Journal of
Sensor and Actuator Networks

MDPI

Article

OSSEC IDS Extension to Improve Log Analysis and Override False Positive or Negative Detections

Diogo Teixeira [1], Leonardo Assunção [1], Teresa Pereira [2], Silvestre Malta [3] and Pedro Pinto [4,*]

1 Instituto Politécnico de Viana do Castelo, 4900-347 Viana do Castelo, Portugal; diogoteixeira@ipvc.pt (D.T.); leonardoassuncao@ipvc.pt (L.A.)
2 Instituto Politécnico de Viana do Castelo, 4900-347 Viana do Castelo and Centro Algoritmi, Universidade do Minho, 4800-058 Guimarães, Portugal; tpereira@esce.ipvc.pt
3 Instituto Politécnico de Viana do Castelo, 4900-347 Viana do Castelo, Portugal and atlanTTic, Universidade de Vigo, E36310 Vigo, Spain; smalta@estg.ipvc.pt
4 Instituto Politécnico de Viana do Castelo, 4900-347 Viana do Castelo, ISMAI, and INESC TEC, 4200-465 Porto, Portugal
* Correspondence: pedropinto@estg.ipvc.pt; Tel.: +351-258-819-700

Received: 18 July 2019; Accepted: 10 September 2019; Published: 13 September 2019

Abstract: Intrusion Detection Systems (IDS) are used to prevent attacks by detecting potential harmful intrusion attempts. Currently, there are a set of available Open Source IDS with different characteristics. The Open Source Host-based Intrusion Detection System (OSSEC) supports multiple features and its implementation consists of Agents that collect and send event logs to a Manager that analyzes and tests them against specific rules. In the Manager, if certain events match a specific rule, predefined actions are triggered in the Agents such as to block or unblock a particular IP address. However, once an action is triggered, the systems administrator is not able to centrally check and obtain detailed information of the past event logs. In addition, OSSEC may assume false positive or negative detections and their triggered actions: previously harmless but blocked IP addresses by OSSEC have to be unblocked in order to reestablish normal operation or potential harmful IP addresses not previously blocked by OSSEC should be blocked in order to increase protection levels. These operations to override OSSEC actions must be manually performed in every Agent, thus requiring time and human resources. Both these limitations have a higher impact on large scale OSSEC deployments assuming tens or hundreds of Agents. This paper proposes an extension to OSSEC that improves the administrator analysis capability by maintaining, organizing and presenting Agent logs in a central point, and it allows for blocking or unblocking IP addresses in order to override actions triggered by false detections. The proposed extension aims to increase efficiency of time and human resources management, mainly considering large scale OSSEC deployments.

Keywords: IDS; OSSEC; cybersecurity; information security; attack detection; security events; intrusions

1. Introduction

Networks and systems connected to the Internet are exposed daily to world wide cyber threats. Every organization running servers and network devices are potential targets of multiple cyberattacks, resulting in serious impacts such as breakdowns, damage to resources and reputation, and financial losses. To prevent or mitigate the impact of these attacks, organizations implement security protection systems and platforms such as firewalls, Access Control Lists (ACL), Honeypots, Security Information and Event Management (SIEM), Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), among others. All these systems and platforms have different goals and, when used together, they

enable a more robust protection against infrastructure related attacks and also enable the safeguard of sensitive information.

An IDS such as the Open Source Host-based Intrusion Detection System (OSSEC) [1] is an important tool for a SysAdmin (Systems Administrator) to prevent potential threats to systems and data. The OSSEC IDS comprehends a Manager that analyzes event logs from a group of Agents and compares them against specific patterns set in the configured rules. In case OSSEC Agents events match a particular pattern of a configured rule, the OSSEC Manager issues predefined actions to the Agents, to be applied during a time interval. An example of these actions can be to block a potential harmful IP address or add an IP address to a specific list of restricted access devices for a defined time. After Agent logs are processed in the OSSEC Manager, they are deleted and maintained in each Agent.

There are specific scenarios where OSSEC presents limitations regarding false positive and false negative detections. The following items depict two scenarios where OSSEC assumes a false positive and a false negative detection with their respective triggered actions:

- Scenario 1—Email Service Down for Company X: The company X has a third-party email service protected by OSSEC. The X collaborators have two ways to access their emails; one is by using a webmail interface and the other one is through the use of an email standalone application. A collaborator named Mary decided to change her email password through a web interface, but she did not update her email standalone application. After insisting multiple times to access her email using the standalone application without success, the OSSEC triggered a protection rule, blocking the company X public IP address and leaving all her colleagues without the email service. The operation to unblock the IP address of company is out of the scope for the current OSSEC implementation. This scenario is considered an example of a false positive detection.
- Scenario 2—VoIP Services of Company Y under attack: A company Y provides his customers a VOIP service and, in a given period, it experiences an abnormal peak in traffic towards its two VoIP servers from a particular IP address. The OSSEC filters are triggered, since this action could likely be an attack to compromise the servers' availability. The OSSEC rules block the presumed attacker's IP address for both servers; however, the SysAdmin considers that the attack may extent to other servers. The operation to protect other servers is out of the scope for the current OSSEC implementation. This scenario is considered as an example of a false negative detection.

In the scenarios depicted above where events match a OSSEC rule and the predefined action is taken, two major limitations arise. First, since the processed logs are maintained only at each Agent, the SysAdmin is unable to centrally check at the Manager the specific event logs that matched the rule and triggered the action. Second, in case SysAdmin detects a false positive (scenario 1) or a false negative (scenario 2), instead of waiting the defined period of time, he would want to rapidly reestablish the previous operation state of the Agents, by overriding the Manager action, and unblock a specific IP address blocked in every Agent. To do so, a set of operations should be performed in each Agent by the SysAdmin or other element of the technical staff, and this intervention consumes time and human resources. The impact of this intervention is higher when OSSEC is deployed in large scale networks, where hundreds of Agents are deployed.

In this context, the paper proposes a novel extension to OSSEC to overcome the depicted limitations. The proposed extension simplifies the SysAdmin task by preserving and organizing Agent logs at the Manager, allowing to check them centrally using filters, even after they are processed. In addition, this extension allows for overriding specific false positive and negative detections and respective triggered actions, centrally in the Manager, in particular for actions involving blocking and unblocking IP addresses in one or multiple Agents. The proposed extension is deployed in the Manager and the Agents, and it includes a graphical user interface and a back-end platform.

In addition, the assessment of the proposed extension comprises a test environment deployed with OSSEC installed on a Manager and ten Agents. In this scenario, a set of events were triggered and the time elapsed for the Manager to analyze logs and revert actions was accounted. After deploying the

proposed extension, it was possible to obtain in the Manager useful information from Agent logs; in particular, the ones related to the events that triggered the IP address block in the Agents, thus reducing complexity and time elapsed in this analysis. In addition, the use of the proposed extension allowed to shorten the time to override particular OSSEC actions, to block or unblock specific IP addresses in the Agents. The impact of this elapsed time would be proportionally higher when multiple Agents are deployed.

The paper is structured as follows. In Section 2, the related work is presented. Section 3 describes in detail the OSSEC basic platform and the proposed extension. Section 4 presents the results and analysis. Section 5 presents the conclusions.

2. Related Work

Hosts, servers, databases and other resources are daily connected to the internet, in order to provide online services for their corporations. Being online, these resources are exposed to a number of sophisticated security threats which aim to compromise the availability of these systems, resulting in critical and financial impacts to the organizations. In order to protect the organizational technological resources against a set of cyberattacks such as tampering services, execution of malicious processes, DoS (Denial of Service) attacks, detecting port scans, and detecting strange network traffic patterns, an IDS can be deployed to monitor and control an infrastructure in a regular-basis. An IDS uses a single point to monitor a set of events in different Agents and, if needed, to reinforce their protections against cyberattacks.

The IDSs can be classified into two different types: the Host-Based Intrusion Detection System (HIDS) and Network-based Intrusion Detection System (NIDS). The HDIS has capabilities of monitoring and analyzing the internals of a computing system. Examples of this type of IDS are OSSEC [1], Tripwire [2], Fail2Ban [3] and Samhain [4]. The NDIS operates in a network where packets are captured and analyzed to enable the implementation of adequate protective measures. Examples of this type of IDS are SNORT [5] and Suricata [6]. There are IDSs that can be classified simultaneously as HIDS and NIDS such as the AlienOpen Source Security Information Management (OSSIM) [7].

Table 1 presents an overview of the main characteristics of a set of IDSs that support Linux, Windows and Mac OS operative systems. According to [1], OSSEC is an open source HIDS with multiple features such as log analysis, health check and adaptive rules. It comprehends a central Manager that collects multiple events in the Agents and, in case of a rule match, protection actions are performed in each Agent. From the HIDS presented, only OSSEC and Fail2ban include the Active Response feature that can change the environment to thwart or block potential attacks, i.e., this feature enables to block potential harmful IP addresses when specific events occur. However, Fail2ban does not maintain the logs in the manager, and it does not provide features such as integrity checking, registry monitoring, rootkit detection, and event correlation.

Recent research efforts assess the deployment of IDSs in different scenarios. According to Lin et al. [8], an HIDS combines two detection technologies, namely log file analysis and back propagation neural networks. Log file analysis allows for performing misuse detection and back propagation neural network to perform anomaly detection. The authors claim that these technologies combined allow for improving the efficiency of intrusion detection. In Singh et al. [9], Singh highlights the importance of having a security protection mechanism and also of understanding the reasons that justify the IP addresses blocking. Kumara et al. [10] propose an OSSEC system for virtualized environments to ensure the correct state of the virtual machines by detecting and eradicating rootkits. Additionally, in Reference [11], it is highlighted that IDSs such as OSSEC, Sguil [12] or Splunk [13] can be deployed to automate network monitoring tasks, correlate events and help to detect breaches in network security. In addition, it is stated that these IDS can be relevant to detect the so-called Advanced Persistent Threats, i.e., persistent threats from a well-coordinated and organized group of people with malicious intents and targeting specific organizations, governments or companies. In Reference [14] Jain et al. propose assessing an OSSEC system for reducing the encryption and decryption time of

the VPN network. Venkatesan et al. in Reference [15] propose preventing DDoS attacks by using the configuration features and rules adjustments of OSSEC and describes the operation of an algorithm used to distinguish the real from the false DDoS alerts.

Table 1. Comparison of multiple Intrusion Detection Systems (IDS).

Software	Type	Supported Features
OSSEC [1]	Host-based IDS (HIDS)	Log Analysis; Health Check; Windows Registry Monitoring; Rootkit Detection; Time Based Alert and Active Response; Adaptive Rules according to software installed on the Agent
Tripwire [2]	Host-based IDS (HIDS)	System integrity checking; Threats detection; Vulnerabilities Identification; Harden Configurations
Fail2ban [3]	Host-based IDS (HIDS)	Integrity checking, registry monitoring, rootkit detection, Event Correlation, Active response
Samhain [4]	Host-based IDS (HIDS)	File integrity checking, Log monitoring, Port monitoring, Rootkit detection
SNORT [5]	Network-based IDS (NIDS)	Distributed Denial of Service (DDoS) detection; stealth port scans detection; Common Gateway Interface (CGI) attacks detection; Server Message Block (SMB) probes detection; Operating System (OS) fingerprinting attempts detection
Suricata [6]	Network-based IDS (NIDS)	Real-Time Intrusion Detection; Inline Intrusion Prevention; Network Security Monitoring; Offline Packet Capture (PCAP) Processing
AlienVault OSSIM [7]	Host-based IDS (HIDS) & Network-based IDS (NIDS)	Asset discovery; Vulnerability assessment; Intrusion detection; Behavioral monitoring; Security information and event management (SIEM) event correlation

OSSEC presents a wide set of features and it is commonly deployed; however, it also presents limitations that have not been addressed. To extend the OSSEC capabilities, the open source solution Wazuh [16] can be deployed to enable features such as security monitoring of threat detection, integrity monitoring, incident response and compliance. The Wazuh extension for OSSEC provides the ability to centrally perform log analysis, add or remove agents and groups, clearing cache and obtain information about decoders and rules. Although the latest available versions of this extension for OSSEC allow for centrally blocking IP addresses in the Agents (useful when overriding false negative detections and their triggered actions), the Wazuh does not allow for overriding OSSEC actions triggered by false positive detections. In addition, according to [17], OSSEC has not presented many new features over the last decade, has been in maintenance mode for a long time and very little development work has been done, where the last releases consist mostly in bug fixes reported by occasional contributors.

In case the SysAdmin intends to control the OSSEC operation in the Manager through a Web interface or a User Interface (UI), he can rely on tools such as OSSEC Web UI plugin [18] or Analogi [19]. However, OSSEC Web UI plugin only allows for checking if the Agent is enabled or not and the project is not maintained now, and Analogi provides a simple dashboard with log graphical elements and filters. Yen et al. in Reference [20] refer to the fact that, in order to perform an adequate log analysis, it is important to provide and maintain these logs centrally. In Reference [21], Alqahtani et al. provide a base of comparison between Snort and OSSEC for virtual machine environments, regarding errors, normal logging and alerting, and attacks such as SQL injection. From the results obtained, this work claims that Snort is more effective than OSSEC in terms of log management and report. In this context, the SysAdmin can deploy more advanced tools to process and analyze logs such as the ELK-Stack [22], which includes Logstash, Elasticsearch and Kibana, but these tools need to be integrated while configured to the OSSEC environment.

Besides log analysis, OSSEC also has limitations on rules management and overriding actions. In Reference [23], the authors discuss the OSSEC limitation regarding its inability to centrally unblock IP addresses; the ultimate solution is to manually unblock them in each Agent. In addition, authors in

Reference [24] refer to the fact that performing troubleshooting tasks can lead to detect false positives in the IDS and, thus, in some cases, it is important to override the actions of an IDS such as to block or unblock an IP address.

This paper focuses on two particular OSSEC limitations with impacts on its scalability, namely the complexity of log analysis that are distributed by each Agent, and the inability to centrally block IP addresses not previously detected (i.e., a false negative classification by OSSEC) or unblock previously blocked IP addresses (i.e., a false positive classification by OSSEC). The following section details OSSEC components and extension proposal.

3. OSSEC Components and Extension Proposal

The basic OSSEC deployment comprises a set of running services in the Agents, which are constantly collecting logs that are locally saved and sent to the Manager. Predefined decoders and configured rules are checked against all Agents logs, and, according to the defined settings, actions are applied.

Figure 1 illustrates the functional blocks of an OSSEC deployment in a Manager, with the *OSSEC Manager*, and, in one or multiple Agents, with the *OSSEC Agent*. The *OSSEC Manager* is deployed in the Manager and contains the *OSSEC binaries* that represents the binary files of the *OSSEC Manager* service. The *Remoted* establishes a connection with each Agent to receive its logs or to send actions. When the Agents logs are received by *Remoted*, they are interpreted and decoded in the *Decode* block and, if they match a rule within *Rules* block, *Analysisd* signals an event to *Alerts* and sends the predefined action to the *Remoted*. The *OSSEC Agent* deployed in each Agent comprehends the *Logcollector*, *Syscheckd* and *Rootcheckd* blocks, which have the function of acquiring logs and performing a deeper security analysis on the local system. These three blocks send their logs to the *Agentd*, which, in turn, sends the data to the *OSSEC Manager*. In case the *OSSEC Manager* has an action configured for specific logs, the *Agentd* receives the instruction and executes the action using the *Execd*.

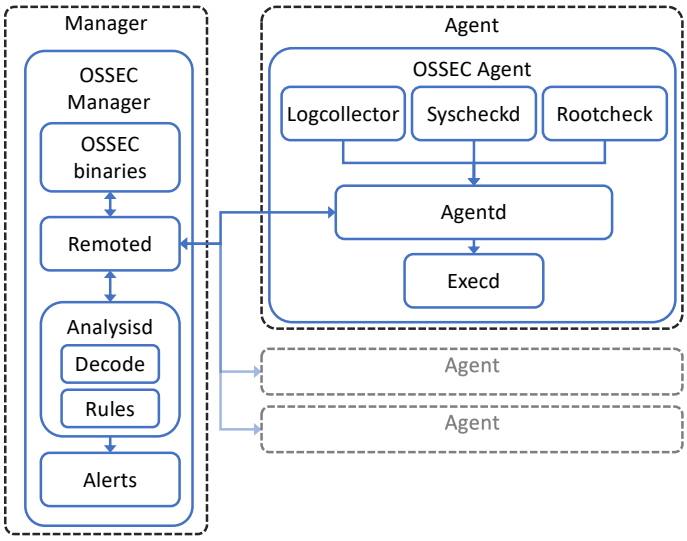

Figure 1. Open Source Host-based Intrusion Detection System (OSSEC) components.

The proposed extension is to be integrated in OSSEC base deployment and intends to tackle OSSEC limitations regarding the lack of detailed log analysis and the inability to centrally block and unblock IP addresses. In Figure 2, the OSSEC deployment architecture is presented with the proposed OSSEC extension and its necessary changes in the Manager and in the Agents.

In OSSEC Agents, the IP addresses are blocked by using security mechanisms such as IPTables, Firewalld and IPset. Thus, the *Execd* was changed in each OSSEC Agent to include a new script created to check IPtables rules and verify the current list of blocked IP addresses, each 30 s. In case differences

on the blocked IP addresses list are detected between samples, a log file is generated, sent to *Logcollector* to be later sent to the Manager by *Agentd*. This log file contains each blocked IP address and the total number of blocked IP addresses.

The OSSEC Manager receives the log file sent by *Agentd* at the *Remoted*, and forwards it to *Analysisd* to test the match of decoders and rules. In *Analysisd*, a new decoder in *Decode* was created in order to process the log file with a REGEX to save each blocked IP address in *Alerts*. This allows for knowing in the Manager which IP addresses are blocked in each Agent, a new feature of the current proposal. The *Alerts* block is connected to the OSSEC Extension, which is installed in the Manager and comprehends an ELK Stack and a Web UI. The ELKStack includes Logstash, Elasticsearch and Kibana UI, tools to improve the log collection, analysis and visualization. The Logstash is a dynamic data processing pipeline that collects the alert records already analyzed by the *OSSEC Manager*. These alert records are then processed by Elasticsearch, a fully scalable text search and analytics engine, and visualized by using Kibana UI, a graphical web interface aimed to explore and visualize recorded events in the Elasticsearch. In order to collect detailed information regarding Agents, the *Wazuh app* was also deployed within Kibana UI in the Web UI. The Kibana UI allows the administrator to check Agents status, i.e., offline or online status, the configuration group, and operating system version, and also allows for checking specific logs of a set of Agent logs by applying filters. The Web UI also includes the *Active-Response Manager UI* developed in PHP and HTML, and presents information about blocked IP addresses, lists the logs related to the blocking action, and also enables to block or unblock IP addresses on one or multiple Agents.

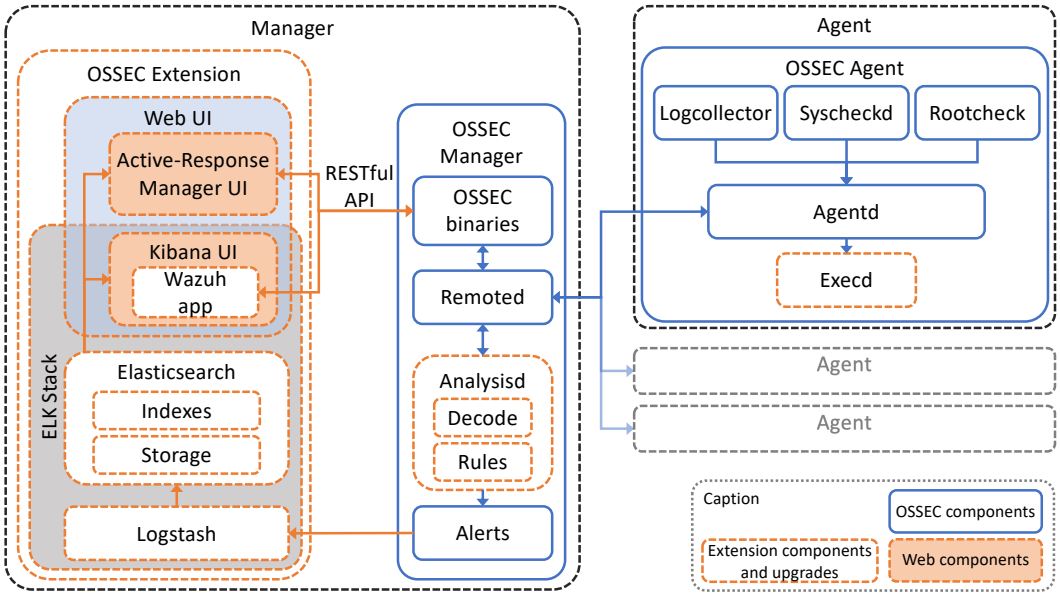

Figure 2. OSSEC components and upgrades.

In case the SysAdmin wants to block an IP address in an Agent, it uses the *Active-Response Manager UI* to trigger the execution of a script in the Manager with the procedures presented in Figure 3. The script starts by reading the IP address to be blocked and the Agent name. Then, it checks the list of IP addresses blocked on the *Alerts* in the Manager to test if the IP address is already blocked in that Agent. In case this IP address is already found as blocked, it logs the information and the procedures end. If the IP address is not found blocked in that Agent, the Manager connects to the Agent, confirms that the IP address is not blocked in IPtables and adds it. Then, the Manager disconnects from the Agent, logs this information and ends the procedures.

In case the SysAdmin wants to override an OSSEC action, and unblock an IP address in an Agent, it relies also on the *Active-Response Manager UI* to analyze the specific logs related to this IP and to

perform the unblocking procedures. The action to unblock IP addresses calls the execution of a script in the Manager with the procedures presented in Figure 4. The script starts by reading the IP address to be checked and unblocked and the Agent name. Then, it checks on the list of IP addresses blocked on the *Alerts* in the Manager, if the IP address is blocked in that Agent. If the IP address is not found as blocked in the Manager, a connection to the Agent is not necessary, a log is generated and the procedures end. In case this IP address is found blocked in that Agent, a log is generated, filtered logs for that IP address and Agent name are presented in the UI and a confirmation is requested to unblock the IP address in that Agent. In case the SysAdmin confirms the action, the Manager connects to the Agent, searches for that IP address in the block rules of IPtables Agent and deletes the rule related to that. Finally, it disconnects from the Agent and the procedures end.

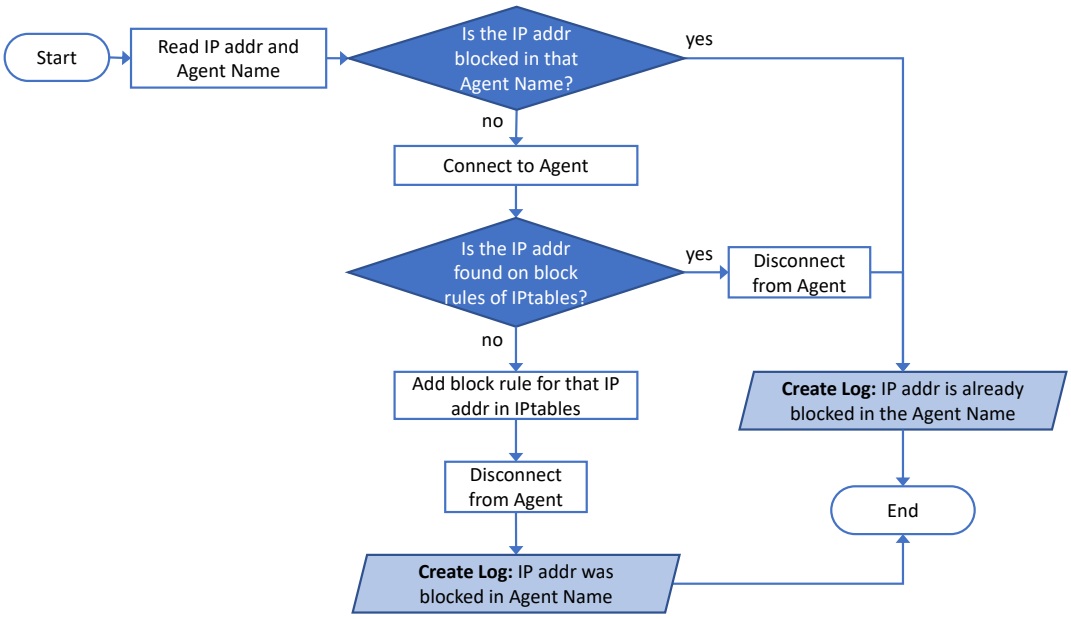

Figure 3. Procedures to block an IP address in an Agent.

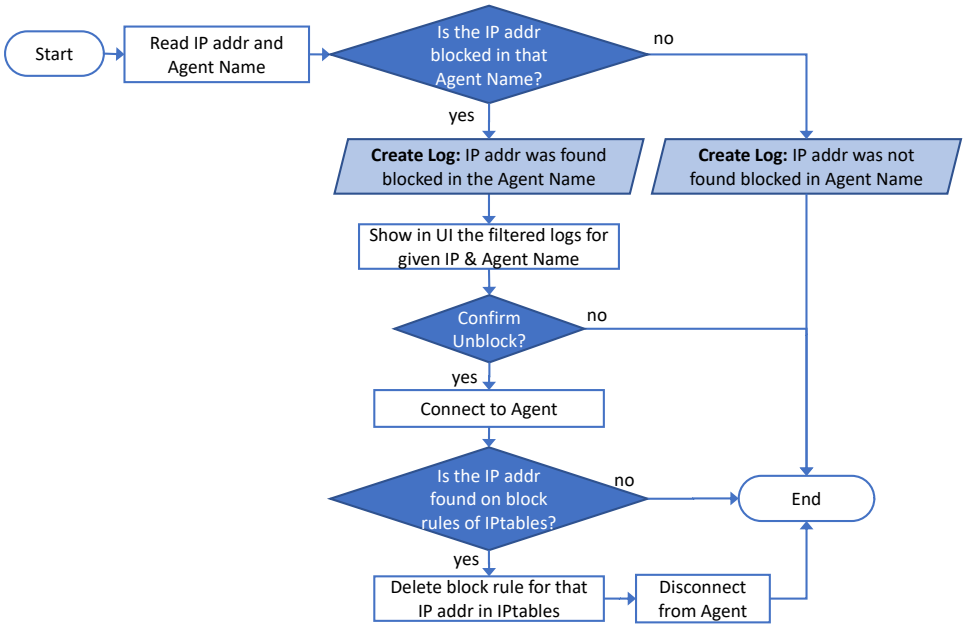

Figure 4. Procedures to check and unblock an IP addresses in an Agent.

The detailed log analysis and the action to block or unblock one or more IP addresses can be performed on one or multiple Agents by the system administrator in a central point, i.e., in the Manager. The results of the proposed OSSEC extension and changes in the Manager and in the Agents are presented in the next section.

4. Results and Analysis

The main interface with the OSSEC and the extension deployment is the Web UI, which includes the Kibana UI and the Active-Response Manager UI. In the basic configuration of OSSEC, the logs are maintained in each Agent and the OSSEC Manager basic operation uses syscheck or rootcheck services to obtain remote information about the Agents status and the logs file. By using the Kibana UI, the SysAdmin can check all the Agent logs in the Manager and collect Agent information and status (i.e., "Active", "Disconnected", "Never Connected"), the last time this information was obtained, the group to which each Agent belongs and the shared settings. In case a deep log analysis is required, the SysAdmin can apply a set of filters to the logs using specific Agent information, such as Agent name or group, rule number, IP address, date or service.

In Figure 5, an example is presented where the SysAdmin uses a filter with an Agent name (in this case, it was the selected "UCoIP-FTP"), and is able to verify a set of failed login attempts from an SSH logs service in the IP address 68.183.175.24.

Figure 5. Layout of the *Kibana UI*.

Active-response Manager UI was created and deployed in the Manager to support blocking or unblocking IPs in one or multiple Agents. This UI is protected by a login interface to allow only authorized personal with privilege access and all the performed actions in this UI are logged. After login, two options are available in the user menu: block IP address and Unblock IP address. In case the SysAdmin chooses the option "block IP", the UI will present a layout as in Figure 6. Here, two fields are required: Agent Name and IP address to block. In order to aid SysAdmin, the UI includes an autocomplete mode while typing the Agent name. When the block button is pressed, the fields are validated, the procedures to block the IP address on the Agent are performed, and the user of the UI is notified if the blocking action was successful or not. In case SysAdmin wants to block an IP in more than one Agent, each Agent is inserted and added to a list highlighted with a red box with the option to remove it, on the right side of the interface. In the example presented, the SysAdmin chooses to block IP 1.1.1.1 in three different Agents: "VPS-PG", "Mail-01" and "Project". In case this action is to be performed in all Agents, the SysAdmin has "all" options available.

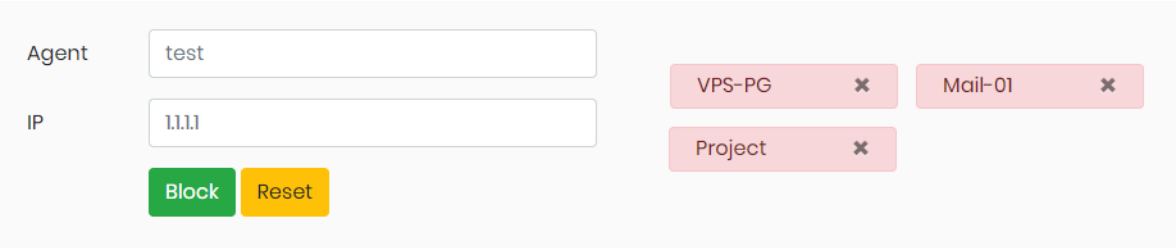

Figure 6. Layout of *Active-response Manager User Interface (UI)* to block IP.

In case SysAdmin chooses to "Unblock IP", the UI will present a layout as in Figure 7. The required fields for Agent name and IP address to unblock are similar to the block option, but, after entering the IP address and the Agent name, the user can use the "Verify" button to confirms if IP address is blocked in that Agent and to verify the last logs related to that IP address and the Agent name. In the example presented, the IP 111.207.49.184 is blocked in the "VPS-PG" Agent, and the last logs related to that IP are displayed. This logs show a set of login failed attempts for a SSH service and that an IP was blocked at 21:17:14. All this detailed information aids the SysAdmin decision prior to the unblock action. If SysAdmin proceeds and hits the "Unblock", the procedures to unblock are performed in that Agent.

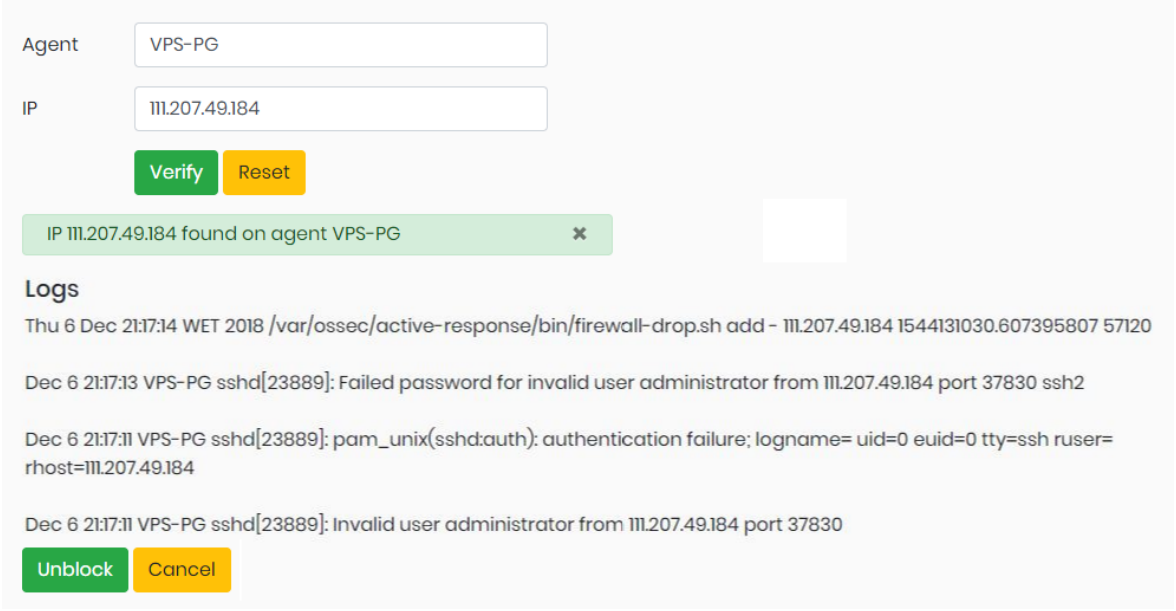

Figure 7. Layout of *Active-response Manager UI* to unblock IP.

Both UIs, Kibana UI and Active-response Manager UI, provide a range of new features to the SysAdmin. Previously in the OSSEC basic deployment, if SysAdmin wanted to check past logs for an IP in an Agent, he had to access each Agent, open a log file and search for specific events considered relevant. In addition, the SysAdmin was not able to manually block or unblock an IP address (overriding the OSSEC Manager actions). With the UI features presented above, the OSSEC Extension aids the SysAdmin in two stages: first, the extension presents detailed information and logs regarding Agents for deep analysis in the Manager and, second, the extension provides an interface that enables manually blocking or unblocking an IP address.

These UI interfaces and their back-end operations tackle the scenarios where OSSEC is implemented and where false positive or negative detections occur, demanding to block or unblock IP addresses. The scenarios such as the ones described in Section 1 were tested in a real environment and

time elapsed was accounted for each step. In the case of the false positive detection scenario (Scenario 1), the email service was down for Company X as its public IP address is blocked. The required steps to unblock the public IP address of company with and without the proposed extension were performed and time for each step was accounted. Without the proposed extension, the SysAdmin must perform the following steps with their respective estimate duration:

- Step 1: Connect to the email server and login (<1 min),
- Step 2: Confirm if the public IP address is blocked IPtables (<1 min),
- Step 3: Check the logs in detail on text-based files for the events occurred (2∼20 min depending on the log extension and analysis),
- Step 4: Issue a command to unblock the public IP address (<1 min),
- Step 5: Report that the issue is successfully resolved (<1 min).

With the proposed extension the required steps and their respective duration are:

- Step 1: Connect to the Proposed Extension UI and login (<1 min),
- Step 2: Select specific server and insert the public IP address (<1 min),
- Step 3: Check in the UI the specific logs related to the IP address blocked (1∼3 min depending on specific log analysis),
- Step 4: Click to unblock the public IP address (procedures to check and unblock IP address are performed as described in Figure 4) (<1 min),
- Step 5: Report that the issue is successfully resolved (<1 min).

As presented in Table 2, in this scenario, the estimated time elapsed without the proposed extension was between 6 and 24 min, and, with the proposed extension, the time elapsed was accounted between 5 and 7 min.

Table 2. Elapsed Times for Scenario 1—without and with the proposed extension.

Steps	Without the Proposed Extension		With the Proposed Extension	
	Min Elapsed Time (minutes)	Max Elapsed Time (minutes)	Min Elapsed Time (minutes)	Max Elapsed Time (minutes)
Step 1	1	1	1	1
Step 2	1	1	1	1
Step 3	2	20	1	3
Step 4	1	1	1	1
Step 5	1	1	1	1
Total	**6**	**24**	**5**	**7**

In the case of the false negative detection scenario (Scenario 2) described in Section 1, the VoIP services of company Y are under attack, and the required operations to protect other servers were performed and the time for each step was accounted. Without the proposed extension, the SysAdmin must perform the following steps with their respective estimate duration:

- Step 1: Connect to the VoIP server and login (<1 min).
- Step 2: Check all logs in detail on text-based files to obtain the potential harmful IP address (2∼20 min depending on the log extension and analysis).
- Step 3: For each server (Agent) to protect:
  - Step 3a: Connect to the server and login (<1 min),
  - Step 3b: Issue a command to block the potential harmful IP address (<1 min).
- Step 4: Report that the issue is successfully resolved (<1 min).

With the proposed extension, the required steps and their respective duration are:

- Step 1: Connect to the Proposed Extension UI and login (<1 min).

– Step 2: Check in the UI the specific logs related to obtain the potential harmful IP address (1–3 min depending on specific log analysis).
– Step 3: Select specific servers (by their IP address) or insert "all" and click to block the potential harmful IP address (<1 min).
– Step 4: Report that the issue is successfully resolved (<1 min).

As presented in Table 3, in this scenario and considering three Agents, the estimated time elapsed without the proposed extension was between 10 and 26 min and with the proposed extension the time elapsed was accounted between 4 and 6 min.

Table 3. Elapsed Times for Scenario 2—without and with the proposed extension.

Steps	Without the Proposed Extension		With the Proposed Extension	
	Min Elapsed Time (minutes)	Max Elapsed Time (minutes)	Min Elapsed Time (minutes)	Max Elapsed Time (minutes)
Step 1	1	1	1	1
Step 2	2	20	1	1
Step 3	2 min × 3 Agents	2 min × 3 Agents	1	3
Step 4	1	1	1	1
Total	**10**	**26**	**4**	**6**

According to the results obtained in these scenarios, the proposed extension allows for saving time and human resources. The results can be more expressive when we consider larger log files and higher complexity in log analysis, and more Agents.

The proposed extension can be used in other scenarios with an OSSEC IDS and multiple Agents to improve log analysis and to override false positive or negative detections, demanding to block or unblock one or a set of IP addresses. Future work may address to shorten the setup time and simplify rules' configurations in OSSEC and enhance the extension interfaces in design and user experience. In addition, additional work can be made in the proposed extension to reduce SysAdmin interactions with the IDS by using machine learning algorithms to recognize false positive or negative detections and automatically override their actions when necessary.

5. Conclusions

A wide range of sophisticated cyberattacks are performed against corporations and individuals. In corporations, the system administrator can use a combination of security mechanisms, such as firewalls, IDS, and SIEM, to prevent and mitigate the security attacks. In this context, the OSSEC IDS is commonly deployed since it presents a wide set of relevant features and it is composed of a Manager and a set of Agents. However, two particular limitations are found in the latest OSSEC version, which impact the scalability of this solution when using multiple Agents: (1) it is highly complex for the Manager to perform deep log analysis centrally, since logs are maintained in each Agent and there is no tool for their detailed filtering and analysis and (2) it is not possible to centrally override OSSEC actions taken by false positive or negative detections, e.g., to block or unblock one or multiple IP addresses in one or multiple Agents.

In this paper, a novel OSSEC extension is proposed to be deployed with the basic OSSEC IDS. This extension comprises changes in the Manager and in the Agents and, in the Manager, it includes an interface that presents detailed information regarding logs of Agents for deep analysis and enables manually blocking or unblocking one or multiple IP address, in one or multiple Agents.

As result, the proposed OSSEC extension increases this IDS scalability by enabling the system administrator to centrally perform deep analysis tasks and override specific actions as a result of false detections. The impact of the proposed extension is proportional to the number of Agents as these operations save time and human resource efforts to achieve a defined security level.

Author Contributions: P.P., T.P. and S.M. conceived the extension proposal; D.T. and L.A. implemented, configured and tested the extension proposal. D.T. and L.A. collected the data; All authors performed the data analysis and wrote the paper.

Acknowledgments: The authors wish to thank Instituto Politécnico de Viana do Castelo and the G9 Telecom company for providing the necessary resources and conditions for the development and the testing phases of this work.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. OSSEC—Open Source Host-Based Intrusion Detection System. Available online: https://www.ossec.net (accessed on 5 July 2019).
2. Tripwire. Available online: https://www.tripwire.com (accessed on 5 July 2019).
3. Fail2ban. Available online: https://www.fail2ban.org (accessed on 5 July 2019).
4. Samhain Labs | Samhain. Available online: https://la-samhna.de/samhain/ (accessed on 5 July 2019).
5. Snort—Network Intrusion Detection Prevention System. Available online: https://www.snort.org (accessed on 5 July 2019).
6. Suricata | Open Source IDS/IPS/NSM Engine. Available online: https://suricata-ids.org (accessed on 5 July 2019).
7. OSSIM: The Open Source SIEM | AlienVault. Available online: https://www.alienvault.com/products/ossim (accessed on 5 July 2019).
8. Lin, Y.; Zhang, Y.; Ou, Y.-J. The Design and Implementation of Host-Based Intrusion Detection System. In Proceedings of the 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, Jinggangshan, China, 2–4 April 2010; pp. 595–598. [CrossRef]
9. Singh, M.D. Analysis of Host-Based and Network-Based Intrusion Detection System. *Comput. Netw. Inf. Secur.* **2014**, *8*, 41–47. doi:10.5815/ijcnis.2014.08.06. [CrossRef]
10. Ajay Kumara, M.A.; Jaidhar, C.D. Hypervisor and virtual machine dependent Intrusion Detection and Prevention System for virtualized cloud environment. In Proceedings of the 2015 1st International Conference on Telematics and Future Generation Networks (TAFGEN), Kuala Lumpur, Malaysia, 26–28 May 2015; pp. 28–33. [CrossRef]
11. Vukalovic, J.; Delija, D. Advanced Persistent Threats - detection and defense. In Proceedings of the 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 25–29 May 2015; pp. 1324–1330. [CrossRef]
12. Sguil—Open Source Network Security Monitoring. Available online: https://bammv.github.io/sguil/index.html (accessed on 5 July 2019).
13. SIEM, AIOps, Application Management, Log Management, Machine Learning, and Compliance | Splunk. Available online: https://www.splunk.com (accessed on 5 July 2019).
14. Jain, R.K.; Trivedi, P. OSSEC Based Authentication Process with Minimum Encryption and Decryption Time for Virtual Private Network. In Proceedings of the 2016 8th International Conference on Computational Intelligence and Communication Networks (CICN), Tehri, India, 23–25 December 2016; pp. 442–445. [CrossRef]
15. Venkatesan, R.; Devi, D.R.; Keerthana, R.; Kumar, A.A. A Novel Approach For Detecting DDoS Attack in H-IDS Using Association Rule. In Proceedings of the 2018 IEEE International Conference on System, Computation, Automation and Networking (ICSCA), Pondicherry, India, 6–7 July 2018; pp. 1–5. [CrossRef]
16. Wazuh·The Open Source Security Platform. Available online: https://wazuh.com/ (accessed on 5 July 2019).
17. Migrating from OSSEC · Wazuh · The Open Source Security Platform. Available online: https://wazuh.com/migrating-from-ossec/ (accessed on 5 July 2019).
18. OSSEC-WUI. Available online: https://github.com/ossec/ossec-wui (accessed on 5 July 2019).
19. Analogi. Available online: https://github.com/ECSC/analogi (accessed on 5 July 2019).

20. Yen, T.F.; Oprea, A.; Onarlioglu, K.; Leetham, T.; Robertson, W.; Juels, A.; Kirda, E. Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks. In Proceedings of the 29th Annual Computer Security Applications Conference on—ACSAC '13, New Orleans, LA, USA, 9–13 December 2013; ACM Press: New York, NY, USA, 2013; pp. 199–208. [CrossRef]
21. Alqahtani, S.M.; Balushi, M.A.; John, R. An Intelligent Intrusion Prevention System for Cloud Computing (SIPSCC). In Proceedings of the 2014 International Conference on Computational Science and Computational Intelligence, Las Vegas, NV, USA, 10–13 March 2014; pp. 152–158. [CrossRef]
22. Elastic Stack—ELK-Stack. Available online: https://www.elastic.co/elk-stack (accessed on 5 July 2019).
23. Manually Unblock IP Blackisted by Active-Reponse before Timeout Expiration. 2012. Available online: https://marc.info/?l=ossec-list&m=135040227316697&w=2 (accessed on 5 July 2019).
24. Debian—OSSEC: Unblock an IP and Increase Threshold - Server Fault. Available online: https://serverfault.com/questions/356729/ossec-unblock-an-ip-and-increase-tresshold (accessed on 5 July 2019).