about the changes occurring in real time (Figure 1). There is one client per web server, which can monitor one or more web applications. In the same manner, it can be used for monitoring of any files and folders or the overall changes that occur during operation of the operating system, not only web applications. Beside intrusion detection, it can also be used for other purposes, such as monitoring and recording information during the installation/uninstallation of a software in the computer, or review and analysis of the files that have access covered by a form application. The existing prototype can currently run only on Windows-based environments, and now it is used on the University Goce Delcev intranet, for monitoring university's web applications.

Additionally, WebSGuard can work in the Periodic FIM mode, by taking a snapshot of the protected file structure in specified time intervals, or on request, and comparing the current state with the previously saved good state.

The new solution uses an encrypted communication between the server component and the clients. In this way attackers can not perform passive attacks by eavesdropping the exchanged messages, but also, without administrator privileges, can not successfully misrepresent them as a server or as an existing client. Administrator (or admin or server) component is a visual desktop application that is installed on the administrative side. It is responsible for clients management, for receiving messages sent by

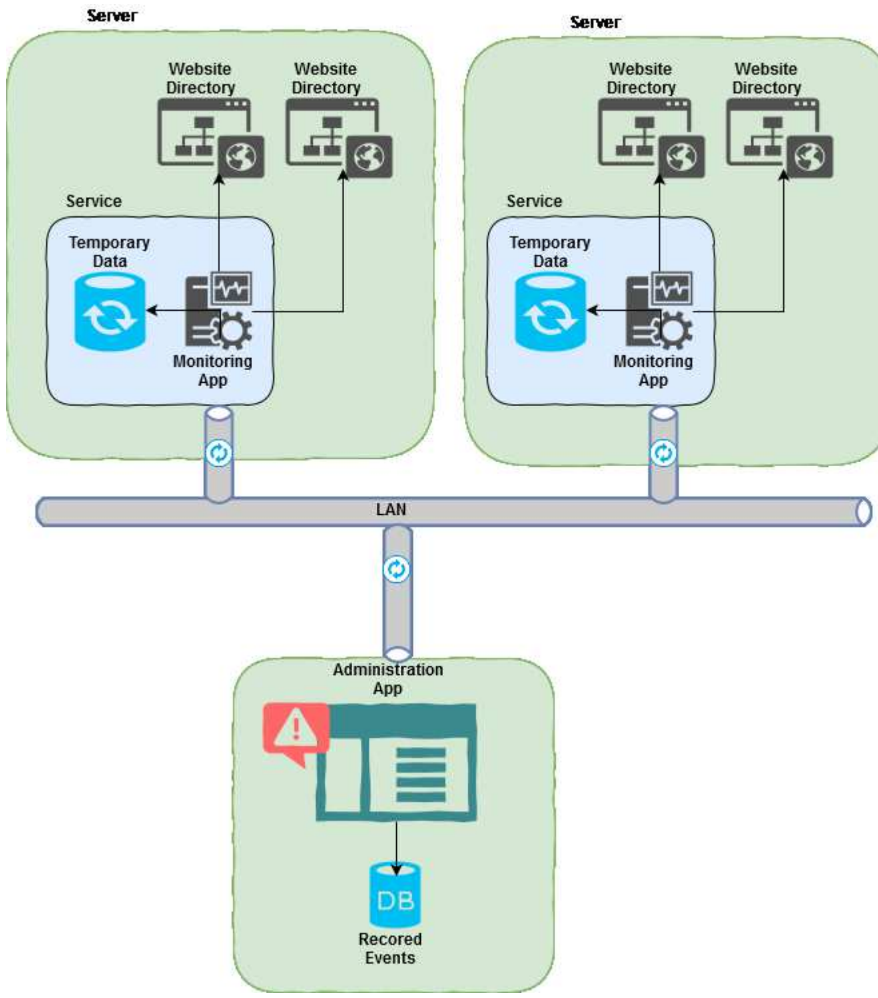


Fig. 1. System architecture

clients and their secure storage in the database, as well as for informing the administrator for occurred changes. administrator component is tasked to add, delete, configure and monitor each client individually, to send commands and settings to clients, receive alerts on any change of the protected web application, to properly handle messages, to store received information in the database and to inform the administrator for important changes.

Client agents are implemented as Windows services that are actively involved in the background of the operating system. Their task is to receive commands by the administrator component and to perform continuous monitoring of specific directories or files, as well as to catch and send information about the changes to the administrator component. Additionally, agents are required to perform the recording of a particular web application state at the administrator request.

All resulting changes obtained from all connected clients are recorded in a database on the administrative side for further processing and extraction of the report for a certain period. When a client fails to establish a connection with the administrator component, it uses its own temporary database for storing changes that are caught on the watched side, while waiting for a connection.

3.1 Administrator Component

This component is a central management system for all client agents that are listed in the application. Unlike the clients, this component has visual appeal, through which the administrator can easily manage and control the operations of the system.

This component is composed of several modules (Figure 2):

1. **GUI Module**: This module is a central administration console through which a detailed overview of all available clients is shown, with the possibility of remote directory listing and traversing for each host web server. There is a menu to all functionalities of the application. There are three sections that provide information on events caused by modification of protected file systems. First section displays all the occurred events, second section displays only the important events, and the third section gives the information about events connected with designated certain types of files. The GUI Module also serves for configuration of the administrator component.
2. **Watcher Parameters Module**: This module is responsible for adding/ removing directories on the web servers with client agents on them,

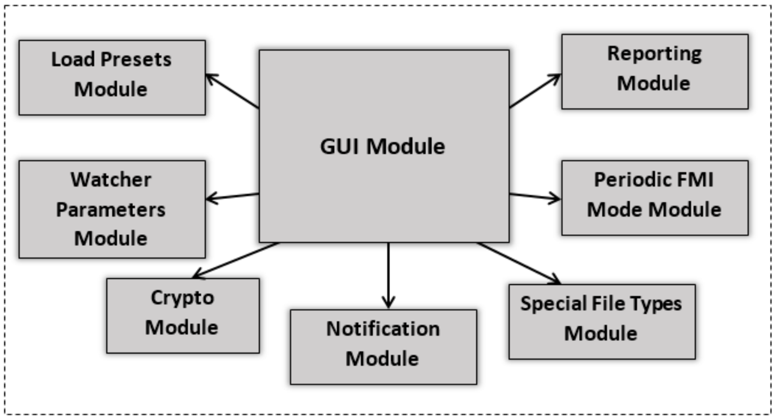


Fig. 2. Administrator component

with specific changes that will be tracked, setting the size of the buffer and the filters needed for the client, etc.

3. **Load Presets Module**: This module allows reloading of the previously defined rules for a specific web application.
4. **Reporting Module**: This module allows searching through all incurred changes recor-ded in the database. It allows searching of events for different time periods, different kind of change, for a particular web application, etc. It also allows removal of the searched data in .CSV file or export directly from the database. Different colors are used for different type of events, with a number of different events and representation with a chart. Through the same module the administrator can perform a physical deletion of data from the database according to specified criteria.
5. **Periodic FMI Mode Module**: This module allows the comparison of pre-recorded state of the structure of a particular web application and the currently recorded state. It displays the details of all newly created files, of all deleted files and of all changed files, and exports the entire report in the .CSV file.
6. **Special File Types Module**: This module allows configuration of events for certain types of files to get a special part in the main screen. For example, php scripts are often used by attackers for hiding shells, so this file extension need to be watched carefully.
7. **Notification Module**: This module is responsible for notification of the administrators.

8. **Crypto Module**: This module is responsible for generating and storing random session secret key for each client, and encrypting/decrypting the messages to/from the clients. The session key is generated randomly and separately for each new TCP connection. This module periodically sends ping packets to each client, to check its aliveness on the Internet. The first message from the server to the client, which contains the randomly generated session secret key K is encrypted with the public key of the particular client, and each other message to the client is encrypted with the session key K. For reliable communication, each obtained message from the client is acknowledged.

3.2 Client Agent

The client agent represents a small software component that is installed and erected on the side of the watched web server as a Windows service, responsible for the continuous monitoring, for executing commands given by the administrator component and for sending notifications to the administrator component. Client part consists of five modules (Figure 3):

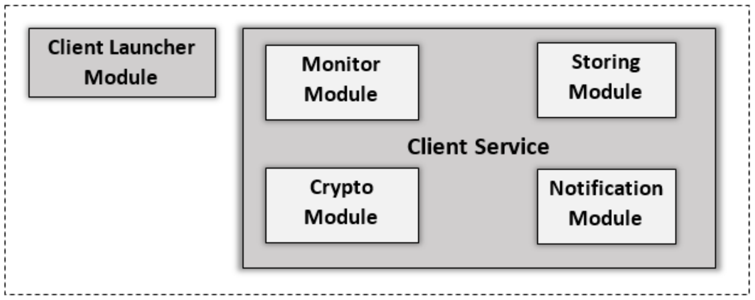


Fig. 3. Client component

1. **Client Launcher Module**: This module offers a visual interface to change the client settings.
2. **Monitor Module**: This module is responsible for monitoring directory with all its subdirectories and files and reporting in real time for any change. Gets the configuration set by the administrator component to know on what criteria to perform monitoring.
3. **Storing Module**: All resulting changes are stored in the temporary database. This enables the client to run independently, even without any interaction with the administrator component.

4. **Notification Module**: If the client is connected to the administrator component, it performs notification in the real time for any change in the watched file system. Notifications are sent encrypted over the network. If the client is disconnected from the administrator component, it stores gathered changes in the temporary database, and after the connection is reestablished, it sends all the records to the administrator components. Once the client will receive a confirmation of successful recorded data, it empties its temporary database. The client checks if the connection with the administrator components is alive, with sending of the ping packets periodically.
5. **Crypto Module**: This module is responsible for the generation of a pair of public and private keys for the client agent (on client launching) and the encryption/decryption of all messages to/from the administrator component. The first message to the administrator component, contains the client's public key. The first response from the server contains the session key K, and it is encrypted with the client's public key. Any other communication between the client and the server is encrypted with the session key K.

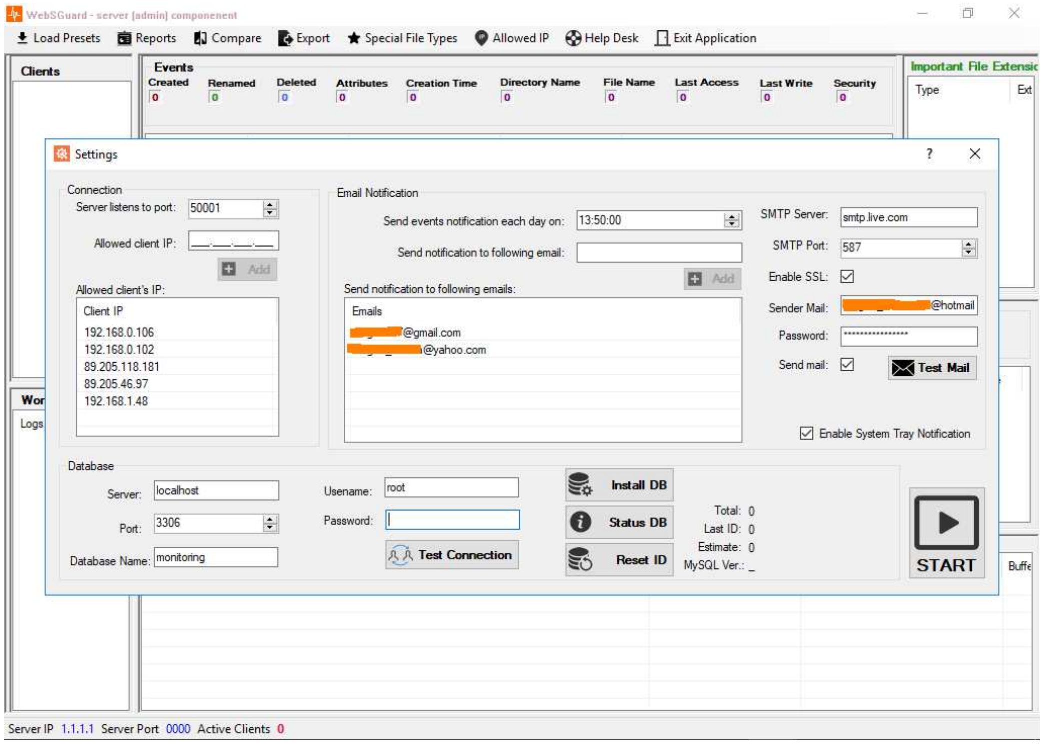
4 Implementation

WebSGuard is a .NET application, which uses temporary SQLite databases for clients and MySQL database for the server component. It is currently intended for use only on Microsoft Windows-based systems. JSON format is used for packing/unpacking messages between the server component and the clients. The minimum requirements for using the prototype are:

- Operating system (client version): Windows Vista, 7, 8, 8.1, 10 with installed .NET Framework 4.5
- Operating system (server version) Microsoft Windows Server 2012, 2012 R2, 2016 with installed .NET Framework 4.5
- Database Server: MySQL version 5.7 or Maria DB version 10.

The administrator component can be configured through the GUI Module (Figure 4). In the Settings part of the GUI Module, one can add or remove clients, specify server listening port, configure the database, and configure and manage e-mail notifications to the administrators. The client component can be configured through the Client Launcher Module (Figure 5), where the server IP address and listening port are specified.

Once the two components (admin and clients) are installed and properly configured on the appropriate machines, a secure connection between



Created	Renamed	Deleted	Attributes	Creation Time	Directory Name	File Name	Last Access	Last Write	Security
0	0	0	0	0	0	0	0	0	0

Connection

Server listens to port: 50001

Allowed client IP: ___.___.___.___

Allowed client's IP:

Client IP
192.168.0.106
192.168.0.102
89.205.118.181
89.205.46.97
192.168.1.48

Email Notification

Send events notification each day on: 13:50:00

Send notification to following email:

Send notification to following emails:

Emails
[redacted]@gmail.com
[redacted]@yahoo.com

SMTP Server: smtp.live.com

SMTP Port: 587

Enable SSL: ☑

Sender Mail: [redacted]@hotmail

Password: ****************

Send mail: ☑

☑ Enable System Tray Notification

Database

Server: localhost

Port: 3306

Database Name: monitoring

Usename: root

Password:

Total: 0

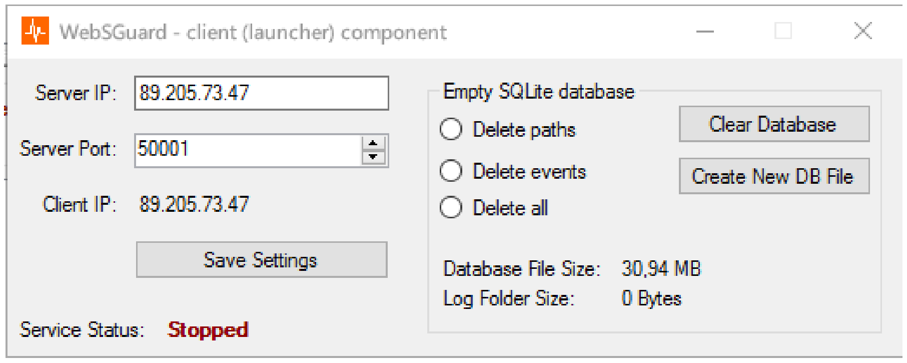
Last ID: 0

Estimate: 0

MySQL Ver.: _

Server IP 1.1.1.1 Server Port 0000 Active Clients 0

Fig. 4. Configuration of the administrator component through GUI Module



Server IP: 89.205.73.47

Server Port: 50001

Client IP: 89.205.73.47

Service Status: Stopped

Empty SQLite database

○ Delete paths

○ Delete events

○ Delete all

Database File Size: 30,94 MB

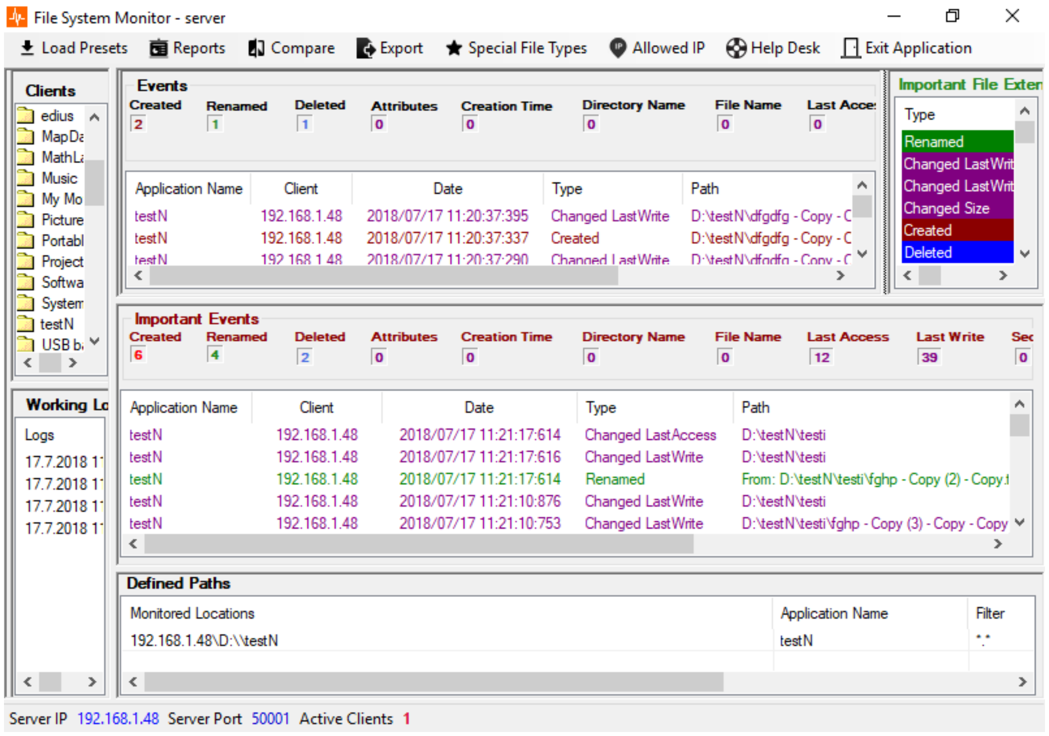
Log Folder Size: 0 Bytes

Fig. 5. Configuration of the client component through Client Launcher Module

them is established. administrator component can communicate only with clients with allowed IP addresses. Otherwise, the administrator component rejects the communication and notifies the administrator for that action. A secure communication is obtained by use of RSA public key cryptosystem and AES-256 block cipher. The client during the configuration phase, generates its RSA pair of private and public keys, and sends its public key to the server component on its listening port. The server component generates a random AES-256 secret key K, per client and per TCP connection, and sends it to the appropriate client as message encrypted with the public key of the client. The client decrypts the message, and obtain the session key K. Any other message transferred between the client and the server during one TCP connection is encrypted with the session key K. Additionally, the recommendation is the administrator component to be accessible only on the organizational intranet. In the next version of the implementation, we are planning the administrator component to generate its RSA pair of private and public keys also, and the server public key to be configured on each client, so, the first message from the client to the server to be encrypted with the server public key. In this way, a man-in-the-middle attacks on the intranet will be mitigated.

When a secure connection is established between the client and the server component, the directory structure of the watched server is displayed in the GUI Module of the administrator component (Figure 6). The administrator can choose which directory or web application to be monitored. For that, a Watcher Parameters Module (Figure 7) is opened, in which the monitoring rules are determined and the configuration of the client about how much memory to reserve for obtained events is done. For monitoring rules, the administrator can define important subnode or sub-path for watching, can exclude sub-node or sub-path from watching, or can specify which kind of events to be watched for giving sub-node or sub-path. There are four main types of events: Created, Deleted, Renamed and Changed, where Changed includes change of file/directory name, file/directory attributes, creation, modification or access time, file/directory size, file/directory owner, file/directory permissions, etc. The administrator can define an additional filter on the specified sub-node or sub-path, which can accept wildcards.

After configuration of the protected files and directories, the administrator component sends this information to the client, and the client begins to monitor them. When the client receives a signal from the operating system that a change has occurred in the monitored structures,



File System Monitor - server

Load Presets | Reports | Compare | Export | Special File Types | Allowed IP | Help Desk | Exit Application

Clients

edius
MapDa
MathLa
Music
My Mo
Picture
Portabl
Project
Softwa
System
testN
USB b

Events

Created	Renamed	Deleted	Attributes	Creation Time	Directory Name	File Name	Last Acce
2	1	1	0	0	0	0	0

Application Name	Client	Date	Type	Path
testN	192.168.1.48	2018/07/17 11:20:37:395	Changed LastWrite	D:\testN\dfgdfg - Copy - C
testN	192.168.1.48	2018/07/17 11:20:37:337	Created	D:\testN\dfgdfg - Copy - C
testN	192.168.1.48	2018/07/17 11:20:37:290	Changed LastWrite	D:\testN\dfgdfg - Copy - C

Important File Exten

Type
Renamed
Changed LastWrit
Changed LastWrit
Changed Size
Created
Deleted

Important Events

Created	Renamed	Deleted	Attributes	Creation Time	Directory Name	File Name	Last Access	Last Write	Sec
6	4	2	0	0	0	0	12	39	0

Application Name	Client	Date	Type	Path
testN	192.168.1.48	2018/07/17 11:21:17:614	Changed LastAccess	D:\testN\testi
testN	192.168.1.48	2018/07/17 11:21:17:616	Changed LastWrite	D:\testN\testi
testN	192.168.1.48	2018/07/17 11:21:17:614	Renamed	From: D:\testN\testi\fghp - Copy (2) - Copy.t
testN	192.168.1.48	2018/07/17 11:21:10:876	Changed LastWrite	D:\testN\testi
testN	192.168.1.48	2018/07/17 11:21:10:753	Changed LastWrite	D:\testN\testi\fghp - Copy (3) - Copy - Copy

Working Lo

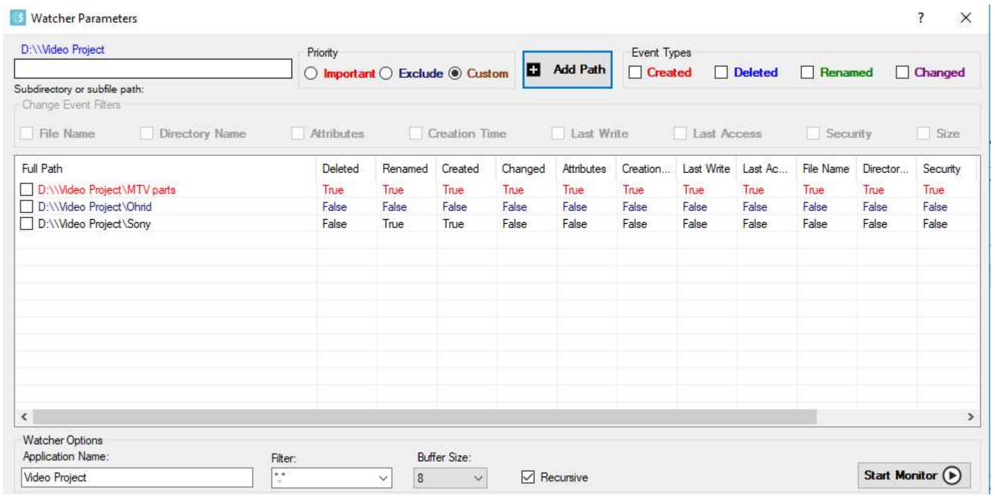
Logs
17.7.2018 1
17.7.2018 1
17.7.2018 1
17.7.2018 1

Defined Paths

Monitored Locations	Application Name	Filter
192.168.1.48\D:\\testN	testN	*.*

Server IP 192.168.1.48 Server Port 50001 Active Clients 1

Fig. 6. Main view of GUI Module.



Watcher Parameters

D:\\Video Project

Subdirectory or subfile path:

Priority: ○ Important ○ Exclude [● Custom]

Add Path

Event Types: ☐ Created ☐ Deleted ☐ Renamed ☐ Changed

Change Event Filters: ☐ File Name ☐ Directory Name ☐ Attributes ☐ Creation Time ☐ Last Write ☐ Last Access ☐ Security ☐ Size

Full Path	Deleted	Renamed	Created	Changed	Attributes	Creation...	Last Write	Last Ac...	File Name	Director...	Security
☐ D:\\Video Project\MTV parts	True	True	True	True	True	True	True	True	True	True	True
☐ D:\\Video Project\Ohrid	False	False	False	False	False	False	False	False	False	False	False
☐ D:\\Video Project\Sony	False	True	True	False	False	False	False	False	False	False	False

Watcher Options

Application Name: Video Project

Filter: *.*

Buffer Size: 8

☑ Recursive

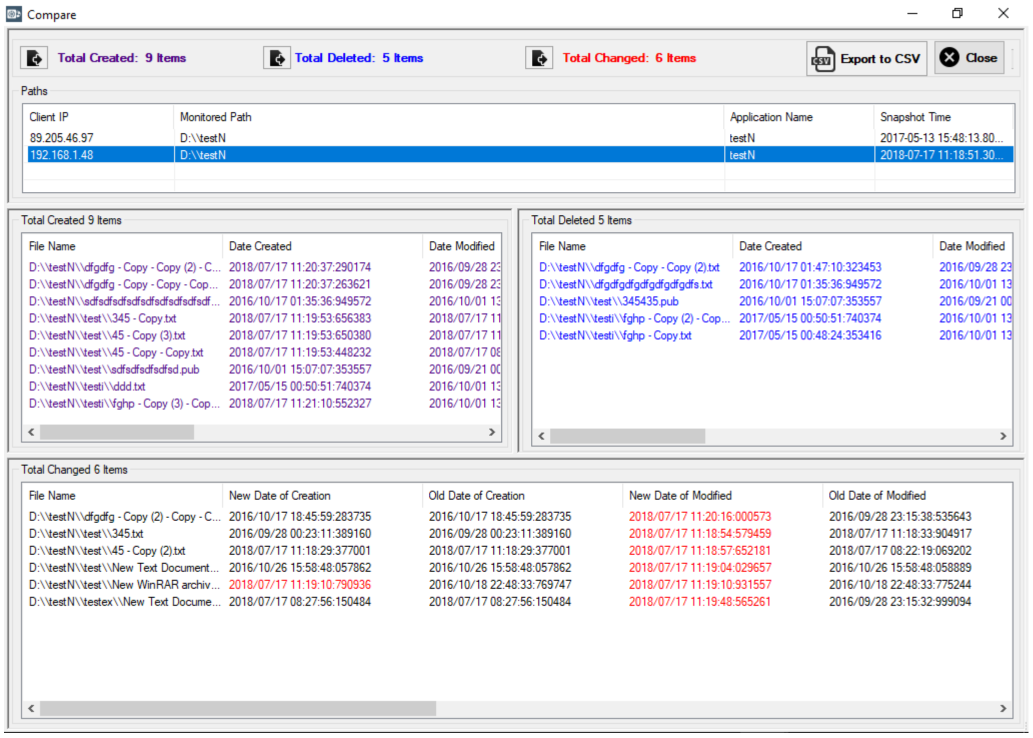
Start Monitor

Fig. 7. Watcher Parameter Module

the appropriate event is saved in the client's temporary SQLite database and if there is a connection with the administrator component, immediately a notification is sent. The administrator component, depending on the type and nature of the event decides which category of visual section will display the information, or what kind of notification to send to the administrator in some cases. The administrator component also stores all received events in its MySQL database for further analysis, and sends a signal to the client by emphasizing that can wipe already processed events from its temporary database. Should there be a break in the connection, the client continues to monitor because it knows the rules and has a temporary repository to store all events. Once the client will have a new connection to the administrator component, it will automatically synchronize with the administrator component.

All notifications to the administrator are performed by the administrator component. The application allows two types of notifications: a notification through email addresses at some point in the day of the events in the past 24 hours, and a notification through system try messages for important events. The administrator can also search and review the events through the Reporting Module. For records can be exported as a .csv file. Searches can be conducted by various criteria, such as for a time interval, for a certain type of event, for a web application or a client, and directly search for a specific phrase. Deleting events from the server database is carried out for a period of time for a specific Web application, a specific IP address or delete all events for a particular phrase that occurs in the path name of the directory/file.

WebSGuard can work in the period FMI mode also, through the module with the same name. The administrator component sends a signal to the client to make a snapshot of the full structure of the requested node, by using cryptographic hashes with SHA-2 hash function. The obtained data is sent to the administrator component and is stored as the initial good state for that node. After some time the administrator can compare the stored state for a node with the current situation, to see the changes. Upon administrators request or in some time interval, the administrator component sends a request to the client to capture the current state, so comparison can be made. Deleted, changed and newly created files are represented separately (Figure 8).



Paths

Client IP	Monitored Path	Application Name	Snapshot Time
89.205.46.97	D:\\testN	testN	2017-05-13 15:48:13.80...
192.168.1.48	D:\\testN	testN	2018-07-17 11:18:51.30...

Total Created 9 Items

File Name	Date Created	Date Modified
D:\\testN\\dfgdfg - Copy - Copy (2) - C...	2018/07/17 11:20:37:290174	2016/09/28 23
D:\\testN\\dfgdfg - Copy - Copy - Cop...	2018/07/17 11:20:37:263621	2016/09/28 23
D:\\testN\\sdfsdfsdfsdfsdfsdfsdfsdf...	2016/10/17 01:35:36:949572	2016/10/01 13
D:\\testN\\test\\345 - Copy.txt	2018/07/17 11:19:53:656383	2018/07/17 11
D:\\testN\\test\\45 - Copy (3).txt	2018/07/17 11:19:53:650380	2018/07/17 11
D:\\testN\\test\\45 - Copy - Copy.txt	2018/07/17 11:19:53:448232	2018/07/17 08
D:\\testN\\test\\sdfsdfsdfsdfsd.pub	2016/10/01 15:07:07:353557	2016/09/21 00
D:\\testN\\testi\\ddd.txt	2017/05/15 00:50:51:740374	2016/10/01 13
D:\\testN\\testi\\fghp - Copy (3) - Cop...	2018/07/17 11:21:10:552327	2016/10/01 13

Total Deleted 5 Items

File Name	Date Created	Date Modified
D:\\testN\\dfgdfg - Copy - Copy (2).txt	2016/10/17 01:47:10:323453	2016/09/28 23
D:\\testN\\dfgdfgdfgdfgdfgdfgdfs.txt	2016/10/17 01:35:36:949572	2016/10/01 13
D:\\testN\\test\\345435.pub	2016/10/01 15:07:07:353557	2016/09/21 00
D:\\testN\\testi\\fghp - Copy (2) - Cop...	2017/05/15 00:50:51:740374	2016/10/01 13
D:\\testN\\testi\\fghp - Copy.txt	2017/05/15 00:48:24:353416	2016/10/01 13

Total Changed 6 Items

File Name	New Date of Creation	Old Date of Creation	New Date of Modified	Old Date of Modified
D:\\testN\\dfgdfg - Copy (2) - Copy - C...	2016/10/17 18:45:59:283735	2016/10/17 18:45:59:283735	2018/07/17 11:20:16:000573	2016/09/28 23:15:38:535643
D:\\testN\\test\\345.txt	2016/09/28 00:23:11:389160	2016/09/28 00:23:11:389160	2018/07/17 11:18:54:579459	2018/07/17 11:18:33:904917
D:\\testN\\test\\45 - Copy (2).txt	2018/07/17 11:18:29:377001	2018/07/17 11:18:29:377001	2018/07/17 11:18:57:652181	2018/07/17 08:22:19:069202
D:\\testN\\test\\New Text Document...	2016/10/26 15:58:48:057862	2016/10/26 15:58:48:057862	2018/07/17 11:19:04:029657	2016/10/26 15:58:48:058889
D:\\testN\\test\\New WinRAR archiv...	2018/07/17 11:19:10:790936	2016/10/18 22:48:33:769747	2018/07/17 11:19:10:931557	2016/10/18 22:48:33:775244
D:\\testN\\testex\\New Text Docume...	2018/07/17 08:27:56:150484	2018/07/17 08:27:56:150484	2018/07/17 11:19:48:565261	2016/09/28 23:15:32:999094

Fig. 8. Periodic FMI Mode Module

5 Conclusion and Future Work

We present a new real-time file integrity monitoring system, named WebSGuard, as a client/server application for Windows-based environments. Main future objective is WebSGuard to became cross-platform. One other possibility of improvement is adding a new functionality to this tool, with the deployment of some artificial intelligence techniques on the database data, for improving intrusion detection on the watched web server. Another possibility is implementation of immediately blocking of a given access to the manipulated file, before notifying the administrator.

References

1. Anderson, J.P.: Computer Security Threat Monitoring and Surveillance [White Paper]. Fort Washington, PA: James P. Anderson Co. (1980) [Online]. Available: http://csrc.nist.gov/publications/history/ande80.pdf (current August 2017)
2. File Integrity AIDE, Ubuntu documentation, (2016), [Online]. Available: https://help.ubuntu.com/community/FileIntegrityAIDE (current August 2017)
3. CimTrak Platform, Cimcor, [Online]. Available: https://www.cimcor.com/cimtrak/ (current August 2017)
4. Hay, A., Cid, D., Bray, R: OSSEC Host-Based Intrusion Detection Guide. Syngress Publishing, Inc., Elsevier, Inc., Burlington, USA (2008)
5. Jin, H., Xiang, G., Zou, D., Zhao, F., Li, M., Yu, C.: A Guest-transparent File Integrity Monitoring Method in Virtualization Environment. Computers & Mathematics with Applications. 60(2), 256–266 (2010)
6. Kim, G., Spafford, E. H.: The Design and Implementation of Tripwire: A File System Integrity Checker. In: 2nd ACM Conference on Computer and Communications Security, pp. 18–29. ACM, Fairfax, VA, USA (1994)
7. Moon, H., Lee, H., Lee, J., Kim, K., Peak, Y., Kang, B.B.: Vigilare: toward Snoop-based Kernel Integrity Monitor. In: ACM conference on Computer and Communications Security 2012, pp. 28–37. ACM, Raleigh, North Carolina, USA (2012)
8. Nel, M.: SAMHAIN: Host Based Intrusion Detection via File Integrity Monitoring. SANS Institute InfoSec Reading Room, (2014)
9. Patil, S., Kashyap, A., Sivathanu, G., Zadok, E.: I3FS: An In-kernel Integrity Checker and Intrusion Detection File System. In: 18th USENIX Large Installation System Administration Conference (LISA '04), pp. 67–78. USENIX Association, Atlanta, GA, USA (2004)
10. Quynh, N.A., Takefuji, Y.: A Novel Approach for a File-system Integrity Monitor Tool of Xen Virtual Machine. In: 2nd ACM Symposium on Information, Computer and Communications Security, pp. 194–203. ACM, Singapore (2007)
11. Ionx Solutions, Verisys File Integrity Monitoring. [Online]. Available: https://www.ionx.co.uk/products/verisys (current August 2017)
12. Wang, Z., Huang, T., Wen, S.: A File Integrity Monitoring System Based on Virtual Machine. In: 2nd International Conference on Instrumentation & Measurement, Computer, Communication and Control, pp. 653–655. IEEE, Harbin, China (2012)
13. Wotring, B., Potter, B.: Host Integrity Monitoring Using Osiris and Samhain. Syngress Publishing, Inc., Elsevier, Inc., Burlington, USA (2005)

14. Zhao, F., Jiang, Y., Xiang, G., Jin, H., Jiang, W.: VRFPS: A Novel Virtual Machine-Based Real-time File Protection System. In: ACIS International Conference on Software Engineering Research, Management and Applications, pp. 217–224. IEEE, Haikou, China (2009)