Chapter
Intrusion Alert Correlation Framework: An Innovative Approach

Huwaida Tagelsir Elshoush[1] and Izzeldin Mohamed Osman[2]

[1] Department of Computer Science, Faculty of Mathematical Sciences, University of Khartoum,
htelshoush@uofk.edu
[2] Sudan University of Science and Technology, Khartoum,
izzeldin@acm.org

Abstract. Alert correlation analyzes the alerts from one or more collaborative intrusion detection systems (IDSs) to produce a concise overview of security-related activity on a network. The process consists of multiple components, each responsible for a different aspect of the overall correlation goal. The sequence order of the correlation components affects the process performance. The total time needed for the whole process depends on the number of processed alerts in each component. An innovative alert correlation framework is introduced based on a model that reduces the number of processed alerts as early as possible by discarding the irrelevant and false alerts in the first phases. A new component, *shushing the alerts*, is added to deal with the unrelated alerts. A modified algorithm for fusing the alerts is presented. The intruders' intention is grouped into attack scenarios and thus used to detect future attacks. DARPA 2000 ID scenario specific datasets is used to evaluate the alert correlator model. The experimental results show that the correlation model is effective in achieving alert reduction and abstraction. The performance is improved after the attention is focused on correlating higher severity alerts.

Keywords: Alert correlation, Alert correlation datasets, Alert reduction, Collaborative intrusion detection systems, False alarm rate, Intrusion detection.

1 Introduction

Alert correlation is a process that contains multiple components with the purpose of analyzing alerts and providing high-level insight view on the security state of the network surveillance [13][14][15]. Correlation aims to relate a group of alerts to build a big picture of the attacks, hence can be used to trace an attack to its source.
The core of this process consists of components that implement specific function, which operate on different spatial and temporal properties [2].
The correlation components are effective in achieving alert reduction and abstraction. Research shows that the effectiveness of each component depends heavily on the nature of the data set analyzed [2]. Moreover, the performance of the correlation process is significantly influenced by the topology of the network, the characteristics of the attack, and the available meta-data [6].
Since alerts can refer to different kinds of attacks at different levels of granularity, the correlation process cannot treat all alerts equally. Instead, it is necessary to have a set of components that focus on different aspects of the overall correlation task. Some components, see Fig.1, e.g. those at the initial and second units, implement general functionality applicable to all alerts, independent of their type. Other components (e.g. in the third unit) are responsible for performing specific correlation tasks that cannot be generalized for arbitrary alerts, but for certain class of alerts.
Thus, one cannot, in general, determine a ranking among components with respect to their effectiveness. Each component can contribute to the overall analysis. Therefore, the most complete set of components should be used [2].

An innovative framework focuses on reordering the correlation components such that redundant, irrelevant and false alerts are reduced as early as possible thus reducing the number of processed alerts to enhance the performance. The unrelated alerts that are not correlated are dealt with in a separate component, *shushing the alerts*. Hence, the overall effectiveness of the correlation process is improved.

2 Overview of Improved Alert Correlation Framework

Our architecture, see Fig. 1, is composed of ten components: *normalization*, *preprocessing*, *prioritization*, *alert verification*, *alert fusion*, *focus recognition*, *shushing the alerts*, *multi-step correlation*, *intention recognition*, and *impact analysis* [22].

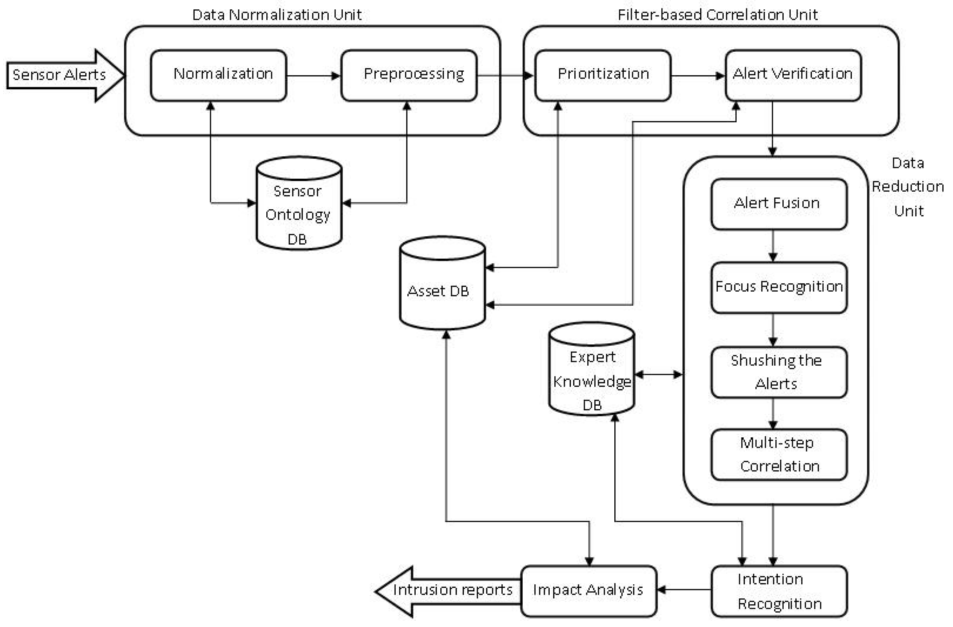

Fig. 1. The Improved Correlation Model.

In the *normalization* component, alerts that are generated by multiple IDSs are collected and stored in a database before they are modeled and converted into a standard format called Intrusion Detection Message Exchange Format (IDMEF). Then data *preprocessing* is required in order to clean the data, do feature extraction and selection, and finally deal with any incomplete or missing data.

The *filter-based correlation* unit either assigns a priority to each alert or identifies irrelevant alerts. Thus, alerts are ranked based on their severity level in order to distinguish between the high and low risks alerts depending on information in the asset DB. In the *alert verification* component, alerts are checked to find out the verifiable alerts, false positives and unverifiable alerts.

Redundant alerts are fused based on similarity functions [17] in the *alert fusion* component in the *data reduction unit*. In the *focus recognition* component, alerts are aggregated then classified using feature similarity. Unrelated and false alerts tend to be random and will not correlate, hence uncorrelated alerts are removed by *shushing the alerts* component. Lastly, *multi-step correlation*, is expected to achieve substantial improvement in the abstraction level and data reduction [5]. In this component, priori information of the network topology,

known scenarios, etc are provided by the expert knowledge DB; hence high level patterns are specified.
In the *intention recognition* component, relevant behavior is grouped into attack scenarios to extract attack strategy and plan recognition.
In the final component, *impact analysis*, the asset DB is consulted to determine all services that are dependent on a specific target. The heartbeat monitor checks whether all dependent services are still operational. If any service is failed, this information can be added to the alert as a likely consequence of the attack [2].
The asset DB stores information about installed network services, dependencies among services, and their importance to the overall operation of a network installation. So the DB does not represent an absolute measure of the importance of any asset, but rather reflect the subjective view of a security administrator. It is updated if there is any new information from the impact analysis or prioritization components.
The *knowledge DB* is a complete repository including all the necessary information about attacks, vulnerabilities, and the topological and configuration information about the protected networks and hosts.

3 Evaluation of Alert Correlators

3.1 Evaluating Alert Correlators

The effectiveness of IDS sensors are evaluated using the detection rate and the false positive rate. These values cannot be easily calculated for an alert correlator as [12]:

- the false positive is not clearly defined for a correlator, as it may receive a false positive (which is an IDS sensors output) and therefore draws a wrong assumption that an attack took place. Thus, validation of the input alerts is necessary.
- if there exists an attack creating a single low-level alert, no correlation will be performed and thus this may be considered a missed attack and hence results in a reduction in the detection rate.

According to [2], the evaluation of the correctness of the correlation process has to be performed manually as:

- there exist very few shared datasets for correlation evaluation, and
- generating alerts from raw data might be risky and may bias the correlation evaluation process,
- no truth files are associated with datasets, which makes it difficult to know if the correlated alerts are representative of a meaningful grouping of detected attacks.

Correlation tools are evaluated “as a whole”, without an assessment of the effectiveness of each component of the analysis process. As a result, it is not clear if and how the different parts of the correlation process contribute to the overall goals of correlation [2]. Recent research proved that the correlation components are effective in achieving alert reduction and abstraction.

3.2 Datasets

The available datasets, e.g. the DARPA-sponsored Cyber Panel Correlation Technology Validation (CTV) effort 2002 and the Defcon 9 dataset, have problems in evaluating the effectiveness of the components of the correlation process [12].

- To assess correlation systems, the sensor alerts output of the IDSs is needed, therefore these datasets are suitable to evaluate IDSs and not correlation tools. Thus, IDS alerts have to be generated by running specific sensors on the collected event streams. As a result, the alert stream associated with a dataset may be different if different sensors are utilized or if the configuration is different. This, in turn, makes it harder to compare the correlation systems’ results.
  Hence, the effectiveness of correlation is not only highly dependent on the nature of the datasets alone, but also on the different sensors used and the different configuration, if IDSs datasets are used.

– Alert verification should be done in real time, and this is hindered by the offline nature of these datasets.
– The lack of a mission model and its relationship to the network assets hinders the impact analysis.
– To determine the actual impact of the attacks, a health monitoring is needed but not usually provided.
– All presented datasets are not real world traffic.
– Finally, also these datasets have no internet anomalous traffic as none were recorded on a network connected to the internet.

3.3 Correlation Evaluation Truth Files

The correlation evaluation truth file should be the high-level plan behind the attacks in the test data, where the plan should describe for example how each low-level alert is related to other alerts [12]. Some problems are:
– Representing this information is difficult, as there is no good definition of a high-level plan and also no standardized way to express such a plan exists.
– It is difficult to compare the correlator output to the plan contained in the truth file, as two widely different high-level views of an attack might both be correct.

3.4 Factors Affecting the Alert Reduction Rate

When testing correlation systems, it is important to calculate reduction rates for different datasets. Furthermore, it is useful to calculate the reduction rate during each correlation step in addition to the total reduction rate as [12].

– experiments showed that the achieved alert reduction rate (RR) is highly dependent on the features of the dataset being processed, and
– a particular dataset that experience a high reduction rate during one correlation step might achieve poor reduction rates during other steps.

4 Implementation and Experimental Results

4.1 Experiments on DARPA 2000 Datasets

DARPA 2000 [8] is a well-known IDS evaluation dataset created by the MIT Lincoln Laboratory. It consists of two multistage attack scenarios, namely LLDOS 1.0 and LLDOS 2.0.2. Both attack scenarios contain a series of attacks in which an attacker probes, breaks-in, installs the components necessary to launch a Distributed Denial of Service (DDoS) attack against an off-site server, with different stealth levels. LLDOS 2.0.2 is a bit more sophisticated than LLDOS 1.0.
In both scenarios, the attacker tries to use the vulnerability of Sadmind RPC service and launches buffer overflow attacks against the vulnerable hosts. The attacker installs the mstream distributed DOS software after he breaks into the hosts successfully. Finally, the attacker launches DDOS attacks from the victims. The differences between these two scenarios lay in two aspects: First, the attacker uses IPSweep and Sadmind Ping to find out the vulnerable hosts in LLDOS 1.0 while DNS HInfo is used in LLDOS 2.0.2. Second, the attacker attacks each host individually in LLDOS1.0, while in LLDOS2.0.2, the attacker breaks into one host first and then fans out from it [10][16].
Each scenario includes network traffic collected from both the demilitarized zone (DMZ) and the inside part of the evaluation network. We performed eight experiments, four on the improved model and four on the comprehensive approach model.

Table 1. Impact of Preprocessing Component on LLDOS Scenarios

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	891	922	430	494
Output alerts	886	922	425	489

4.2 Analysis and Performance Evaluation of the Improved Correlation Components

In this section, each component's function of the innovative alert correlation framework is explained in details, together with the implementation of the model based on the improved framework. In the implementation, Microsoft SQL Server 2005 was used as the relational database to store the alert datasets, the intermediate data, and the analysis results of each component as well as the correlated alerts. Programs written in C#, Microsoft Visual Studio 2010, were created to implement the correlation components' functionalities. The alert log files generated by RealSecure IDS of the DARPA simulation network is used [9].

Data Normalization Unit

The data normalization unit contains two components, namely the *normalization* and the *preprocessing* components.

- **Normalization**
  The interaction of collaborators from distinct IDSs poses various requirements. Dependent on the level of collaboration, these include a common language, protocol or even a complete framework. Hence, in a distributed environment with heterogeneous IDSs, specifying a common alert format is fundamental for providing interpretability among IDSs. Moreover, high level analysis such as alert correlation also requires that the alerts that are processed to be in a generic format. There exists a variety of exchange formats; prominent examples include IDMEF and IODEF. Therefore, in this component, all attributes of each sensor alert will be translated into a common data format, IDMEF in particular [3][8]. The IDMEF data model is implemented using a Document Type Definition (DTD) to describe Extensible Markup Language (XML) documents. IDMEF is also an object-oriented representation and a Unified Modeling Language (UML) model. If only one type of sensor was used, then that sensor's format could be used as the standard format. DARPA alerts were normalized in IDMEF standard format.

- **Preprocessing**
  The goal of this component is to supply, as accurately as possible, missing alert attributes that are important for later correlation components. Reducing the dimensionality of the data improves the correlation process considerably and detection accuracy is improved as a result of removing irrelevant and redundant features. Hence, this component handles null values, missing and incomplete data [21]. Feature selection is used to reduce the alert attributes, as it is revealed from recent research [18][19][20] that it improves detection accuracy and performance.
  The type information is useful because it allows one to group attacks with a similar impact together.
  In both scenarios, there are 45 features, of which only 7 features were extracted, namely EventID, timesec, SrcIPAddress, DestPort, DestIPAddress, OrigEventName, and SrcPort. The date attribute was represented in date/time format, and I converted it to time in seconds (represented as timesec). 5 alerts, representing incomplete data, were removed in all datasets, except for the inside segment of scenario 1.0., see Table 1.

Filter-based Correlation Unit

The primary goal is to reduce the number of alerts to be correlated by eliminating false, irrelevant and low risk alerts. False alerts need to be handled at an early stage as they will have negative impact on the correlation result, and moreover the number of processed alerts will be greatly reduced.

Table 2. Impact of Prioritization Component on LLDOS Scenarios

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	886	922	425	489
Output alerts	188	167	54	71
Reduction Rate	78.78%	81.89%	87.29%	85.48%

- **Prioritization**
  In this component, depending on the information contained in the asset DB, high and low risks alerts are identified. It takes into account the security policy and the security requirements of the site where the correlation system is deployed. The low risks that do not have significant effect on the protected system are discarded. By alert ranking, the data fed into the remaining components is reduced as only the high and medium risk alerts are considered in the later components. Thus as acknowledged in [1], when the number of processed alerts is reduced, the performance is improved as the total time needed for the whole process depends on the number of processed alerts in each component.
  The ranking/priority of alerts of LLDOS scenarios from [7] is used. It is a very naive alert prioritization policy just to give a rough idea of how much reduction this step could achieve, but in most cases a much more complex policy is needed. Thus low risk alerts are discarded, and only the medium and high risk alerts are sent to the next component. Table 2 shows the implementation results.

- **Alert Verification**
  The verification component issue a non-relevant positive attack. Thus, it distinguishes between successful and failed intrusion attempts.
  The verification of an attack can be accomplished by extending ID rules with an expected "outcome" of the attack that describes the visible and verifiable traces that a certain attack leaves at a host or on the network [2][6].
  Verification can be performed using both *passive* and *active* techniques, and each approach requires different support from the ID infrastructure [2][6][12].
  Passive verification mechanisms can be done by storing the actual security status of the network in the knowledge base.
  Active alert verification is a technique designed to reduce the false positive rate of IDSs by actively probing for a vulnerability associated with detected attacks. If the vulnerability corresponding to a detected attack is found to exist in the host or network against which the attack was directed, the alert is generated, invoking any logging and response functions as normal. If, however, the vulnerability is determined not to exist, the alert is considered a false positive and is suppressed.
  The alert verification process requires that the protected assets be available for real-time verification of the actual exposure of the system and/or that a detailed model of the installed network services be available. Unfortunately, this information is not available for the data set analyzed and there is no sufficient information found about the asset DB, so this component could not be implemented.

Data Reduction Unit

Similar alerts are fused and thus data is reduced by eliminating data redundancies, and irrelevant, false and unreal alarms using alert correlation. False alerts are usually less likely to be correlated using alert correlation.

- **Alert Fusion**
  It combines a series of alerts that refer to attacks launched by one attacker against a single target. This component removes duplicates created by the independent detection of the same attack by different sensors, and also correlates alerts that are caused by an attacker who tests different exploits against a certain program or that runs the same exploit multiple times to guess correct values for certain parameters (e.g., offsets and memory addresses for buffer overflow) [2][6][12].

The alert fusion component keeps a sliding timewindow of alerts. The alerts within the timewindow are stored in a time-ordered queue. When a new alert arrives, it is compared to the alerts in the queue, starting with the alert with the earliest timestamp.
A *fusion* match is found if all overlapping attributes are equal and the new alert is produced by a different sensor. The timestamp of the meta-alert is assigned the earlier of the sub-alerts times. The later timestamp simply indicates a delayed detection by the other sensor. On the other hand, attack *threads* are constructed by merging alerts with equivalent source and target attributes that occur in a certain temporal proximity but the alerts need not be produced by different sensors. The timestamp of the meta-alert is assigned the earlier of the two start-times and the later of the two end-times.
The value of the time window should be a good trade-off between a small value, which would cause several attack threads to go undetected, and a larger value, which would slow down the system by requiring the component to keep a large number of alerts in the queue.

Algorithm 1 is the alert fusion component method [22].

Algorithm 1 Alert Fusion

```
Parameter window-size, fuse-window, thread-window
Global alert-queue, fuse, thread

fuse(alert)
al ← get a:alert with lowest start-time from alert-queue where
if alert.analyzer ∩ a.analyzer is empty and all overlapping attributes except start-time,
end-time, analyzer, alertid are equal then
    fuse
    window-size = fuse-window
else
    if alert.victimhosts = a.victimhosts and alert.attackerhosts = a.attackerhosts then
        thread
        window-size = thread-window
    end if
end if
if al¬null then
    replace al in alert-queue with fuse-merge(alert, al)
else
    add alert to alert-queue
    remove all a:alert from alert-queue where
    a.start-time < (alert.start-time - window-size)
    pass removed alerts to next correlation component
end if

fuse-merge(alert1, alert2)
r ← new alert
r.alertid ← get unique-id()
r.start-time ← min(alert1.start-time, alert2.start-time)
r.reference ← (alert1.alertid ∪ alert2.alertid)
if fuse then
    r.end-time ← min(alert1.end-time, alert2.end-time)
    for each attr:attribute except start-time, end-time, reference, alertid do
        r.attr ← alert1.attr ∪ alert2.attr
    end for
    fuse ← false
else
    if thread then
        r.end-time ← max(alert1.end-time, alert2.end-time)
        r.analyzer = alert1.analyzer ∪ alert2.analyzer
        thread ← false
    end if
end if
if alert1.name = alert2.name then
    r.name ← alert1.name
else
    r.name ← "Attack Thread"
end if
for each attr:attribute except start-time, end-time, reference, analyzer, alertid do
    if alert1.attr = alert2.attr then
```

Table 3. Impact of Alert Fusion Component on LLDOS Scenarios

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	188	167	54	71
Output alerts	92	110	34	45
Reduction Rate	51.06%	34.13%	37.04%	36.62%

Table 4. Impact of Focus Recognition Component(one-to-many)

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	92	110	34	45
Output alerts	42	57	23	27
Reduction Rate	56.52%	48.18%	32.35%	40%

Table 5. Impact of Focus Recognition Component(many-to-one)

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	42	57	23	27
Output alerts	31	28	5	24
Reduction Rate	26.19%	50.88%	78.26%	11.11%

```
            r.attr ← alert1.attr
        else
            r.attr ← null
        end if
    end for
    return r
end
```

There were two sensors in DARPA data sets, but all the alerts generated by one of the sensors contained null and incomplete values and thus were removed by the preprocessing component. Thus, there were no fusion in the data set used as all traffic injected into this component were seen by one sensor, but there were thread reconstruction. Table 3 shows the results of the implementation.

– **Focus Recognition**
This component has the task of identifying hosts that are either the source or the target of a substantial number of attacks. This is used to identify denial-of-service (DoS) attacks or port scanning attempts. It aggregates the alerts associated with single hosts attacking multiple victims (called a one-to-many scenario) and single victims that are targeted by multiple attackers (called a many-to-one scenario).
The one-to-many scenario has two tunable parameters: the size of the timeout, which is used for the initial window size, and the minimum number of alerts for a meta-alert to be generated. On the other hand, the many-to-one scenario has three tunable parameters: the first two are the same as for the one-to-many scenario. The third parameter is the number of meta-alerts required before a many-to-one alert is labeled as a denial-of-service attack [2][6][12].
Specifically, a many-to-one attack is classified as a DDoS attack when the total number of attacks against a victim exceeds a user-defined limit. A one-to-many alert is classified as a horizontal scan when a single port is scanned on all victim machines, or as a horizontal multiscan when more than one port is scanned [2].
In the carried out experiments, the minimum number of alerts in a meta-alert was two. We first applied one-to-many focus recognition on DARPA datasets, then followed by many-to-one focus recognition. Some horizontal scan and multi-scan attacks were observed. Tables 4 and 5 show the reduction rates. DMZ in scenario 2.0.2 shows a great RR as is expected being a multistage attack scenario.

Table 6. Impact of Shushing the alerts Component on LLDOS Scenarios

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	31	28	5	24
Output alerts	6	7	3	5
Reduction Rate	80.65%	75%	40%	79.17%

Table 7. Impact of Multi-step Correlation Component on LLDOS Scenarios

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	6	7	3	5
Output alerts	6	5	2	4
Reduction Rate	0%	28.57%	33.33%	20%

– **Shushing the Alerts**
  As shown in [10], alert correlation can be used to differentiate between false and true alerts. False alerts and unreal alarms tend to be more random than actual alerts, and are less likely to be correlated. Thus, based on this founding, we intentionally removed the alerts that are not correlated in the alert fusion and focus recognition, resulting in Table 6, which shows great reduction rates.

– **Multi-step Correlation**
  The goal of this component is to identify high-level attack patterns that are composed of several individual attacks. The high-level patterns are usually specified using some form of expert knowledge [2][11][12][17].
  Specifically, it may also associate network-based alerts with host-based alerts that are related to the same attack, called *attack session reconstruction*. It automatically links a network-based alert to a host-based alert when the victim process listens to the destination port mentioned in the network-based alert. This requires either real-time access to the systems being protected or very detailed auditing information in order to map network traffic to host activity. Identifying relations between these two types of alerts is difficult because the information that is present in the alerts differs significantly [2].
  While multistep attack analysis may not generate the same level of alert reduction achieved by other components, it often provides a substantial improvement in the abstraction level of the produced meta-alerts. Newly detected attack strategies are fed back to the expert knowledge DB to keep it up-to-date.
  Relying on the information in [16], attack patterns are identified, and used to implement this component resulting in Table 7.

Intention Recognition

Intention or plan recognition is the process of inferring the goals of an intruder by observing his/her actions [4]. It deduces strategies and objectives of attackers based on attack scenarios that are output by correlation systems. Failed attacks can be useful to know so to be avoided in the future. Using alert correlation, the intruders' relevant behavior can be grouped into attack scenarios, and later on, their attack strategy or plan can be extracted and fed back to update the expert knowledge DB.
Inadequate information of attack strategies or plans of intruders in the data set used hindered the implementation of this component.

Impact Analysis

This component contextualizes the alerts with respect to a specific target network. It combines the alerts from the previous correlation components with data from an asset DB and a number of heartbeat monitors to determine the impact of the detected attacks on the operation of the monitored network and on the assets that are targeted by the attacker. Thus, it requires a precise modeling of the relationships among assets in a protected network and

Table 8. Total Alert Reduction for the Improved Model

	DMZ 1.0	Inside 1.0	DMZ 2.0.2	Inside 2.0.2
Input alerts	891	922	430	494
Output alerts	6	5	2	4
Reduction Rate	99.33.%	99.46%	99.53%	99.19%

constant health monitoring of those assets. Hence, insufficient information of asset DB of LLDOS scenarios deters the implementation.

4.3 Summary of Experimental Results

Hereby, we summarize the results of our experiments.

The Total Alert Reduction of the Improved Model

Table 8 shows the total alert reduction for each dataset. Fig. 2 shows the effect of the improved correlation model on LLDOS 1.0 and 2.0.2 scenarios. There is a substantial drop in the number of alerts in the priority component for all datasets. This reduces the number of processed alerts considerably and thus improves the correlation process performance.

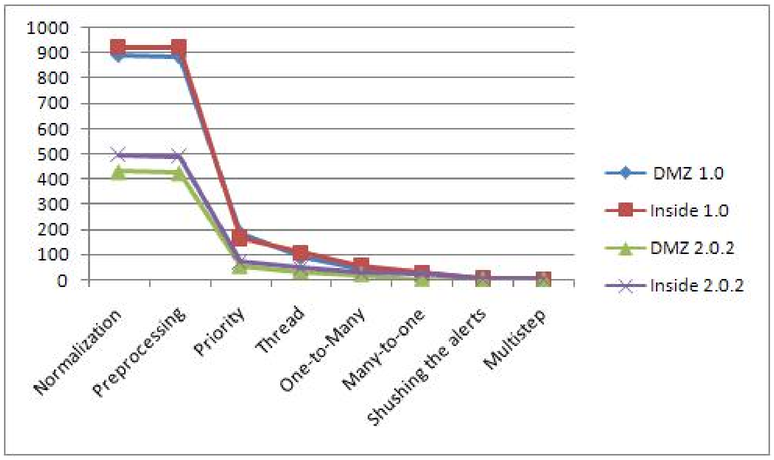

Fig. 2. Effect of Improved Correlation Model on LLDOS Scenarios 1.0 and 2.0.2.

Comparison of the Performance of the Improved Model with the Comprehensive Approach on LLDOS Scenario 1.0

Tables 9 and 10 show the number of processed alerts in each component for scenario 1.0 for the improved model compared to the Comprehensive approach discussed in [2]. Since the processing time is proportional to the number of processed alerts, hence Fig. 3 shows that the improved model gives better results.

Comparison of the Performance of the Improved Model with the Comprehensive Approach on LLDOS Scenario 2.0.2

Tables 11 and 12 show the number of processed alerts in each component for scenario 2.0.2 for the improved model and the Comprehensive approach [2] respectively. The graph of Fig. 4 assured the affirmation of the better performance of the innovative improved model.

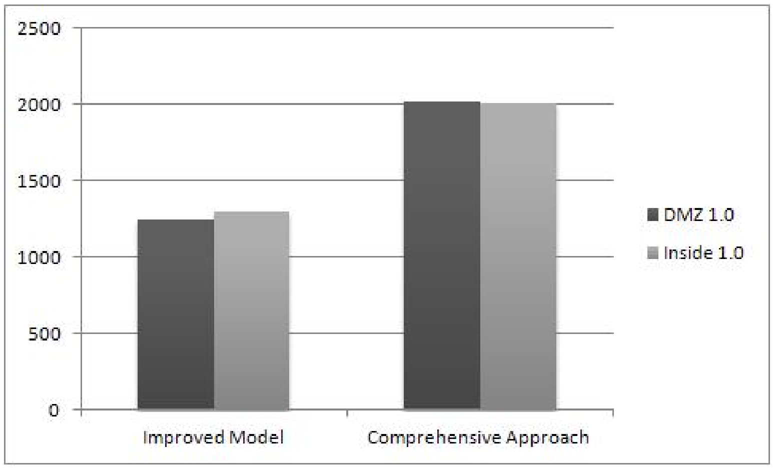

Fig. 3. Comparison of Processing Time of Improved Correlation Model and Comprehensive Approach on LLDOS Scenario 1.0.

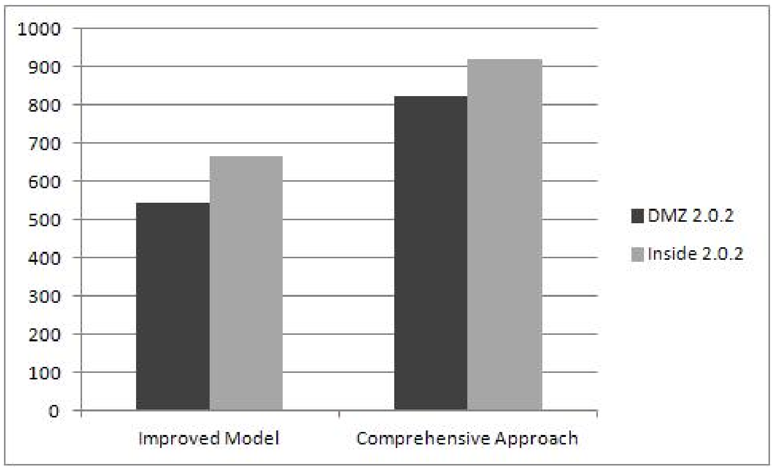

Fig. 4. Comparison of Processing Time of Improved Correlation Model and Comprehensive Approach on LLDOS Scenario 2.0.2.

Table 9. No. of Processed Alerts using Improved Model for Scenario 1.0

	Prepr.	Prio.	Fus.	1:M	M:1	Shush.	Multi	total
DMZ	886	188	92	42	31	6	6	1251
Inside	922	167	110	57	28	7	5	1296

Table 10. No. of Processed Alerts using Comprehensive Approach for Scenario 1.0

	Prepr.	Fus.	1:M	M:1	Multi.	Prio.	total
DMZ	886	619	208	175	118	13	2019
Inside	922	622	193	151	107	16	2011

Table 11. No. of Processed Alerts using Improved Model for Scenario 2.0.2

	Prepr.	Prio.	Fus.	1:M	M:1	Shush.	Multi	total
DMZ	425	54	34	23	5	3	2	546
Inside	489	71	45	27	24	5	4	665

Table 12. No. of Processed Alerts using Comprehensive Approach for Scenario 2.0.2

	Prepr.	Fus.	1:M	M:1	Multi.	Prio.	total
DMZ	425	241	63	46	44	5	824
Inside	489	276	71	44	33	7	920

References

1. Taha, A. E., Abdel Ghaffar I., Bahaa Eldin, A. M., Mahdi, H. M. K.: Agent Based Correlation Model For Intrusion Detection Alerts. IEEE Computer Society (May 2010).
2. Valeur, F., Vigna, G., Kruegel, C., Kemmerer, R.: A Comprehensive Approach to Intrusion Detection Alert Correlation. IEEE Transactions on Dependable and Secure Computing, the IEEE Computer Society, **1**, Issue No. 3, 146–169 (July-September, 2004).
3. Debar, H., Curry, D., Feinstein, B.: The Intrusion Detection Message Exchange Format (IDMEF), 2007.
   http://www.ietf.org/rfc/rfc4765.txt.
4. Ghorbani, A. A., Lu, W., Tavallaee, M.: Network Intrusion Detection and Prevention: Concepts and Techniques. Springer, a textbook (2010).
5. Yusof, R., Selamat, S. R., Sahib, S.: Intrusion Alert Correlation Technique Analysis for Heterogeneous Log. International Journal of Computer Science and Network Security (IJCSNS), **8** Issue No. 9, 132–138 (Sept. 2008).
6. Valeur, F.: Real-time ID Alert Correlation. PhD Thesis (June 2006).
7. Siraj, M. M., Maarof, M. A., Hashim, S. Z. M.: Intelligent Alert Clustering Model for Network Intrusion Analysis. Published by ICSRS Publication, Int. J. Advance. Soft Comput. Appl., **1**, Issue No. 1 (July 2009), ISSN 2074-8523.
8. MIT Lincoln Laboratory 2000 DARPA Intrusion Detection Scenario Specific Datasets. http://www.ll.mit.edu/index.html.
9. Ning, P. TIAA: A Toolkit for Intrusion Alert Analysis (2007).
   http://discovery.csc.ncsu.edu/software/correlator/.
10. Ning, P., Cui, Y., Reeves, D. S.: Constructing Attack Scenarios through Correlation of Intrusion Alerts. In Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington D.C., pp. 245–254 (November 2002).
11. Ning, P., Cui, Y., Reeves, D. S.: Analyzing Intensive Intrusion Alerts Via Correlation. In Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002), LNCS 2516, Zurich, Switzerland, pp. 74–94 (October 2002).

12. Kruegel, C., Valeur, F., Vigna, G.: Intrusion Detection and Correlation - Challenges and Solutions. Springer, a textbook (2005).
13. Zhou, C. V., Leckie, C., Karunasekera, S.: Decentralized multidimensional alert correlation for collaborative intrusion detection. Elsevier Ltd., Journal of Network and Computer Applications, **32**, 1106–1123 (Sept. 2009).
14. Zhou, C. V., Leckie, C., Karunasekera, S.: A survey of coordinated attacks and collaborative intrusion detection. Elsevier Ltd., Computer Security, pp. 1–17 (June 2009).
15. Bye, R., Camtepe, S. A., Albayrak, S.: Collaborative Intrusion Detection Framework: Characteristics, Adversarial Opportunities and Countermeasures (August 2010).
16. Cui, Y.: A Toolkit for Intrusion Alerts Correlation Based on Prerequistes and Consequences of Attacks. MSc. Thesis (Dec. 2002).
17. Elshoush, H. T., Osman, I. M.: Alert Correlation in Collaborative Intelligent Intrusion Detection Systems - A Survey. Elsevier Ltd., Journal of Applied Soft Computing, **11**, Issue No. 7, 4349–4365 (October 2011).
18. Zainal1, A., Maarof1, M. A., Shamsuddin, S. M.: Features Selection Using Rough-PSO in Anomaly Intrusion Detection (2007).
19. Zainal, A., Maarof, M. A., Shamsuddin, S. M.: Feature Selection Using Rough Set in Intrusion Detection.
20. Amiri, F., Yousefi, M. M. R., Lucas, C., Shakery, A.: Improved Feature Selection for Intrusion Detection System. Elsevier Ltd., Journal of Network and Computer Applications (Jan. 2011).
21. Davis, J. J., Clark, A. J.: Data Preprocessing for Anomaly-based Network Intrusion Detection: A Review. Elsevier Ltd., Journal of Computer and Security, 353–375 (October 2011).
22. Elshoush, H. T., Osman, I. M.: An Improved Framework for Intrusion Alert Correlation. Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2012, WCE 2012, 4-6 July, 2012, London, U.K., pp. 518–523.